Amazon

(AWS Certified SysOps Administrator - Associate)

AWS Certified SysOps Administrator - Associate (SOA-C02)

Total: **477 Questions**

Link:

Question: 1

A company has an infernal web application that runs on Amazon EC2 instances behind an Application Load Balancer. The instances run in an Amazon EC2 Auto
Scaling group in a single Availability Zone. A SysOps administrator must make the application highly available. Which action should the SysOps administrator take to meet this requirement?

A. Increase the maximum number of instances in the Auto Scaling group to meet the capacity that is required at peak usage.

B. Increase the minimum number of instances in the Auto Scaling group to meet the capacity that is required at peak usage.

C. Update the Auto Scaling group to launch new instances in a second Availability Zone in the same AWS Region.

D. Update the Auto Scaling group to launch new instances in an Availability Zone in a second AWS Region.

Answer: C

Explanation:

The goal is to achieve high availability for a web application currently running in a single Availability Zone (AZ). High availability inherently means the application should be resilient to failures within a single AZ.

Option A only addresses capacity, not availability. Increasing the maximum number of instances simply allows the Auto Scaling group to scale up to handle increased load, but if the single AZ fails, all instances will be unavailable.

Option B also focuses on capacity and has the same drawback as Option A. Increasing the minimum number of instances ensures a baseline capacity is always available, but doesn't protect against AZ failures.

Option C is the correct solution. By launching instances in a second Availability Zone within the same AWS Region, the application becomes resilient to the failure of a single AZ. If one AZ becomes unavailable, the Application Load Balancer will route traffic to healthy instances in the other AZ. This is the core principle of high availability in AWS. The instances must be within the same region to access the same VPC.

Option D would introduce unnecessary complexity and potential latency issues by spanning AWS Regions. While cross-region failover is a valid disaster recovery strategy, it's not required to achieve high availability within a Region, and adds complexity with data replication and DNS changes.

Therefore, launching instances in another AZ within the same region is the most efficient and effective way to achieve high availability.

Relevant links:

AWS Auto Scaling: https://docs.aws.amazon.com/autoscaling/ec2/userguide/auto-scaling-groups.html AWS Regions and Availability Zones: https://aws.amazon.com/about-aws/global-infrastructure/ Application Load Balancer:
https://docs.aws.amazon.com/elasticloadbalancing/latest/application/introduction.html

Question: 2

A company has deployed a web application in a VPC that has subnets in three Availability Zones. The company launches three Amazon EC2 instances from an
EC2 Auto Scaling group behind an Application Load Balancer (ALB).
A SysOps administrator notices that two of the EC2 instances are in the same Availability Zone, rather than being distributed evenly across all three Availability
Zones. There are no errors in the Auto Scaling group's activity history.
What is the MOST likely reason for the unexpected placement of EC2 instances?

A. One Availability Zone did not have sufficient capacity for the requested EC2 instance type.

B. The ALB was configured for only two Availability Zones.

C. The Auto Scaling group was configured for only two Availability Zones.

D. Amazon EC2 Auto Scaling randomly placed the instances in Availability Zones.

Answer: C

Explanation:

Here's a detailed justification for why option C is the most likely reason for the uneven EC2 instance distribution across Availability Zones (AZs) in the given scenario:

The core issue is that the Auto Scaling group is not launching instances across all three AZs as expected. Auto Scaling groups are designed to distribute instances evenly across the AZs they are configured for, provided resources are available.

Option A (Insufficient capacity): While capacity issues can occur in AWS AZs, it's less likely to be the root cause here. If capacity were the problem, the Auto Scaling group activity history would likely show errors indicating that instances couldn't be launched in a specific AZ due to capacity constraints. The prompt explicitly states there are no errors.

Option B (ALB configured for only two AZs): The ALB's configuration primarily impacts how traffic is routed to the instances. The ALB must be configured for the same AZs that the EC2 instances are in to route traffic to them. It wouldn't directly cause the Auto Scaling group to launch instances in only two AZs. The ALB being configured for only two AZs would cause the instances in the unavailable AZ to not receive traffic, it would not prevent the Auto Scaling group from creating them.

Option C (Auto Scaling group configured for only two AZs): This is the most plausible explanation. Auto Scaling groups have a configuration setting that specifies which AZs they are allowed to launch instances in. If the Auto Scaling group is configured to only use two of the three available AZs, it will only launch instances within those two. This would directly explain the observed behavior, even if the ALB is correctly configured to use all AZs.

Option D (Random placement): Auto Scaling does not randomly place instances. It attempts to distribute instances evenly across the configured AZs to achieve high availability.

Therefore, the most straightforward explanation for the instances being placed in only two AZs, without any errors reported, is that the Auto Scaling group itself is configured for only those two AZs.

Supporting Resources:

AWS Auto Scaling documentation: Provides an overview of Auto Scaling and how to configure it for availability and fault tolerance.

Application Load Balancer Documentation: Provides an overview of how to configure Application Load Balancers for availability and fault tolerance.

Question: 3

A company is running a website on Amazon EC2 instances behind an Application Load Balancer (ALB). The company configured an Amazon CloudFront distribution and set the ALB as the origin. The company created an Amazon Route 53 CNAME record to send all traffic through the CloudFront distribution. As an unintended side effect, mobile users are now being served the desktop version of the website.

Which action should a SysOps administrator take to resolve this issue?

A. Configure the CloudFront distribution behavior to forward the User-Agent header.

B. Configure the CloudFront distribution origin settings. Add a User-Agent header to the list of origin custom

headers.

C. Enable IPv6 on the ALB. Update the CloudFront distribution origin settings to use the dualstack endpoint.

D. Enable IPv6 on the CloudFront distribution. Update the Route 53 record to use the dualstack endpoint.

Answer: A

Explanation:

The issue arises because CloudFront, by default, does not forward the User-Agent header to the origin (ALB).

The User-Agent header is crucial for the origin server to determine the type of device (mobile or desktop) making the request and serve the appropriate version of the website. Without forwarding this header, the ALB treats all requests as originating from a generic source (CloudFront), defaulting to the desktop version.

Option A, configuring CloudFront to forward the User-Agent header, directly addresses this problem. By adding User-Agent to the "Cache Based on Selected Request Headers" (also known as "Whitelist Headers") in the CloudFront behavior settings, the header will be passed through to the ALB. The ALB can then correctly identify the device type based on the User-Agent and serve the mobile version to mobile users.

Option B is incorrect. Adding User-Agent to the origin custom headers would set a specific User-Agent header for all requests going to the origin, instead of forwarding the actual header from the client. This would not solve the problem and would likely make it worse.

Options C and D relate to IPv6, which is not the root cause of the issue. While enabling IPv6 can improve performance and future-proof the infrastructure, it doesn't directly address the problem of incorrect device detection. Updating to dualstack endpoints is only relevant if enabling IPv6, but the fundamental problem persists if the User-Agent is not forwarded. The issue is not IP addressing, but rather HTTP header handling.

Therefore, forwarding the User-Agent header allows the origin (ALB) to properly determine the client's device and deliver the appropriate content, resolving the described problem.

Supporting Documentation:

CloudFront documentation on caching based on request headers:
https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/controlling-the-cache-key.html#cache-key-headers
AWS Knowledge Center article on troubleshooting CloudFront serving desktop version to mobile users:
https://aws.amazon.com/premiumsupport/knowledge-center/cloudfront-serve-mobile-desktop-content/

Question: 4

A SysOps administrator has enabled AWS CloudTrail in an AWS account. If CloudTrail is disabled, it must be re-enabled immediately.
What should the SysOps administrator do to meet these requirements WITHOUT writing custom code?

A. Add the AWS account to AWS Organizations. Enable CloudTrail in the management account.

B. Create an AWS Config rule that is invoked when CloudTrail configuration changes. Apply the AWS-ConfigureCloudTrailLogging automatic remediation action.

C. Create an AWS Config rule that is invoked when CloudTrail configuration changes. Configure the rule to invoke an AWS Lambda function to enable CloudTrail.

D. Create an Amazon EventBridge (Amazon CloudWatch Event) hourly rule with a schedule pattern to run an AWS Systems Manager Automation document to enable CloudTrail.

Answer: B

Explanation:

The correct answer is B. Here's why:

Explanation:

The requirement is to automatically re-enable CloudTrail if it's disabled, without custom coding. AWS Config is a service designed for assessing, auditing, and evaluating the configurations of your AWS resources. It provides a way to track configuration changes over time and automatically remediate non-compliant states.

Option B leverages AWS Config's built-in capabilities effectively:

1. **AWS Config Rule:** Creating an AWS Config rule that triggers upon CloudTrail configuration changes allows you to detect when CloudTrail is disabled. The rule continuously monitors the CloudTrail configuration.
2. **AWS-ConfigureCloudTrailLogging Remediation Action:** AWS Config offers pre-built remediation actions. AWS-ConfigureCloudTrailLogging is a managed remediation action specifically designed to automatically enable CloudTrail logging. By associating this automatic remediation action with the Config rule, AWS Config will automatically re-enable CloudTrail when the rule detects that it has been disabled. This is all without writing code.

Why other options are less suitable:

A: AWS Organizations CloudTrail integration can ensure CloudTrail is enabled across multiple accounts, but it doesn't immediately re-enable CloudTrail if it's disabled in a specific account. It's more for centralized logging.

C: While feasible, using a Lambda function adds unnecessary complexity when AWS Config offers a built-in remediation action. Writing and managing custom Lambda code increases the maintenance overhead.

D: EventBridge and Systems Manager Automation can re-enable CloudTrail, but it is a schedule-based solution, it's not triggered immediately upon CloudTrail being disabled. This approach would introduce a delay. Further, it requires creating and managing an automation document. The eventbridge will fire at the configured cadence (Hourly) and remediate which is not immediate and less efficient.

In summary, Option B provides the most efficient, codeless, and immediate solution for automatically re-enabling CloudTrail upon disabling, leveraging the native capabilities of AWS Config.

Supporting Links:

AWS Config:https://aws.amazon.com/config/
AWS Config Rules:https://docs.aws.amazon.com/config/latest/developerguide/evaluate-config.html **AWS Config Remediation:**https://docs.aws.amazon.com/config/latest/developerguide/remediation.html **AWS-ConfigureCloudTrailLogging Remediation Action:**
https://docs.aws.amazon.com/config/latest/developerguide/remediation-actions.html#aws-configurecloudtraillogging

Question: 5

A company hosts its website on Amazon EC2 instances behind an Application Load Balancer. The company manages its DNS with Amazon Route 53, and wants to point its domain's zone apex to the website. Which type of record should be used to meet these requirements?

A. An AAAA record for the domain's zone apex
B. An A record for the domain's zone apex
C. A CNAME record for the domain's zone apex
D. An alias record for the domain's zone apex

Answer: D

Explanation:

Here's a detailed justification for why the correct answer is D (Alias record):

The company wants to point its domain's zone apex (e.g., example.com) to an Application Load Balancer (ALB). The key issue here is the zone apex limitation with CNAME records.

CNAME records cannot be used at the zone apex. CNAME records map an alias to another domain name. However, at the zone apex, other record types (like SOA and NS) must also exist. CNAMEs cannot coexist with these required records. Using a CNAME would violate DNS standards and likely cause resolution failures. Option C is therefore incorrect.

A and AAAA records directly map a domain name to IP addresses (IPv4 and IPv6, respectively). While seemingly a possibility, ALBs don't have static, predictable IP addresses. They dynamically change as the ALB scales. Hardcoding an A or AAAA record would quickly become outdated and result in intermittent or complete website outages. Option B is incorrect because of ALB dynamic IPs.

Alias records, specifically in Route 53, provide a Route 53-specific extension to DNS. They offer similar functionality to CNAMEs but are specifically designed to work with AWS resources like ALBs, CloudFront distributions, and S3 buckets configured for website hosting. Alias records essentially resolve the zone apex to the underlying AWS resource without exposing its volatile IP address. When Route 53 receives a DNS query for the zone apex, it dynamically retrieves the ALB's current IP addresses and returns them in the response.

Using an Alias record abstracts away the underlying IP address changes of the ALB. Route 53 automatically keeps the DNS records updated with the current IP addresses associated with the ALB. This ensures that traffic is consistently routed to the correct endpoint, even as the ALB scales and its IP addresses change. An alias record is the right approach, as it allows you to route traffic to an AWS resource like an ALB at the zone apex.

In summary, A and AAAA records are unsuitable because they require static IP addresses which are not provided by the Application Load Balancer. A CNAME record can’t be used for a zone apex. An alias record provides the required mapping functionality and automatically handles the dynamic IP addresses associated with AWS resources at the zone apex.

Supporting documentation:

AWS Route 53 Alias Records:https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/resource-record-sets-choosing-alias-non-alias.html
AWS Application Load Balancer:
https://docs.aws.amazon.com/elasticloadbalancing/latest/application/introduction.html

Question: 6

A company must ensure that any objects uploaded to an S3 bucket are encrypted. Which of the following actions will meet this requirement? (Choose two.)

A. Implement AWS Shield to protect against unencrypted objects stored in S3 buckets.

B. Implement Object access control list (ACL) to deny unencrypted objects from being uploaded to the S3 bucket.

C. Implement Amazon S3 default encryption to make sure that any object being uploaded is encrypted before it is stored.

D. Implement Amazon Inspector to inspect objects uploaded to the S3 bucket to make sure that they are encrypted.

E. Implement S3 bucket policies to deny unencrypted objects from being uploaded to the buckets.

Answer: CE

Explanation:

The correct answer is **C and E** because they directly address the requirement of ensuring all objects uploaded to an S3 bucket are encrypted.

Justification:

Option C: Implement Amazon S3 default encryption: S3 default encryption automatically encrypts all new objects placed in the bucket. It applies a server-side encryption method (SSE-S3, SSE-KMS, or SSE-C) to every object without requiring the uploader to specify encryption in their requests. Enabling default encryption offers a simple and comprehensive solution to guarantee encryption at rest for all future uploads. This helps meet compliance requirements and protect sensitive data stored within S3.
https://docs.aws.amazon.com/AmazonS3/latest/userguide/default-bucket-encryption.html

Option E: Implement S3 bucket policies to deny unencrypted objects: S3 bucket policies allow you to define fine-grained access control for your bucket and its objects. By creating a bucket policy that denies s3:PutObject requests that do not include the s3:x-amz-server-side-encryption header, you can enforce encryption at the time of upload. This ensures that only encrypted objects are stored in the bucket, preventing unencrypted objects from being added. This approach provides a proactive measure to enforce encryption.
https://docs.aws.amazon.com/AmazonS3/latest/userguide/example-bucket-policies.html (See example policies regarding encryption.)

Why other options are incorrect:

Option A: Implement AWS Shield: AWS Shield is a DDoS (Distributed Denial of Service) protection service. It does not deal with data encryption.

Option B: Implement Object access control list (ACL): ACLs are used to manage permissions on S3 objects but cannot be used to enforce encryption. While you can control who can access an object, ACLs cannot mandate how the object is stored (encrypted or unencrypted).

Option D: Implement Amazon Inspector: Amazon Inspector is a vulnerability assessment service. It's used to identify security vulnerabilities in your EC2 instances and network configurations but it is not designed to inspect the encryption status of objects stored in S3 buckets. Inspector is a post-upload analysis tool, and it does not prevent unencrypted uploads from occurring.

Question: 7

A company has a stateful web application that is hosted on Amazon EC2 instances in an Auto Scaling group. The instances run behind an Application Load
Balancer (ALB) that has a single target group. The ALB is configured as the origin in an Amazon CloudFront distribution. Users are reporting random logouts from the web application.
Which combination of actions should a SysOps administrator take to resolve this problem? (Choose two.)

A. Change to the least outstanding requests algorithm on the ALB target group.
B. Configure cookie forwarding in the CloudFront distribution cache behavior.
C. Configure header forwarding in the CloudFront distribution cache behavior.
D. Enable group-level stickiness on the ALB listener rule.
E. Enable sticky sessions on the ALB target group.

Answer: BE

Explanation:

The issue of random logouts in a stateful web application points to a problem with maintaining user sessions. Stateful applications require the server to remember user-specific information across multiple requests. When users are being routed to different backend servers unexpectedly, their session data is lost, leading to logouts.

Option B (Configure cookie forwarding in the CloudFront distribution cache behavior) is correct because it ensures that the session cookies set by the Application Load Balancer (ALB) or the backend servers are passed through CloudFront to the user's browser and back to the ALB. Without cookie forwarding, CloudFront might cache responses without the session cookie, or not forward the user's session cookie, leading to the user being considered a new, unauthenticated user on subsequent requests. This disrupts the session and causes logouts.
https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/Cookies.html

Option E (Enable sticky sessions on the ALB target group) is correct because it ensures that all requests from a specific user are routed to the same EC2 instance behind the ALB for the duration of the session. This is crucial for stateful applications because it allows the server to maintain the user's session data. Sticky sessions are typically implemented using cookies. The ALB inserts a cookie into the initial response to the user, and subsequent requests from that user contain the cookie, allowing the ALB to direct the requests to the same target.
https://docs.aws.amazon.com/elasticloadbalancing/latest/application/sticky-sessions.html

Option A is incorrect because the least outstanding requests algorithm is an ALB feature for load balancing, not session management. While it helps distribute load, it doesn't guarantee a user will stay on the same instance. Option C is incorrect because while forwarding certain headers might be important in some applications, it does not address the core issue of session management. Session data is typically maintained using cookies, not headers. Option D is incorrect because group-level stickiness is not a standard ALB feature. The intended functionality is more accurately described by sticky sessions on the target group.

Question: 8

A company hosts its website in the us-east-1 Region. The company is preparing to deploy its website into the eu-central-1 Region. Website visitors who are located in Europe should access the website that is hosted in eu-central-1. All other visitors access the website that is hosted in us-east-1. The company uses
Amazon Route 53 to manage the website's DNS records.
Which routing policy should a SysOps administrator apply to the Route 53 record set to meet these requirements?

A. Geolocation routing policy
B. Geoproximity routing policy
C. Latency routing policy
D. Multivalue answer routing policy

Answer: A

Explanation:

The correct answer is **A. Geolocation routing policy**.

Geolocation routing in Amazon Route 53 allows you to route traffic to different resources based on the geographic location of your users. This perfectly fits the requirement of sending European users to the eu-central-1 Region and all other users to the us-east-1 Region. You would create two records for the same domain name, one configured for the "Europe" location pointing to the eu-central-1 website resources, and another configured for the "Default" location (which handles all regions not specifically defined) pointing to the us-east-1 website resources.

Geoproximity routing (B) routes traffic based on the physical distance between users and your resources, taking into account bias values you define, which is not the direct requirement here. Latency routing (C) routes

traffic to the region with the lowest latency for each user, but it doesn't explicitly guarantee that European users go to eu-central-1. Multivalue answer routing (D) returns multiple IP addresses in response to DNS queries, which can improve availability but doesn't provide geographic routing control. It's primarily for distributing traffic amongst healthy resources within a single region. The explicit geographic control offered by geolocation routing is the key differentiator that makes it the optimal choice. The problem specifically asks for a routing policy to route traffic based on location, and only Geolocation explicitly does this.

Authoritative Links:

AWS Route 53 Geolocation Routing: https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/routing-policy.html#routing-policy-geo
AWS Route 53 Routing Policies: https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/routing-policy.html

Question: 9

A company is running a website on Amazon EC2 instances that are in an Auto Scaling group. When the website traffic increases, additional instances take several minutes to become available because of a long-running user data script that installs software. A SysOps administrator must decrease the time that is required for new instances to become available. Which action should the SysOps administrator take to meet this requirement?

A. Reduce the scaling thresholds so that instances are added before traffic increases.
B. Purchase Reserved Instances to cover 100% of the maximum capacity of the Auto Scaling group.
C. Update the Auto Scaling group to launch instances that have a storage optimized instance type.
D. Use EC2 Image Builder to prepare an Amazon Machine Image (AMI) that has pre-installed software.

Answer: D

Explanation:

The correct answer is **D. Use EC2 Image Builder to prepare an Amazon Machine Image (AMI) that has pre-installed software.**

Here's why:

The problem is the long-running user data script delaying the readiness of newly launched EC2 instances in the Auto Scaling group during traffic spikes. This script is installing software, which is a time-consuming process.

Option D directly addresses this bottleneck by using EC2 Image Builder. EC2 Image Builder allows you to automate the creation, management, and deployment of custom, secure, and up-to-date "golden" server images (AMIs). By pre-installing the necessary software into the AMI, you eliminate the need to run the lengthy installation process every time a new instance is launched. When the Auto Scaling group launches new instances, they are created from this pre-baked AMI, significantly reducing the instance startup time and making them available much faster. This approach drastically reduces the time to service web traffic during scaling events.

Let's look at why the other options are not optimal:

A. Reduce the scaling thresholds so that instances are added before traffic increases: While proactive scaling can help anticipate load, it doesn't address the core problem of slow instance startup. It simply tries to get ahead of the problem instead of fixing it. The instances will still take a long time to become ready, and resources may be wasted by unnecessarily launching instances when the increase might not be sustained.
B. Purchase Reserved Instances to cover 100% of the maximum capacity of the Auto Scaling group:

Reserved Instances provide cost savings for long-term usage but do nothing to improve instance launch time.
They only guarantee capacity reservation and provide billing discounts; they do not pre-install software. **C. Update the Auto Scaling group to launch instances that have a storage optimized instance type:** Storage optimized instances are designed for I/O intensive applications. They do not directly reduce the time required to install software. While faster storage might improve the speed of software installation to some extent, it wouldn't address the core problem of the time taken for user data scripts to execute. This is more of an indirect improvement and doesn't solve the root cause.

In summary, using EC2 Image Builder to pre-bake an AMI with the software is the most effective and direct approach to decreasing the time required for new instances to become available in the Auto Scaling group.

Supporting Documentation:

EC2 Image Builder:https://aws.amazon.com/image-builder/
Auto Scaling Groups:https://docs.aws.amazon.com/autoscaling/ec2/userguide/auto-scaling-groups.html

Question: 10

A SysOps administrator is troubleshooting an AWS CloudFormation template whereby multiple Amazon EC2 instances are being created. The template is working in us-east-1, but it is failing in us-west-2 with the error code: AMI [ami-12345678] does not exist
How should the Administrator ensure that the AWS CloudFormation template is working in every region?

A. Copy the source region's Amazon Machine Image (AMI) to the destination region and assign it the same ID.

B. Edit the AWS CloudFormation template to specify the region code as part of the fully qualified AMI ID.

C. Edit the AWS CloudFormation template to offer a drop-down list of all AMIs to the user by using the AWS::EC2::AMI::ImageID control.

D. Modify the AWS CloudFormation template by including the AMI IDs in the Mappings section. Refer to the proper mapping within the template for the proper AMI ID.

Answer: D

Explanation:

The error "AMI [ami-12345678] does not exist" clearly indicates that the specified Amazon Machine Image (AMI) ID used in the CloudFormation template is not valid in the region where the deployment is failing (us-west-2). AMIs are region-specific resources; an AMI available in us-east-1 will likely have a different ID or not exist at all in us-west-2.

Option A is incorrect because AMI IDs are assigned by AWS when the AMI is created or copied. You cannot assign a specific ID to a copied AMI.

Option B is not the ideal solution. While specifying the region as part of the AMI ID might help in some cases (e.g., ami-12345678/us-east-1), it doesn't address the fundamental problem that the same AMI ID is unlikely to be valid in different regions. It also isn't a standardized format.

Option C proposes a user-interactive approach, which is generally not desirable for automated infrastructure deployments like those managed by CloudFormation. Relying on user input can introduce inconsistencies and manual errors.

Option D, using the Mappings section in the CloudFormation template, is the most robust and recommended approach. The Mappings section allows you to define a lookup table that maps regions to their corresponding AMI IDs. This enables the template to dynamically select the correct AMI ID based on the region in which the stack is being created. This approach ensures that the correct AMI is used in each region, resolving the initial error. By defining a mapping for each region and its corresponding AMI, the template becomes region-

agnostic and can be deployed successfully in multiple AWS regions without modification. The template will automatically choose the correct AMI based on the region it is being deployed in.

For more information on using mappings in CloudFormation templates, refer to the AWS documentation:

AWS CloudFormation Mappings:
https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/mappings-section-structure.html

Question: 11

A SysOps administrator is provisioning an Amazon Elastic File System (Amazon EFS) file system to provide shared storage across multiple Amazon EC2 instances. The instances all exist in the same VPC across multiple Availability Zones. There are two instances in each Availability Zone. The SysOps administrator must make the file system accessible to each instance with the lowest possible latency.
Which solution will meet these requirements?

A. Create a mount target for the EFS file system in the VPC. Use the mount target to mount the file system on each of the instances.

B. Create a mount target for the EFS file system in one Availability Zone of the VPC. Use the mount target to mount the file system on the instances in that Availability Zone. Share the directory with the other instances.

C. Create a mount target for each instance. Use each mount target to mount the EFS file system on each respective instance.

D. Create a mount target in each Availability Zone of the VPC. Use the mount target to mount the EFS file system on the instances in the respective Availability Zone.

Answer: D

Explanation:

The correct answer is D because it provides the lowest possible latency access to the EFS file system for EC2 instances across multiple Availability Zones within the same VPC.

Here's why:

Latency and Mount Targets: EFS is designed to be mounted directly on EC2 instances. Mount targets are the access points to the EFS file system within each Availability Zone.

Availability Zone Consideration: Placing mount targets in each Availability Zone where EC2 instances reside minimizes the network distance between the instances and the file system. This significantly reduces latency compared to routing traffic through a single Availability Zone (as in option B).

EFS Design: EFS is built for highly available and durable shared storage. To leverage this effectively, you need mount points near the instances that are using the EFS.

Option A Inefficiency: Option A creates only one mount target, forcing instances in other Availability Zones to communicate across Availability Zones. This adds latency and can impact performance.

Option C Limitation: Option C is impractical and not how EFS is designed to be used. EFS scales automatically, and managing a mount target per instance is not necessary.

Therefore, creating a mount target in each Availability Zone and using it to mount the file system on instances in that zone is the optimal way to achieve the lowest possible latency while leveraging the full benefits of EFS's shared storage capabilities.

Relevant Documentation:

Amazon EFS Mount Targets
Amazon EFS Performance

Question: 12

A SysOps administrator has successfully deployed a VPC with an AWS CloudFormation template. The SysOps administrator wants to deploy the same template across multiple accounts that are managed through AWS Organizations.
Which solution will meet this requirement with the LEAST operational overhead?

A. Assume the OrganizationAccountAccessRole IAM role from the management account. Deploy the template in each of the accounts.

B. Create an AWS Lambda function to assume a role in each account. Deploy the template by using the AWS CloudFormation CreateStack API call.

C. Create an AWS Lambda function to query for a list of accounts. Deploy the template by using the AWS CloudFormation CreateStack API call.

D. Use AWS CloudFormation StackSets from the management account to deploy the template in each of the accounts.

Answer: D

Explanation:

The correct answer is **D. Use AWS CloudFormation StackSets from the management account to deploy the template in each of the accounts.**

Here's a detailed justification:

AWS CloudFormation StackSets are specifically designed to deploy and manage stacks across multiple AWS accounts and Regions from a single point of control, typically the management account of an AWS Organization. StackSets provide a centralized and streamlined approach for deploying and managing infrastructure across multiple accounts, significantly reducing operational overhead compared to other methods.

Option A (assuming the OrganizationAccountAccessRole and deploying in each account manually) is cumbersome and error-prone. It requires manual intervention in each account, increasing the risk of inconsistencies and configuration drift. Furthermore, it doesn't scale well as the number of accounts grows.

Option B and C (using a Lambda function to assume roles and deploy via the CloudFormation API) introduce unnecessary complexity. While technically feasible, this approach requires writing and maintaining custom code, increasing the operational burden. It involves more moving parts and potential points of failure compared to StackSets. A Lambda function may also have issues with rate limiting or execution timeouts when dealing with a large number of accounts. You would need to handle error scenarios and retries explicitly within the Lambda function.

StackSets, on the other hand, provide a managed service that handles the complexities of deploying stacks across multiple accounts. They offer features such as automatic retries, dependency management, and rollback capabilities. By using StackSets from the management account, you leverage a native AWS service that is optimized for multi-account deployments. This solution minimizes the operational overhead associated with managing infrastructure across multiple accounts. StackSets directly integrate with AWS Organizations, simplifying the process of targeting deployments to specific organizational units or accounts.

Therefore, using AWS CloudFormation StackSets is the most efficient and scalable solution for deploying the CloudFormation template across multiple accounts within AWS Organizations, requiring the least operational overhead.

Relevant links for further research:

AWS CloudFormation StackSets:
https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/stacksets-concepts.html **AWS Organizations:**https://aws.amazon.com/organizations/

Question: 13

A company is running distributed computing software to manage a fleet of 20 Amazon EC2 instances for calculations. The fleet includes 2 control nodes and 18 task nodes to run the calculations. Control nodes can automatically start the task nodes.
Currently, all the nodes run on demand. The control nodes must be available 24 hours a day, 7 days a week. The task nodes run for 4 hours each day. A SysOps administrator needs to optimize the cost of this solution.
Which combination of actions will meet these requirements? (Choose two.)

A. Purchase EC2 Instance Savings Plans for the control nodes.
B. Use Dedicated Hosts for the control nodes.
C. Use Reserved Instances for the task nodes.
D. Use Spot Instances for the control nodes. Use On-Demand Instances if there is no Spot availability.
E. Use Spot Instances for the task nodes. Use On-Demand Instances if there is no Spot availability.

Answer: AE

Explanation:

The requirement is to minimize the cost of running a distributed computing system consisting of control and task nodes on EC2. The control nodes must be running 24/7, and task nodes run for 4 hours daily.

Option A, purchasing EC2 Instance Savings Plans for the control nodes, is a cost-effective solution. Savings Plans provide significant discounts (up to 72%) compared to On-Demand pricing in exchange for a commitment to a consistent amount of compute usage (measured in $/hour) for a 1- or 3-year term. Because control nodes require continuous availability, this commitment aligns perfectly with their usage pattern, guaranteeing cost savings. Refer to https://aws.amazon.com/savingsplans/ for more details.

Option E, using Spot Instances for the task nodes, backed by On-Demand Instances if Spot capacity isn't available, is appropriate for task nodes. Spot Instances offer substantial discounts (up to 90%) compared to On-Demand pricing. Because task nodes only run for 4 hours daily, the risk of interruption associated with Spot Instances is acceptable. If a Spot Instance is interrupted, the system can fall back to On-Demand instances, ensuring task completion at a higher cost only when needed. This provides a good balance between cost savings and reliability for the task nodes. Learn more at https://aws.amazon.com/ec2/spot/.

Option B is incorrect. Dedicated Hosts are best for bring-your-own-license (BYOL) scenarios or specific compliance requirements, not general cost optimization.

Option C is incorrect. Reserved Instances are beneficial for instances running a predictable, near-continuous workload. While task nodes have a predictable runtime, Savings plans offer similar benefits to reserved instances with greater flexibility regarding instance size and family.

Option D is incorrect. Using Spot Instances for critical control nodes is risky due to potential interruptions. As control nodes must be available 24/7, relying solely on Spot Instances, even with an On-Demand fallback, poses an unacceptable risk to system availability.

Question: 14

A company is supposed to receive a data file every hour in an Amazon S3 bucket. An S3 event notification invokes an AWS Lambda function each time a file arrives. The function processes the data for use by an application. The application team notices that sometimes the file does not arrive. The application team wants to receive a notification whenever the file does not arrive.
What is the MOST operationally efficient solution that meets these requirements?

A. Add an S3 Lifecycle rule on the S3 bucket with a scope that is limited to objects that were created in the last hour. Configure another S3 event notification to be invoked by the lifecycle transition when the number of objects transitioned is zero. Publish a message to an Amazon Simple Notification Service (Amazon SNS) topic to notify the application team.

B. Configure another S3 event notification to invoke a Lambda function that posts a message to an Amazon Simple Queue Service (Amazon SQS) queue. Create an Amazon CloudWatch alarm to publish a message to an Amazon Simple Notification Service (Amazon SNS) topic to notify the application team when the ApproximateAgeOfOldestMessage metric of the queue is greater than 1 hour.

C. Create an Amazon CloudWatch alarm to publish a message to an Amazon Simple Notification Service (Amazon SNS) topic to alert the application team when the Invocations metric of the Lambda function is zero for an hour. Configure the alarm to treat missing data as breaching.

D. Create a new Lambda function to get the timestamp of the newest file in the S3 bucket. If the timestamp is more than 1 hour ago, publish a message to an Amazon Simple Notification Service (Amazon SNS) topic to notify the application team. Create an Amazon EventBridge (Amazon CloudWatch Events) rule to invoke the new function hourly.

Answer: C

Explanation:

The most operationally efficient solution is **C**. Here's why:

Direct Metric Monitoring: Option C leverages the Invocations metric of the existing Lambda function. This directly indicates if the function was triggered by an S3 event within the expected timeframe (1 hour).

CloudWatch Alarm Simplification: CloudWatch Alarms are designed to monitor metrics and take actions based on thresholds. Using the Invocations metric and setting a threshold of zero for one hour allows for the triggering of an SNS notification directly when no files are processed.

Missing Data Treatment: The configuration of the CloudWatch alarm to treat "missing data as breaching" is crucial. If, for some reason, the Invocations metric is not reported for an hour (due to service disruptions or other unforeseen issues), the alarm will still trigger, ensuring the application team is notified.

Reduced Complexity and Cost: Option A involves S3 Lifecycle rules, which are primarily designed for cost optimization and data archiving, not for real-time anomaly detection. It also adds complexity with another S3 event notification and lifecycle transitions, potentially increasing costs.

Option B introduces an SQS queue and relies on the ApproximateAgeOfOldestMessage metric. While this can detect missing messages, it adds complexity in the architecture with the need to ensure proper queue processing. Further, if a message enters the queue, the alarm won't fire until after it's older than 1 hour.

Option D creates a new Lambda function and an EventBridge rule, adding unnecessary overhead. It requires the new function to scan the S3 bucket which can become expensive and slower as the bucket grows. It does not reuse existing resources, making it less efficient.

Operational Efficiency: The elegance of Option C is its simplicity and directness. It reuses existing components (the original Lambda function) and utilizes CloudWatch's monitoring capabilities effectively, minimizing administrative overhead and potential failure points.

Therefore, using the existing function's invocation count, a CloudWatch alarm, and SNS notification provides a direct, cost-effective, and operationally efficient approach to addressing the application team's requirements.

Relevant links:

AWS Lambda Metrics: https://docs.aws.amazon.com/lambda/latest/dg/monitoring-metrics.html Amazon CloudWatch Alarms: https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/AlarmThatSendsEmail.html

Question: 15

A company recently acquired another corporation and all of that corporation's AWS accounts. A financial analyst needs the cost data from these accounts. A
SysOps administrator uses Cost Explorer to generate cost and usage reports. The SysOps administrator notices that "No Tagkey" represents 20% of the monthly cost.
What should the SysOps administrator do to tag the "No Tagkey" resources?

A. Add the accounts to AWS Organizations. Use a service control policy (SCP) to tag all the untagged resources.

B. Use an AWS Config rule to find the untagged resources. Set the remediation action to terminate the resources.

C. Use Cost Explorer to find and tag all the untagged resources.

D. Use Tag Editor to find and tag all the untagged resources.

Answer: D

Explanation:

Here's a detailed justification for why option D is the correct answer, and why the other options are incorrect:

Justification for Option D (Use Tag Editor to find and tag all the untagged resources):

The core problem is identifying and tagging resources that currently lack tags ("No Tagkey" in Cost Explorer). Tag Editor is the most direct and appropriate AWS service designed for this purpose. It allows you to search for resources across regions and services based on tag criteria (or lack thereof). After identifying untagged resources, Tag Editor enables you to apply new tags in bulk, simplifying the process considerably. This helps in improving cost allocation and visibility. By properly tagging the "No Tagkey" resources, the financial analyst gains a clearer picture of where costs are originating, enabling better budget management and resource optimization. Tag Editor promotes consistent tagging policies, which is crucial for maintaining cost
transparency as the organization integrates the acquired company's infrastructure. This tool centralizes tag management, reducing the risk of inconsistent or missing tags in the future. This aligns directly with the goal of understanding and managing costs effectively.

Why other options are incorrect:

A (Add the accounts to AWS Organizations. Use a service control policy (SCP) to tag all the untagged resources): While AWS Organizations is beneficial for central management of multiple AWS accounts, using SCPs for tagging is not its primary purpose and is overly complex for this specific scenario. SCPs primarily enforce policies to control what actions IAM users and roles can perform within member accounts. While SCPs can prevent actions on untagged resources, they don't apply tags retroactively. This solution does not address the immediate problem.

B (Use an AWS Config rule to find the untagged resources. Set the remediation action to terminate the resources): This is an extremely drastic and inappropriate solution. Terminating untagged resources would likely cause significant disruption and data loss, making it unacceptable from a business perspective. AWS Config is useful for compliance and governance, but termination is an extreme remediation for a tagging issue. Furthermore, the intention is to identify and tag the resources, not eliminate them.

C (Use Cost Explorer to find and tag all the untagged resources): Cost Explorer is primarily designed for visualizing and analyzing cost and usage data. While Cost Explorer highlights the "No Tagkey" cost component, it doesn't provide the functionality to directly tag the underlying resources. Cost Explorer's purpose is to identify trends and anomalies to take appropriate actions with other tools. The core issue is to find and tag the untagged resources, a function that Cost Explorer doesn't provide.

Authoritative Links:

AWS Tag Editor:https://docs.aws.amazon.com/awsconsole/latest/userguide/resource-groups-tag-editor.html **AWS Organizations:**https://aws.amazon.com/organizations/

AWS Config:https://aws.amazon.com/config/
AWS Cost Explorer:https://aws.amazon.com/aws-cost-management/aws-cost-explorer/

Question: 16

While setting up an AWS managed VPN connection, a SysOps administrator creates a customer gateway resource in AWS. The customer gateway device resides in a data center with a NAT gateway in front of it.
What address should be used to create the customer gateway resource?

A. The private IP address of the customer gateway device
B. The MAC address of the NAT device in front of the customer gateway device
C. The public IP address of the customer gateway device
D. The public IP address of the NAT device in front of the customer gateway device

Answer: D

Explanation:

The correct answer is **D. The public IP address of the NAT device in front of the customer gateway device.**

Here's why:

When establishing an AWS managed VPN connection with a customer gateway behind a NAT (Network Address Translation) device, AWS needs a publicly routable IP address to establish the VPN tunnel. The customer gateway device itself likely has a private IP address assigned to it within the data center's network. However, the internet-facing VPN gateway in AWS cannot directly reach this private IP address because it's hidden behind the NAT.

The NAT device translates the private IP address of the customer gateway device to a public IP address. This public IP address is what the outside world, including AWS, sees as the source of traffic originating from the customer network. AWS uses this public IP address to initiate and maintain the VPN tunnel. Therefore, the customer gateway resource in AWS must be configured with the public IP address of the NAT device.

Options A and B are incorrect because AWS VPN connections require a publicly routable IP address. The private IP address of the customer gateway device is not accessible from the public internet, and the MAC address of the NAT device is a Layer 2 identifier and not used for internet routing. Option C is incorrect because if the customer gateway device had its own public IP, there wouldn't be a need for a NAT device. In this scenario, the NAT device is acting as an intermediary, and its public IP is the only one visible to the AWS VPN gateway.

Authoritative Links:

AWS Documentation - Customer Gateway:https://docs.aws.amazon.com/vpn/latest/s2svpn/cgw-options.html
AWS Documentation - Site-to-Site VPN:https://docs.aws.amazon.com/vpn/latest/s2svpn/VPC_VPN.html

These links provide comprehensive information regarding AWS VPN connections, customer gateways, and the configurations required for successful VPN setup, particularly when dealing with NAT devices.

Question: 17

A company has a web application that is experiencing performance problems many times each night. A root cause analysis reveals sudden increases in CPU utilization that last 5 minutes on an Amazon EC2 Linux instance. A SysOps administrator must find the process ID (PID) of the service or process that is consuming more CPU.

What should the SysOps administrator do to collect the process utilization information with the LEAST amount of effort?

A. Configure the Amazon CloudWatch agent procstat plugin to capture CPU process metrics.
B. Configure an AWS Lambda function to run every minute to capture the PID and send a notification.
C. Log in to the EC2 instance by using a .pem key each night. Then run the top command.
D. Use the default Amazon CloudWatch CPU utilization metric to capture the PID in CloudWatch.

Answer: A

Explanation:

The correct answer is A: Configure the Amazon CloudWatch agent procstat plugin to capture CPU process metrics. Here's why:

Justification:

The primary goal is to identify the process consuming high CPU during the performance spikes on the EC2 instance with minimal effort. Option A directly addresses this requirement using the Amazon CloudWatch agent and its procstat plugin.

1. **Granular Data Collection:** The procstat plugin is specifically designed to monitor process-level metrics, including CPU utilization, memory usage, and other relevant data. This provides the necessary granularity to pinpoint the problematic process.
   https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/procstat.html

2. **Automated Monitoring:** Once configured, the CloudWatch agent automatically collects and sends the process metrics to CloudWatch. This eliminates the need for manual intervention or scripting.

3. **Centralized Monitoring:** CloudWatch provides a centralized platform for monitoring all AWS resources, including EC2 instances. This allows the administrator to easily visualize the CPU utilization of different processes over time and correlate them with the observed performance issues.
   https://aws.amazon.com/cloudwatch/

4. **Historical Analysis:** CloudWatch retains historical data, enabling the administrator to analyze past performance spikes and identify recurring patterns.

5. **Least Effort:** Compared to other options, configuring the CloudWatch agent with the procstat plugin requires the least amount of manual effort. It avoids the need for custom scripting (Option B), manual login (Option C), or reliance on default metrics that don't provide process-level granularity (Option D).

Why other options are less optimal:

Option B: While a Lambda function could capture the PID, it involves writing, deploying, and managing custom code, which is more complex than using the procstat plugin. Also, managing function execution at very short intervals is costly and adds operational overhead.

Option C: Manually logging in and running the top command each night is time-consuming, error-prone, and not scalable. It's also a reactive approach rather than proactive monitoring.

Option D: The default CloudWatch CPU utilization metric provides an aggregate view of CPU usage for the entire instance, not at the individual process level. It doesn't directly identify the problematic PID.

Question: 18

A SysOps administrator configured AWS Backup to capture snapshots from a single Amazon EC2 instance that has one Amazon Elastic Block Store (Amazon
EBS) volume attached. On the first snapshot, the EBS volume has 10 GiB of data. On the second snapshot, the EBS

volume still contains 10 GiB of data, but 4
GiB have changed. On the third snapshot, 2 GiB of data have been added to the volume, for a total of 12 GiB. How much total storage is required to store these snapshots?

A. 12 GiB
B. 16 GiB
C. 26 GiB
D. 32 GiB

Answer: B

Explanation:

Here's a breakdown of why the correct answer is B (16 GiB) and a detailed explanation:

Understanding EBS Snapshots and Incremental Backups

EBS snapshots are designed to be incremental. This means that the first snapshot of a volume captures all the data on the volume at that time. Subsequent snapshots only store the changes that have occurred since the last snapshot. This approach significantly reduces storage costs and snapshot creation time.

Analyzing the Scenario

Snapshot 1: The EBS volume contains 10 GiB of data. The first snapshot stores this full 10 GiB.

Snapshot 2: 4 GiB of data have changed since the first snapshot. Only these 4 GiB of changes are stored. **Snapshot 3:** 2 GiB of data have been added. Since EBS snapshots are incremental, this 2GB is also stored as a new change.

Calculating Total Storage

To calculate the total storage used, we sum the storage required for each snapshot:

Snapshot 1: 10 GiB
Snapshot 2: 4 GiB
Snapshot 3: 2 GiB
Total: 10 + 4 + 2 = 16 GiB

Why other options are incorrect:

A. 12 GiB: This is incorrect because it only accounts for the final size of the volume and doesn't include the initial snapshot data, and change in the second snapshot.

C. 26 GiB: This is incorrect, likely calculated as the sum of data on all the snapshots (10+10+6), ignoring the incremental nature.

D. 32 GiB: This is incorrect because it doesn't account for incremental snapshots. A full snapshot would only be taken for the first backup.

Authoritative Links for further research:

AWS EBS Snapshots:https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSSnapshots.html **AWS Backup Documentation:**https://docs.aws.amazon.com/aws-backup/latest/devguide/whatisbackup.html

Question: 19 A team is managing an AWS account that is a member of an organization in AWS Organizations. The organization

has consolidated billing features enabled. The account hosts several applications.
A SysOps administrator has applied tags to resources within the account to reflect the environment. The team needs a report of the breakdown of charges by environment.
What should the SysOps administrator do to meet this requirement?

A. Filter, map, and categorize resource groups in Tag Editor.

B. Ensure that the organization's service control policies (SCPs) allow access to cost allocation tags.

C. Ensure that the IAM credentials that are used to access Cost Explorer have permissions to group cost by tags.

D. Activate the tag keys for cost allocation on the organization's management account.

Answer: D

Explanation:

The correct answer is D: Activate the tag keys for cost allocation on the organization's management account. Here's why:

Cost Allocation Tags: AWS Cost Allocation Tags are used to track your AWS costs on a detailed level. They allow you to categorize your AWS resources based on tags and then generate cost reports based on those tags. This directly addresses the requirement of breaking down charges by environment, as reflected in the resource tags.

Organizational Context: Because the AWS account is part of an AWS Organization with consolidated billing, cost allocation tag activation must be performed at the organization's management account level. This is because consolidated billing rolls up all the costs from member accounts to the management account. Tag activation at the member account level will not be sufficient for organization-wide cost reporting.

Activation is Necessary: Simply tagging resources is not enough. AWS needs to be told to track costs based on these tags. Activating the tag keys tells AWS to include those tags in cost allocation reports.

Why other options are incorrect:

A. Filter, map, and categorize resource groups in Tag Editor: Tag Editor helps you manage tags, but it doesn't directly contribute to cost allocation. Tag Editor allows you to apply and modify tags across AWS resources. It's not related to generating cost reports.

B. Ensure that the organization's service control policies (SCPs) allow access to cost allocation tags: SCPs control which services and actions member accounts can access. They're used for permission management, not cost allocation configuration. While SCPs can restrict access related to billing, they do not enable cost allocation tags.

C. Ensure that the IAM credentials that are used to access Cost Explorer have permissions to group cost by tags: IAM permissions for Cost Explorer control who can view cost reports and group costs, but they do not enable the cost allocation tagging itself. IAM permissions only govern access and do not activate cost tracking functionality.

In summary, activating cost allocation tags at the organization's management account ensures that AWS begins tracking costs based on the applied environment tags, fulfilling the reporting requirement.

Relevant Documentation:

AWS Cost Allocation Tags
Activating Cost Allocation Tags

Question: 20

A company uses an AWS CloudFormation template to provision an Amazon EC2 instance and an Amazon RDS DB

instance. A SysOps administrator must update the template to ensure that the DB instance is created before the EC2 instance is launched.
What should the SysOps administrator do to meet this requirement?

A. Add a wait condition to the template. Update the EC2 instance user data script to send a signal after the EC2 instance is started.

B. Add the DependsOn attribute to the EC2 instance resource, and provide the logical name of the RDS resource.

C. Change the order of the resources in the template so that the RDS resource is listed before the EC2 instance resource.

D. Create multiple templates. Use AWS CloudFormation StackSets to wait for one stack to complete before the second stack is created.

Answer: B

Explanation:

The correct answer is **B. Add the DependsOn attribute to the EC2 instance resource, and provide the logical name of the RDS resource.**

Here's a detailed justification:

AWS CloudFormation processes resources in parallel by default to minimize deployment time. This means that the EC2 instance and RDS DB instance might be created simultaneously. However, sometimes dependencies exist between resources. In this case, the requirement is that the RDS DB instance must be fully provisioned before the EC2 instance is launched, possibly because the EC2 instance needs to connect to the database during its initialization.

The DependsOn attribute in CloudFormation is designed precisely for this scenario. It explicitly defines a dependency between resources. By adding DependsOn to the EC2 instance resource and setting its value to the logical ID of the RDS DB instance resource, we are instructing CloudFormation to create the RDS instance first and only begin the EC2 instance creation after the RDS instance has been successfully created. The logical ID is the unique name you give the RDS resource within the CloudFormation template.

Option A is incorrect because wait conditions are typically used when you need to wait for a custom action within a resource to complete before proceeding with other actions. This doesn't inherently establish a creation order between completely separate resources like an RDS and EC2 instance in the CloudFormation configuration itself.

Option C is incorrect because the order of resources listed in a CloudFormation template does not guarantee creation order. CloudFormation's default behavior is to attempt parallel creation of resources to optimize deployment time.

Option D is incorrect because using StackSets to create separate stacks introduces unnecessary complexity.
StackSets are designed for deploying stacks across multiple AWS accounts or regions, not for defining dependencies within a single stack deployment. Furthermore, using multiple stacks when a single stack can handle the requirement adds management overhead. The DependsOn attribute provides a simpler, more direct solution within a single CloudFormation template.

In summary, the DependsOn attribute is the most straightforward and efficient way to enforce a specific creation order between resources within a CloudFormation template, ensuring that the RDS DB instance is created before the EC2 instance. This achieves the desired behavior of the updated CloudFormation template.

Refer to the AWS documentation for more information:

DependsOn Attribute:https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/aws-attribute-dependson.html

Question: 21

A company hosts a static website on Amazon S3. The website is served by an Amazon CloudFront distribution with a default TTL of 86,400 seconds.
The company recently uploaded an updated version of the website to Amazon S3. However, users still see the old content when they refresh the site. A SysOps administrator must make the new version of the website visible to users as soon as possible.
Which solution meets these requirements?

A. Adjust the TTL value for the DNS CNAME record that is pointing to the CloudFront distribution.
B. Create an invalidation on the CloudFront distribution for the old S3 objects.
C. Create a new CloudFront distribution. Update the DNS records to point to the new CloudFront distribution.
D. Update the DNS record for the website to point to the S3 bucket.

Answer: B

Explanation:

The correct answer is B: Create an invalidation on the CloudFront distribution for the old S3 objects. Here's why:

CloudFront caches content for a specified Time To Live (TTL). When content is updated in S3, CloudFront continues to serve the cached, outdated version until the TTL expires. The problem states the TTL is 86,400 seconds (24 hours), which is too long to wait for users to see the new content immediately.

Option A is incorrect because DNS TTL controls how long DNS resolvers cache the mapping between a domain name and its corresponding IP address. Adjusting the DNS TTL will only affect how quickly DNS propagates changes to the CloudFront distribution's hostname, not how quickly CloudFront refreshes the cached content.

Option C, creating a new CloudFront distribution, is not efficient. It introduces unnecessary complexity and cost. While it would eventually serve the new content, it's a slower and more resource-intensive solution compared to invalidating the cache. It also requires DNS changes, adding to the delay.

Option D, updating the DNS record to point directly to the S3 bucket, bypasses CloudFront altogether. This negates the benefits of using a CDN, such as improved performance, lower latency, and DDoS protection. It's also not suitable for serving static websites directly from S3 in most production scenarios.

Creating an invalidation is the fastest way to force CloudFront to retrieve the updated objects from the origin (S3) before the TTL expires. When you invalidate a file path, the next request for that path will result in CloudFront retrieving the latest version from S3 and serving it to the user. This immediate refresh addresses the problem requirement of making the new version of the website visible to users as soon as possible. You can invalidate specific objects or use a wildcard to invalidate all objects.

Refer to these AWS documentation links for more details:

Invalidating Files
How CloudFront Delivers Content

Question: 22

A SysOps administrator is responsible for managing a company's cloud infrastructure with AWS CloudFormation. The SysOps administrator needs to create a single resource that consists of multiple AWS services. The resource must support creation and deletion through the CloudFormation console.
Which CloudFormation resource type should the SysOps administrator create to meet these requirements?

A. AWS::EC2::Instance with a cfn-init helper script
B. AWS::OpsWorks::Instance
C. AWS::SSM::Document
D. Custom::MyCustomType

Answer: D

Explanation:

The correct answer is **D. Custom::MyCustomType**. Here's why:

The requirement is to create a single CloudFormation resource that encompasses multiple AWS services and supports create and delete operations via the CloudFormation console. A standard CloudFormation resource like AWS::EC2::Instance only manages EC2 instances directly and cannot handle the complex interactions with multiple services as a single unit.

A. AWS::EC2::Instance with a cfn-init helper script: While cfn-init can help configure an EC2 instance after creation, it doesn't create a single resource managing multiple AWS services as a unit. It primarily configures a single EC2 instance.

B. AWS::OpsWorks::Instance: OpsWorks is a configuration management service. While it automates server configuration, it's not designed to manage multiple disparate AWS services as a single CloudFormation resource. It is more focused on application management and deployment within a managed stack.

C. AWS::SSM::Document: SSM Documents are used for defining configuration or automation tasks that can be run on managed instances. They don't create or manage multiple AWS services as a single CloudFormation resource.

D. Custom::MyCustomType: CloudFormation Custom Resources are the ideal choice for this scenario. They allow you to define a custom resource type, backed by a Lambda function, that can create, update, and delete multiple AWS services. The Lambda function is responsible for implementing the logic to interact with the underlying services. This approach allows you to encapsulate the complexity of managing multiple services into a single CloudFormation resource that can be managed through the console. The Lambda function receives lifecycle events (Create, Update, Delete) from CloudFormation and performs the necessary actions.

Therefore, using a Custom Resource provides the flexibility to manage complex dependencies and orchestrate the creation and deletion of multiple AWS services as a single, manageable unit within CloudFormation.

Authoritative Links:

CloudFormation Custom Resources:
https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/template-custom-resources.html
Walkthrough: Creating Custom Resources with AWS Lambda:
https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/walkthrough-custom-resources-lambda.html

Question: 23

A new website will run on Amazon EC2 instances behind an Application Load Balancer. Amazon Route 53 will be used to manage DNS records.
What type of record should be set in Route 53 to point the website's apex domain name (for example, `company.com`) to the Application Load Balancer?

A. CNAME

B. SOA

C. TXT

D. ALIAS

Answer: D

Explanation:

Here's a detailed justification for why an ALIAS record is the correct answer, with supporting cloud computing concepts and authoritative links:

The question focuses on directing the website's apex domain (e.g., company.com) to an Application Load Balancer (ALB) using Route 53. Apex domains are special because they represent the root of the domain and cannot directly use CNAME records. CNAME records can only point to another domain name, not directly to an IP address or resource.

An ALIAS record is a Route 53 specific extension that is used to map an apex domain name (or a subdomain name) to an Elastic Load Balancer, CloudFront distribution, an Elastic Beanstalk environment, or an S3 bucket configured as a static website. Alias records are specific to Route 53 and are designed to overcome the limitations of CNAME records when dealing with apex domains.

While A records could theoretically point to the IP addresses of the ALB, this is not the correct approach. ALBs are dynamic resources; their IP addresses can change. Hardcoding IP addresses would lead to website downtime when the ALB's infrastructure changes. Also, using the IP address defeats the purpose of using an ALB's DNS name, which abstracts the underlying instances.

SOA (Start of Authority) records contain administrative information about the domain and zone, and TXT records hold arbitrary text data, neither of which would direct traffic to the ALB.

ALIAS records in Route 53 provide a seamless and robust way to map the apex domain to the ALB's DNS name. Route 53 automatically monitors the IP addresses associated with the ALB and updates the DNS record accordingly.

In summary, ALIAS records provide a dynamic and managed mechanism for resolving the apex domain to the ALB without requiring manual updates or exposing the underlying IP addresses, while still adhering to DNS constraints for apex domains.

Here are authoritative links for further research:

AWS Route 53 Documentation on Alias Records:
https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/resource-record-sets-choosing-alias-non-alias.html
AWS Whitepaper on DNS Services:https://d1.awsstatic.com/whitepapers/aws-best-practices-dns.pdf

Question: 24

A company is implementing security and compliance by using AWS Trusted Advisor. The company's SysOps team is validating the list of Trusted Advisor checks that it can access.
Which factor will affect the quantity of available Trusted Advisor checks?

A. Whether at least one Amazon EC2 instance is in the running state

B. The AWS Support plan

C. An AWS Organizations service control policy (SCP)

D. Whether the AWS account root user has multi-factor authentication (MFA) enabled

Answer: B

Explanation:

The number of AWS Trusted Advisor checks available to a user directly depends on the AWS Support plan associated with the AWS account. AWS Support plans range from Basic to Enterprise, and each tier unlocks a different set of Trusted Advisor checks. Basic support offers only core security checks, while Business and Enterprise support unlock the full suite of checks, including cost optimization, performance, security, fault tolerance, and service limits. Option A (EC2 instance running) is incorrect because the presence of an EC2 instance doesn't govern the availability of Trusted Advisor checks. Option C (SCP) is incorrect as SCPs primarily control the services and actions that IAM users and roles can perform within an organization, not the features available in Trusted Advisor. Option D (MFA for root user) is also incorrect because MFA for the root user is a security best practice, but it doesn't influence the number of Trusted Advisor checks available.

Therefore, the AWS Support plan is the decisive factor influencing which Trusted Advisor checks the company can access and use for security and compliance validation.

Refer to AWS documentation for more details: https://aws.amazon.com/premiumsupport/technology/trusted-advisor/ and https://docs.aws.amazon.com/awssupport/latest/user/trusted-advisor.html

Question: 25

A SysOps administrator is investigating issues on an Amazon RDS for MariaDB DB instance. The SysOps administrator wants to display the database load categorized by detailed wait events.
How can the SysOps administrator accomplish this goal?

A. Create an Amazon CloudWatch dashboard.
B. Enable Amazon RDS Performance Insights.
C. Enable and configure Enhanced Monitoring.
D. Review the database logs in Amazon CloudWatch Logs.

Answer: B

Explanation:

The correct answer is **B. Enable Amazon RDS Performance Insights.**

Here's why:

Amazon RDS Performance Insights is specifically designed for monitoring the performance of RDS database instances. It visualizes database load, allowing administrators to quickly identify performance bottlenecks and understand the root causes. Performance Insights displays metrics categorized by wait events, including detailed wait events, which provides insights into where the database is spending its time (e.g., CPU, I/O, lock contention). This aligns directly with the requirement of displaying database load categorized by detailed wait events.

Option A, creating an Amazon CloudWatch dashboard, could show general RDS metrics, but it won't give the same level of detail regarding specific wait events. CloudWatch provides high-level metrics, but it isn't granular enough to categorize database load by detailed wait events as required.

Option C, Enhanced Monitoring, provides metrics about the underlying OS of the RDS instance (CPU utilization, memory usage, disk I/O, etc.). While useful for troubleshooting, it doesn't directly show database load categorized by detailed wait events. Enhanced Monitoring operates at the instance level, not the database engine level where wait events are exposed.

Option D, reviewing database logs in CloudWatch Logs, requires manually parsing through potentially large

volumes of log data. While valuable for other troubleshooting scenarios, it is not an efficient way to identify and visualize database load categorized by detailed wait events. Analyzing logs requires significant effort to correlate entries with performance.

Therefore, Performance Insights is the optimal tool for the described scenario because it directly addresses the need to visualize database load categorized by detailed wait events in a user-friendly and insightful manner.

Authoritative Links:

Amazon RDS Performance Insights:
https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_PerfInsights.html Amazon CloudWatch: https://aws.amazon.com/cloudwatch/
Amazon RDS Enhanced Monitoring:
https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_Monitoring.OS.html

Question: 26

A company is planning to host an application on a set of Amazon EC2 instances that are distributed across multiple Availability Zones. The application must be able to scale to millions of requests each second.
A SysOps administrator must design a solution to distribute the traffic to the EC2 instances. The solution must be optimized to handle sudden and volatile traffic patterns while using a single static IP address for each Availability Zone.
Which solution will meet these requirements?

A. Amazon Simple Queue Service (Amazon SQS) queue
B. Application Load Balancer
C. AWS Global Accelerator
D. Network Load Balancer

Answer: D

Explanation:

The correct answer is **D. Network Load Balancer (NLB)**. Here's why:

High Throughput and Low Latency: NLBs are designed to handle millions of requests per second with extremely low latency, making them suitable for applications experiencing sudden and volatile traffic patterns. NLBs operate at Layer 4 (TCP/UDP) and forward traffic without inspecting the payload, minimizing processing overhead.
https://docs.aws.amazon.com/elasticloadbalancing/latest/network/introduction.html

Static IP Addresses per Availability Zone: NLBs provide a static IP address for each Availability Zone they are enabled in. This fulfils the requirement of using a single static IP address for each Availability Zone. Application Load Balancers (ALBs) also have static IPs, but lack the scaling capabilities of an NLB.

Traffic Distribution: NLBs distribute traffic to healthy EC2 instances across multiple Availability Zones. They can handle volatile and unpredictable traffic spikes efficiently because of their architecture.

Why other options are incorrect:

A. Amazon Simple Queue Service (Amazon SQS) queue: SQS is a message queuing service, not a load balancing solution. It decouples components of an application, enabling asynchronous communication, but it doesn't distribute real-time traffic to EC2 instances.

B. Application Load Balancer (ALB): ALBs operate at Layer 7 (HTTP/HTTPS) and are more suited for routing HTTP/HTTPS traffic based on content. While they offer advanced features like content-based routing and

host-based routing, they don't handle the same extreme throughput as NLBs. Although ALBs provide static IP addresses, NLBs are the better choice for high throughput and low latency.

C. AWS Global Accelerator: Global Accelerator is a service that directs traffic to optimal endpoints over the AWS global network, improving performance for geographically dispersed users. However, it doesn't inherently provide static IP addresses per Availability Zone in the same way an NLB does. While it uses static IPs, its primary purpose isn't to serve as a standard load balancer distributing traffic within a region and providing static IPs per AZ for that internal distribution. Instead, it provides two global static IPs to improve network performance for users across the world.

Question: 27

A company wants to be alerted through email when IAM CreateUser API calls are made within its AWS account. Which combination of actions should a SysOps administrator take to meet this requirement? (Choose two.)

A. Create an Amazon EventBridge (Amazon CloudWatch Events) rule with AWS CloudTrail as the event source and IAM CreateUser as the specific API call for the event pattern.

B. Create an Amazon EventBridge (Amazon CloudWatch Events) rule with Amazon CloudSearch as the event source and IAM CreateUser as the specific API call for the event pattern.

C. Create an Amazon EventBridge (Amazon CloudWatch Events) rule with AWS IAM Access Analyzer as the event source and IAM CreateUser as the specific API call for the event pattern.

D. Use an Amazon Simple Notification Service (Amazon SNS) topic as an event target with an email subscription.

E. Use an Amazon Simple Email Service (Amazon SES) notification as an event target with an email subscription.

Answer: AD

Explanation:

The correct answer is AD. Here's why:

A. Create an Amazon EventBridge (Amazon CloudWatch Events) rule with AWS CloudTrail as the event source and IAM CreateUser as the specific API call for the event pattern.

CloudTrail logs all API calls made to AWS services within an account. To monitor CreateUser API calls, you need a mechanism to detect these events from the CloudTrail logs. EventBridge is designed for this purpose. By creating an EventBridge rule with CloudTrail as the source and specifying CreateUser as the event pattern, EventBridge will trigger when this specific API call occurs. This action effectively detects the desired event.

D. Use an Amazon Simple Notification Service (Amazon SNS) topic as an event target with an email subscription.

Once the EventBridge rule detects the CreateUser event, it needs to trigger an action to notify the company.

SNS is a messaging service ideal for sending notifications. By configuring the EventBridge rule to send notifications to an SNS topic, and then subscribing an email address to that SNS topic, every time CreateUser is called, an email will be sent. SNS serves as the bridge for notifying the company via email.

Why other options are incorrect:

B. Create an Amazon EventBridge (Amazon CloudWatch Events) rule with Amazon CloudSearch as the event source and IAM CreateUser as the specific API call for the event pattern. CloudSearch is for searching within data and is not related to monitoring API calls to IAM.

C. Create an Amazon EventBridge (Amazon CloudWatch Events) rule with AWS IAM Access Analyzer as the

event source and IAM CreateUser as the specific API call for the event pattern. IAM Access Analyzer helps identify resource access patterns; it's not directly involved in monitoring API calls like CreateUser.

E. Use an Amazon Simple Email Service (Amazon SES) notification as an event target with an email subscription. While SES can send emails, it's not a direct target for EventBridge events. SNS is the recommended service for routing notifications to email subscribers from EventBridge. SES is better suited for direct application-driven emails. Furthermore, SES requires more initial setup (e.g., verifying email addresses).

In Summary:

EventBridge using CloudTrail is used to monitor API calls, and SNS with email subscription is the chosen notification service.

Authoritative Links:

AWS CloudTrail: Provides detailed documentation on CloudTrail's function of logging AWS API calls. Amazon EventBridge: Explains how EventBridge can be used as an event bus to respond to changes in your AWS environment.

Amazon SNS: Outlines how SNS enables application-to-application (A2A) and application-to-person (A2P) communication.

Question: 28

A SysOps administrator must configure Amazon S3 to host a simple nonproduction webpage. The SysOps administrator has created an empty S3 bucket from the
AWS Management Console. The S3 bucket has the default configuration in place.
Which combination of actions should the SysOps administrator take to complete this process? (Choose two.)

A. Configure the S3 bucket by using the "Redirect requests for an object" functionality to point to the bucket root URL.

B. Turn off the "Block all public access" setting. Allow public access by using a bucket ACL that contains <Permission>WEBSITE</Permission>.

C. Turn off the "Block all public access" setting. Allow public access by using a bucket ACL that allows access to the AuthenticatedUsers grantee.

D. Turn off the "Block all public access" setting. Set a bucket policy that allows "Principal": the s3:GetObject action.

E. Create an index.html document. Configure static website hosting, and upload the index document to the S3 bucket.

Answer: DE

Explanation:

The correct answer is DE. Here's a detailed justification:

To host a static website using Amazon S3, several key steps are required. First, the S3 bucket must be configured for static website hosting. This involves enabling the "Static website hosting" feature and specifying an index document (usually index.html). This handles requests to the root URL of the bucket. Therefore, option E is correct. You need to create the index.html file (which contains the content of your webpage), and then upload this file into the root of your configured S3 bucket.

However, by default S3 buckets prevent public access. To serve the static website to public users, you must disable "Block all public access". Then, the S3 bucket must be configured to allow public read access to the objects. The most reliable and recommended method is to set a bucket policy that allows the s3:GetObject action for public users. Therefore, option D is correct. This policy grants anyone access to read the objects within the bucket.

Option A is incorrect because "Redirect requests for an object" redirects requests to another URL, not for serving the content of the bucket itself. Option B is incorrect because the WEBSITE permission doesn't exist in S3 ACLs. Option C is incorrect because granting access to AuthenticatedUsers in ACLs provides access only to authenticated AWS users, not the general public trying to access a static website.

In summary, enabling static website hosting and setting a bucket policy allowing public read access are the two fundamental steps for hosting a static website with S3.

Relevant Documentation:

Hosting a static website on Amazon S3:
https://docs.aws.amazon.com/AmazonS3/latest/userguide/HostingWebSitesOnS3.html
Permissions for website access:
https://docs.aws.amazon.com/AmazonS3/latest/userguide/WebsiteAccessPermissionsReqd.html **Bucket Policies:**https://docs.aws.amazon.com/AmazonS3/latest/userguide/using-iam-policies.html

Question: 29

A company hosts its website on Amazon EC2 instances behind an Application Load Balancer. The company manages its DNS with Amazon Route 53, and wants to point its domain's zone apex to the website. Which type of record should be used to meet these requirements?

A. An AAAA record for the domain's zone apex
B. An A record for the domain's zone apex
C. A CNAME record for the domain's zone apex
D. An alias record for the domain's zone apex

Answer: D

Explanation:

The correct answer is D: An alias record for the domain's zone apex. Here's why:

An Application Load Balancer (ALB) dynamically manages IP addresses as it scales and handles traffic. Zone apex records (e.g., example.com) cannot traditionally use CNAME records because CNAME records require a specific hostname target, not dynamic IP addresses. A records can point to a static IPv4 address. However, since the ALB's IP addresses may change, directly using an A record isn't ideal or reliable. AAAA records are for IPv6 addresses and are not relevant if the ALB is configured for IPv4 traffic.

Alias records are a Route 53-specific extension that allows you to map your domain name to AWS resources like ALBs, ELBs, CloudFront distributions, S3 buckets configured for website hosting, and other Route 53 record sets. Alias records offer several benefits over A records in this scenario. They automatically update when the underlying IP addresses of the AWS resource change. Route 53 monitors the IP address changes associated with the ALB and keeps the DNS record up to date. This eliminates the need for manual updates or complex scripting.

Furthermore, alias records support zone apex records (the root domain). This makes them the ideal choice for pointing your domain's root to an ALB without the limitations of CNAME records. They also provide health checking, ensuring that traffic is only directed to healthy instances behind the ALB. Using an alias record is more efficient and reliable than using a standard A record that would require manual maintenance of IP address changes of the ALB.

Therefore, an alias record is the most suitable and recommended approach for pointing a domain's zone apex to an ALB managed in Route 53.

Further Reading:

AWS Route 53 Alias Records:https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/resource-record-sets-choosing-alias-non-alias.html
Using Alias Records:https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/resource-record-sets-alias.html

Question: 30

A user working in the Amazon EC2 console increased the size of an Amazon Elastic Block Store (Amazon EBS) volume attached to an Amazon EC2 Windows instance. The change is not reflected in the file system. What should a SysOps administrator do to resolve this issue?

A. Extend the file system with operating system-level tools to use the new storage capacity.
B. Reattach the EBS volume to the EC2 instance.
C. Reboot the EC2 instance that is attached to the EBS volume.
D. Take a snapshot of the EBS volume. Replace the original volume with a volume that is created from the snapshot.

Answer: A

Explanation:

The correct answer is A: Extend the file system with operating system-level tools to use the new storage capacity.

Here's a detailed justification:

When you increase the size of an EBS volume, the underlying block storage capacity is expanded. However, the operating system and its file system within the EC2 instance are not automatically aware of this change. The file system still sees the original volume size. To make use of the added storage, you must explicitly extend the file system within the operating system to utilize the newly available space.

For Windows instances, this typically involves using the Disk Management tool or the diskpart command-line utility. These tools allow you to extend the existing partition and file system to encompass the newly provisioned space on the EBS volume. Until you perform this step, the operating system will continue to operate as if the volume size hasn't changed.

Option B, reattaching the EBS volume, is incorrect. Reattaching will not automatically extend the file system. The OS will still see the old file system size.

Option C, rebooting the EC2 instance, is also incorrect. While rebooting might be necessary in some rare situations after extending the file system, it does not extend the file system itself.

Option D, taking a snapshot and replacing the volume, is an overly complex and unnecessary solution. Creating a snapshot and recreating a volume from it will not inherently extend the file system. It also introduces potential downtime during the volume replacement process. Furthermore, you'd still need to extend the file system after recreating the volume.

Therefore, extending the file system using operating system-level tools is the direct and correct approach to resolving this issue.

Here are authoritative links for further research:

Amazon EBS Elastic Volumes:https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-expand-volume.html

Extending a Windows Partition:https://learn.microsoft.com/en-us/windows-server/storage/disk-management/extend-a-volume

Question: 31

A SysOps administrator is using Amazon EC2 instances to host an application. The SysOps administrator needs to grant permissions for the application to access an Amazon DynamoDB table.
Which solution will meet this requirement?

A. Create access keys to access the DynamoDB table. Assign the access keys to the EC2 instance profile.
B. Create an EC2 key pair to access the DynamoDB table. Assign the key pair to the EC2 instance profile.
C. Create an IAM user to access the DynamoDB table. Assign the IAM user to the EC2 instance profile.
D. Create an IAM role to access the DynamoDB table. Assign the IAM role to the EC2 instance profile.

Answer: D

Explanation:

The correct answer is D: Create an IAM role to access the DynamoDB table. Assign the IAM role to the EC2 instance profile.

Here's why: IAM roles provide a secure way to grant permissions to applications running on EC2 instances to access AWS services, like DynamoDB, without embedding long-term credentials (such as access keys) directly in the application or the instance. An IAM role is essentially a collection of permissions policies.

When you assign an IAM role to an EC2 instance, the AWS security token service (STS) automatically provides temporary security credentials (access key ID, secret access key, and session token) to the instance. These credentials are automatically refreshed by the AWS SDK or CLI, ensuring they remain valid. The application running on the EC2 instance can then use these temporary credentials to access the specified DynamoDB table, without needing to manage or store sensitive access keys itself. This is a more secure and manageable approach.

Option A is incorrect because embedding access keys within the EC2 instance creates a security risk. If the instance is compromised, the access keys could be exposed. Also, managing the rotation of these keys becomes challenging.

Option B is incorrect because EC2 key pairs are used for SSH access to the EC2 instance itself and are not related to granting access to other AWS services like DynamoDB.

Option C is incorrect because assigning an IAM user to an EC2 instance profile is not the intended way to manage permissions for applications. IAM users are typically used for human users, not applications running on EC2 instances. IAM roles are designed specifically for this type of use case.

For further reading, refer to the official AWS documentation:

IAM Roles for EC2:https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles_use_switch-role-ec2.html **Using an IAM Role to Grant Permissions to Applications Running on Amazon EC2 Instances:** https://aws.amazon.com/blogs/security/using-an-iam-role-to-grant-permissions-to-applications-running-on-amazon-ec2-instances/

Question: 32

A SysOps administrator wants to protect objects in an Amazon S3 bucket from accidental overwrite and deletion. Noncurrent objects must be kept for 90 days and then must be permanently deleted. Objects must reside within

the same AWS Region as the original S3 bucket.
Which solution meets these requirements?

A. Create an Amazon Data Lifecycle Manager (Amazon DLM) lifecycle policy for the S3 bucket. Add a rule to the lifecycle policy to delete noncurrent objects after 90 days.

B. Create an AWS Backup policy for the S3 bucket. Create a backup rule that includes a lifecycle to expire noncurrent objects after 90 days.

C. Enable S3 Cross-Region Replication on the S3 bucket. Create an S3 Lifecycle policy for the bucket to expire noncurrent objects after 90 days.

D. Enable S3 Versioning on the S3 bucket. Create an S3 Lifecycle policy for the bucket to expire noncurrent objects after 90 days.

Answer: D

Explanation:

The correct answer is D because it leverages S3 Versioning and Lifecycle policies, which are key components for data protection and lifecycle management in S3.

S3 Versioning protects against accidental overwrites and deletions. When versioning is enabled, every object write creates a new version of the object, while deletions create delete markers instead of permanently removing the object. This ensures that previous versions are retained.

S3 Lifecycle policies automate the process of transitioning or expiring objects after a specified period. The requirement to retain noncurrent objects for 90 days and then permanently delete them is precisely what an S3 Lifecycle policy is designed for. You configure the policy to expire noncurrent versions after 90 days, ensuring compliance with the data retention requirement.

Option A is incorrect because Amazon Data Lifecycle Manager (DLM) is designed primarily for EBS volumes and snapshots, not S3 objects. Option B is incorrect because AWS Backup primarily focuses on backing up and restoring entire resources (like EC2 instances, databases, and EFS file systems), and its lifecycle management is more oriented toward backup retention rather than managing individual object versions within S3. Option C is incorrect because S3 Cross-Region Replication (CRR) creates copies of objects in a different AWS Region, which does not meet the requirement that objects must reside within the same AWS Region as the original S3 bucket. CRR is also primarily for disaster recovery, not for protecting against accidental overwrites or deletions within the same region. While you could use a lifecycle policy after CRR, versioning is the necessary first step for data protection.

Therefore, enabling S3 Versioning and then using a Lifecycle policy to expire noncurrent versions meets all the requirements of the scenario in the most efficient and cost-effective manner.

Further research:

S3 Versioning:https://docs.aws.amazon.com/AmazonS3/latest/userguide/Versioning.html
S3 Lifecycle:https://docs.aws.amazon.com/AmazonS3/latest/userguide/object-lifecycle-mgmt.html

Question: 33

A company has an application that customers use to search for records on a website. The application's data is stored in an Amazon Aurora DB cluster. The application's usage varies by season and by day of the week. The website's popularity is increasing, and the website is experiencing slower performance because of increased load on the DB cluster during periods of peak activity. The application logs show that the performance issues occur when users are searching for information. The same search is rarely performed multiple times.
A SysOps administrator must improve the performance of the platform by using a solution that maximizes resource efficiency.
Which solution will meet these requirements?

A. Deploy an Amazon ElastiCache for Redis cluster in front of the DB cluster. Modify the application to check the cache before the application issues new queries to the database. Add the results of any queries to the cache.

B. Deploy an Aurora Replica for the DB cluster. Modify the application to use the reader endpoint for search operations. Use Aurora Auto Scaling to scale the number of replicas based on load.

C. Use Provisioned IOPS on the storage volumes that support the DB cluster to improve performance sufficiently to support the peak load on the application.

D. Increase the instance size in the DB cluster to a size that is sufficient to support the peak load on the application. Use Aurora Auto Scaling to scale the instance size based on load.

Answer: B

Explanation:

The correct answer is B. Here's why:

The problem describes a read-heavy workload (searches) that experiences performance issues during peak activity on an Aurora DB cluster. The search queries are not repetitive, meaning that caching the results (option A) would provide minimal benefit. The problem states that the same search is rarely performed multiple times, so adding the results to the cache won't help with performance.

Option B suggests deploying an Aurora Replica and directing read traffic (search operations) to it. This solution effectively offloads the read workload from the primary Aurora instance, reducing the load and improving performance, as described in the problem. Aurora Auto Scaling dynamically adjusts the number of replicas based on load, ensuring that resource efficiency is maximized. This is crucial because usage varies by season and day of the week.

Option C suggests using Provisioned IOPS. While increasing IOPS can improve performance, it's more relevant for write-heavy workloads or scenarios where IOPS is the bottleneck. It's also more costly than simply using read replicas.

Option D suggests increasing the instance size and using Aurora Auto Scaling to scale the instance size. While this can also improve performance, it's less efficient than using read replicas for read-heavy workloads.

Scaling the instance size impacts both read and write capacity, and is more expensive. Since the problem identifies searches (reads) as the source of performance degradation, targeting that specific area (reads) with read replicas is a more efficient strategy. Also, scaling up an instance can take time.

In summary, using Aurora Read Replicas is the most cost-effective and efficient way to improve the performance of a read-heavy Aurora database workload, especially when combined with Aurora Auto Scaling.

Here are some authoritative links for further research:

Aurora Read Replicas:
https://docs.aws.amazon.com/AmazonRDS/latest/AuroraUserGuide/Aurora.Managing.Replication.html **Aurora Auto Scaling:**
https://docs.aws.amazon.com/AmazonRDS/latest/AuroraUserGuide/Aurora.Managing.Performance.html

Question: 34

A company uses AWS Organizations to manage multiple AWS accounts. Corporate policy mandates that only specific AWS Regions can be used to store and process customer data. A SysOps administrator must prevent the provisioning of Amazon EC2 instances in unauthorized Regions by anyone in the company.
What is the MOST operationally efficient solution that meets these requirements?

A. Configure AWS CloudTrail in all Regions to record all API activity. Create an Amazon EventBridge (Amazon CloudWatch Events) rule in all unauthorized Regions for ec2:RunInstances events. Use AWS Lambda to terminate the launched EC2 instances.

B. In each AWS account, create a managed IAM policy that uses a Region condition to deny the ec2:RunInstances action in all unauthorized Regions. Attach this policy to all IAM groups in each AWS account.

C. In each AWS account, create an IAM permissions boundary policy that uses a Region condition to deny the ec2:RunInstances action in all unauthorized Regions. Attach the permissions boundary policy to all IAM users in each AWS account.

D. Create a service control policy (SCP) in AWS Organizations to deny the ec2:RunInstances action in all unauthorized Regions. Attach this policy to the root level of the organization.

Answer: D

Explanation:

The correct answer is D because it offers the most operationally efficient and centrally managed solution to enforce the regional restrictions across all AWS accounts within the AWS Organization. SCPs are designed for central governance and provide guardrails at the organization level, directly preventing the provisioning of EC2 instances in unauthorized regions. This is a preventative measure, stopping the action from occurring in the first place. Attaching the SCP to the root level ensures that the policy applies to all accounts and organizational units (OUs) unless explicitly overridden.

Option A is less efficient because it relies on reactive termination of instances after they are launched. It requires CloudTrail to record API activity, EventBridge rules to detect ec2:RunInstances events, and Lambda functions to terminate the instances. This adds complexity, potential delays, and unnecessary resource consumption. Moreover, it doesn't prevent the initial attempt to launch instances in unauthorized regions, which generates audit noise and may incur temporary costs.

Options B and C require managing IAM policies or permissions boundaries in each AWS account separately. This introduces significant administrative overhead and the risk of inconsistencies across accounts. While these options can achieve the desired outcome, they are less scalable and harder to maintain compared to using SCPs. They also place the responsibility on individual account administrators to correctly implement and maintain the regional restrictions.

SCPs, in contrast, provide a centralized mechanism to enforce organizational policies without relying on individual account configurations. This simplifies management, reduces the risk of errors, and ensures consistent enforcement of the corporate policy across all AWS accounts. SCPs provide a definitive block on the action at the organization level, which aligns with the requirement of preventing provisioning in the unauthorized regions, making it the most operationally efficient approach.

Relevant links for further research:

AWS Organizations Service Control Policies (SCPs): https://docs.aws.amazon.com/organizations/latest/userguide/orgs_manage_policies_scp.html IAM Policies: https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies.html AWS CloudTrail: https://aws.amazon.com/cloudtrail/
Amazon EventBridge: https://aws.amazon.com/eventbridge/

Question: 35

A company's public website is hosted in an Amazon S3 bucket in the us-east-1 Region behind an Amazon CloudFront distribution. The company wants to ensure that the website is protected from DDoS attacks. A SysOps administrator needs to deploy a solution that gives the company the ability to maintain control over the rate limit at which DDoS protections are applied.
Which solution will meet these requirements?

A. Deploy a global-scoped AWS WAF web ACL with an allow default action. Configure an AWS WAF rate-based rule to block matching traffic. Associate the web ACL with the CloudFront distribution.

B. Deploy an AWS WAF web ACL with an allow default action in us-east-1. Configure an AWS WAF rate-based

rule to block matching traffic. Associate the web ACL with the S3 bucket.

C. Deploy a global-scoped AWS WAF web ACL with a block default action. Configure an AWS WAF rate-based rule to allow matching traffic. Associate the web ACL with the CloudFront distribution.

D. Deploy an AWS WAF web ACL with a block default action in us-east-1. Configure an AWS WAF rate-based rule to allow matching traffic. Associate the web ACL with the S3 bucket.

Answer: A

Explanation:

Here's a detailed justification for why option A is the correct solution, along with relevant supporting information:

The primary goal is to protect a public website hosted in S3 behind CloudFront from DDoS attacks while maintaining control over rate limiting. AWS WAF (Web Application Firewall) is the service designed for this purpose. To protect a CloudFront distribution, the WAF needs to be associated with the CloudFront distribution itself, not the S3 bucket directly. Therefore, options B and D are incorrect.

AWS WAF rules are evaluated in order. The default action defines what happens if no rules match. A default action of "allow" with a rate-based rule that blocks matching traffic allows you to control the traffic that exceeds a defined threshold. The rate-based rule is the mechanism to maintain control over the rate limit. If a request doesn't match any of the rules (in this case, the rate-based rule), the default action (allow) is applied.
Conversely, a default action of "block" would block all traffic that doesn't explicitly match an allow rule.

The crucial aspect here is DDoS protection. DDoS attacks are often distributed across multiple geographic locations. To effectively mitigate these attacks at the edge, the AWS WAF should be configured as a global service associated with CloudFront. AWS WAF deployed globally (i.e. associated with CloudFront) provides low-latency inspection for all HTTP/S requests.

Therefore, option A, deploying a global-scoped AWS WAF with an allow default action and a rate-based rule to block matching traffic, associated with the CloudFront distribution, is the correct solution because it directly protects the CloudFront distribution (the edge point of access), uses rate-limiting to protect from DDoS attacks, and provides controlled blocking based on exceeded request limits.

Key considerations include:

CloudFront Association: AWS WAF should be associated with CloudFront for edge protection.
Global Scope: WAF rules targeting CloudFront must have global scope.
Default Action: An "allow" default action lets all traffic through unless a specific rule blocks it.
Rate-Based Rules: Give you granular control over the rate limit.
AWS WAF: The service designed to protect web applications and APIs from attacks using customizable rules.

Supporting Documentation:

AWS WAF documentation: Official AWS documentation for WAF.
Rate-Based Rule
Using AWS WAF with CloudFront

Question: 36

A SysOps administrator developed a Python script that uses the AWS SDK to conduct several maintenance tasks. The script needs to run automatically every night.
What is the MOST operationally efficient solution that meets this requirement?

A. Convert the Python script to an AWS Lambda function. Use an Amazon EventBridge (Amazon CloudWatch Events) rule to invoke the function every night.

B. Convert the Python script to an AWS Lambda function. Use AWS CloudTrail to invoke the function every night.

C. Deploy the Python script to an Amazon EC2 instance. Use Amazon EventBride (Amazon CloudWatch Events) to schedule the instance to start and stop every night.

D. Deploy the Python script to an Amazon EC2 instance. Use AWS Systems Manager to schedule the instance to start and stop every night.

Answer: A

Explanation:

Here's a detailed justification for why option A is the most operationally efficient solution for automating the Python script:

Option A proposes using AWS Lambda and Amazon EventBridge (formerly CloudWatch Events) which is the optimal choice because it leverages serverless technology. Lambda eliminates the need to manage any underlying infrastructure like EC2 instances. This significantly reduces operational overhead as there's no need for patching, OS maintenance, or instance scaling. The script only runs when triggered, conserving resources and costs.

EventBridge provides a straightforward and reliable way to schedule the Lambda function execution. You can define a cron expression to trigger the function every night. This tight integration between EventBridge and Lambda simplifies the scheduling process. CloudTrail, as suggested in option B, is an auditing service and is not designed for scheduling events.

Options C and D, which involve deploying the script to an EC2 instance, are less operationally efficient. They require managing an EC2 instance, including patching, security updates, and ensuring the instance is running and available. Although these options also offer a scheduler, the operational burden is much higher. While Systems Manager can automate EC2 instance management, it still introduces complexity compared to the serverless approach. Furthermore, constantly starting and stopping an EC2 instance incurs additional operational overhead and introduces potential for errors during startup/shutdown. Also, running a whole EC2 instance just to execute a simple script regularly is far more costly than using Lambda.

The serverless approach of Lambda and EventBridge offers the best balance of automation, cost efficiency, and minimal operational overhead for this specific use case.

Authoritative Links:

AWS Lambda:https://aws.amazon.com/lambda/
Amazon EventBridge:https://aws.amazon.com/eventbridge/

Question: 37

A SysOps administrator must create a solution that immediately notifies software developers if an AWS Lambda function experiences an error.
Which solution will meet this requirement?

A. Create an Amazon Simple Notification Service (Amazon SNS) topic with an email subscription for each developer. Create an Amazon CloudWatch alarm by using the Errors metric and the Lambda function name as a dimension. Configure the alarm to send a notification to the SNS topic when the alarm state reaches ALARM.

B. Create an Amazon Simple Notification Service (Amazon SNS) topic with a mobile subscription for each developer. Create an Amazon EventBridge (Amazon CloudWatch Events) alarm by using the LambdaError as the event pattern and the SNS topic name as a resource. Configure the alarm to send a notification to the SNS topic when the alarm state reaches ALARM.

C. Verify each developer email address in Amazon Simple Email Service (Amazon SES). Create an Amazon CloudWatch rule by using the LambdaError metric and developer email addresses as dimensions. Configure the rule to send an email through Amazon SES when the rule state reaches ALARM.

D. Verify each developer mobile phone in Amazon Simple Email Service (Amazon SES). Create an Amazon EventBridge (Amazon CloudWatch Events) rule by using Error as the event pattern and the Lambda function name as a resource. Configure the rule to send a push notification through Amazon SES when the rule state reaches ALARM.

Answer: A

Explanation:

The correct answer is A because it utilizes the appropriate AWS services for monitoring Lambda function errors and sending notifications. Here's a detailed breakdown:

Amazon SNS (Simple Notification Service): SNS is a highly scalable and flexible messaging service that's perfect for sending notifications to multiple subscribers. Creating an SNS topic allows for a single point of publishing notifications, and developers can subscribe to this topic via email to receive error alerts.

Amazon CloudWatch Alarms: CloudWatch Alarms monitor specific metrics and trigger actions when those metrics cross a defined threshold. In this scenario, we need to monitor the Errors metric for the Lambda function.

Errors Metric and Lambda Function Name as a Dimension: Specifying the Lambda function name as a dimension ensures that the CloudWatch Alarm is only monitoring the Errors metric for that specific Lambda function, and not all Lambda functions.

ALARM State: Configuring the CloudWatch Alarm to notify the SNS topic when it enters the ALARM state guarantees that developers receive a notification only when the error threshold has been breached, preventing unnecessary alerts.

Option B is incorrect because EventBridge (formerly CloudWatch Events) is generally used for reacting to state changes in your AWS resources, not primarily for alerting based on metrics. While it can be used in this way, CloudWatch Alarms are a more direct and suitable solution for monitoring metrics like Lambda errors.

Using mobile subscriptions directly might also be more cumbersome than email for many developers. The concept of an EventBridge alarm isn't quite right; it's more about reacting to events.

Option C introduces unnecessary complexity by involving Amazon SES (Simple Email Service) for email verification and sending. While SES can be used to send emails, SNS provides a simpler and more managed notification service directly integrated with CloudWatch Alarms. Also, CloudWatch rules are typically used for event-driven actions rather than metric monitoring in the same way CloudWatch alarms are.

Option D similarly misuses Amazon SES for mobile push notifications. SNS is the more appropriate service for sending both email and mobile push notifications. Also EventBridge rule for error metric is not as straightforward as using CloudWatch Alarms.

In summary, option A is the most efficient and cost-effective solution using the appropriate AWS services to monitor Lambda function errors and notify developers promptly.

Supporting Links:

Amazon SNS:https://aws.amazon.com/sns/
Amazon CloudWatch:https://aws.amazon.com/cloudwatch/
CloudWatch Alarms:
https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/AlarmThatSendsEmail.html
Monitoring Lambda Function with CloudWatch:https://docs.aws.amazon.com/lambda/latest/dg/monitoring-functions.html

Question: 38

A company has a private Amazon S3 bucket that contains sensitive information. A SysOps administrator needs to keep logs of the IP addresses from authentication failures that result from attempts to access objects in the bucket. The logs must be stored so that they cannot be overwritten or deleted for 90 days.
Which solution will meet these requirements?

A. Create an AWS CloudTrail trail. Configure the log files to be saved to Amazon CloudWatch Logs. Configure the log group with a retention period of 90 days.

B. Create an AWS CloudTrail trail. Configure the log files to be saved to a different S3 bucket. Turn on CloudTrail log file integrity validation for 90 days.

C. Turn on access logging for the S3 bucket. Configure the access logs to be saved to Amazon CloudWatch Logs. Configure the log group with a retention period of 90 days.

D. Turn on access logging for the S3 bucket. Configure the access logs to be saved in a second S3 bucket. Turn on S3 Object Lock on the second S3 bucket, and configure a default retention period of 90 days.

Answer: A

Explanation:

The correct answer is **D**. Here's a detailed justification:

Why Option D is Correct:

S3 Access Logging: The core requirement is to capture authentication failures against the S3 bucket. S3 access logging directly records access requests to the bucket, including failed authentication attempts (as indicated by HTTP 403 errors). (https://docs.aws.amazon.com/AmazonS3/latest/userguide/logging-with-S3.html)
Separate Bucket: Storing the access logs in a separate S3 bucket is a best practice for security and manageability. It prevents accidental modification or deletion of logs within the same bucket they're tracking.

S3 Object Lock: S3 Object Lock is crucial for meeting the immutability requirement. It prevents objects (in this case, access logs) from being deleted or overwritten during a specified retention period. Configuring a 90-day retention period ensures the logs are protected for the necessary duration.
(https://docs.aws.amazon.com/AmazonS3/latest/userguide/object-lock.html)

Why Other Options are Incorrect:

Option A (CloudTrail to CloudWatch Logs): While CloudTrail is useful for auditing API calls, it's not designed to capture S3 access logs in the same detailed way as S3's built-in access logging. CloudTrail tracks who made API calls against S3, not necessarily every attempted access, including failed authentications at the object level. Also, while CloudWatch Logs can store logs for a specified retention, they're not as immutable as S3 Object Lock, thus it can be configured in such a way that it is deleted.

Option B (CloudTrail to S3 with Integrity Validation): Similar to Option A, CloudTrail doesn't capture the specific authentication failures required. CloudTrail log file integrity validation ensures the CloudTrail logs themselves haven't been tampered with after delivery to S3, but it doesn't protect against the logs being deleted altogether (Object Lock does this).

Option C (S3 Access Logging to CloudWatch Logs): This option correctly utilizes S3 access logging. However, CloudWatch Logs, while providing retention, doesn't offer the same strong immutability guarantees as S3 Object Lock, thus it can be configured in such a way that it is deleted before 90 days.

In summary, Option D combines S3's built-in access logging with the immutability of S3 Object Lock to fulfill all the requirements of capturing authentication failure logs and protecting them from deletion or overwriting for 90 days.

Question: 39

A company is migrating its production file server to AWS. All data that is stored on the file server must remain accessible if an Availability Zone becomes unavailable or when system maintenance is performed. Users must be able to interact with the file server through the SMB protocol. Users also must have the ability to manage file permissions by using Windows ACLs.
Which solution will net these requirements?

A. Create a single AWS Storage Gateway file gateway.

B. Create an Amazon FSx for Windows File Server Multi-AZ file system.

C. Deploy two AWS Storage Gateway file gateways across two Availability Zones. Configure an Application Load Balancer in front of the file gateways.

D. Deploy two Amazon FSx for Windows File Server Single-AZ 2 file systems. Configure Microsoft Distributed File System Replication (DFSR).

Answer: B

Explanation:

The correct answer is B: Create an Amazon FSx for Windows File Server Multi-AZ file system. Here's why:

High Availability: The question explicitly states the need for data accessibility even during Availability Zone outages or system maintenance. Amazon FSx for Windows File Server Multi-AZ is designed to provide high availability. It synchronously replicates data across two Availability Zones, ensuring failover capabilities in case of an AZ failure.

SMB Protocol: The requirement for users to interact via the SMB protocol is directly addressed. Amazon FSx for Windows File Server natively supports the SMB protocol, allowing seamless integration with existing Windows environments.

Windows ACLs: Windows Access Control Lists (ACLs) are crucial for file permissions management. Amazon FSx for Windows File Server fully supports Windows ACLs, providing granular control over file access for users and groups.

Storage Gateway limitations: While AWS Storage Gateway file gateway allows on-premises applications to store files in Amazon S3, using it alone does not satisfy the high availability and Windows ACL requirements as directly and efficiently as FSx for Windows File Server. Options A and C using Storage Gateway do not provide native support for SMB and Windows ACLs with high availability in the same way. Also the set up and management of the file share would be significantly more complex.

FSx for Windows File Server Single-AZ limitation: Option D involves deploying Single-AZ file systems, which directly contradicts the requirement for resilience against Availability Zone failures. DFSR setup would also be complex, and also the question explicitly requests a Multi-AZ solution.

Authoritative Links:

Amazon FSx for Windows File Server:https://aws.amazon.com/fsx/windows/
Amazon FSx for Windows File Server High Availability:
https://docs.aws.amazon.com/fsx/latest/WindowsGuide/high-availability.html

Question: 40

A company runs an application on an Amazon EC2 instance. A SysOps administrator creates an Auto Scaling group and an Application Load Balancer (ALB) to handle an increase in demand. However, the EC2 instances are failing the health check.
What should the SysOps administrator do to troubleshoot this issue?

A. Verify that the Auto Scaling group is configured to use all AWS Regions.

B. Verify that the application is running on the protocol and the port that the listener is expecting.

C. Verify the listener priority in the ALB. Change the priority if necessary.

D. Verify the maximum number of instances in the Auto Scaling group. Change the number if necessary.

Answer: B

Explanation:

The correct answer is B: Verify that the application is running on the protocol and the port that the listener is expecting.

Here's a detailed justification:

When EC2 instances fail the Application Load Balancer's (ALB) health checks, it indicates that the ALB cannot successfully communicate with the instances. Several factors can cause this, but one of the most common and fundamental is a mismatch between the ALB's expectations and the application's actual configuration.

The ALB's listener is configured to forward traffic to the instances using a specific protocol (e.g., HTTP, HTTPS) and port (e.g., 80, 443). If the application on the EC2 instance is not listening on that same protocol and port, the ALB will be unable to establish a connection, resulting in a failed health check.

Option A is incorrect because the failure of health checks is not directly related to the Auto Scaling group's configuration across AWS Regions. While multi-region deployments are beneficial for high availability, a failed health check within a single region would cause issues before region-specific considerations become relevant.

Option C is incorrect because listener priority in the ALB primarily determines the order in which rules are evaluated, which dictate how traffic is routed to different target groups. While a misconfigured listener rule could theoretically prevent traffic from reaching the instance at all, it is less common for a health check failure due to a rule issue, and it is more fundamental to ensure the connection works in the first place.

Option D is incorrect because the maximum number of instances in the Auto Scaling group is a capacity setting and doesn't directly affect whether individual instances are healthy and responsive to the ALB.

Therefore, the most immediate and logical step is to ensure that the application is indeed listening on the protocol and port specified in the ALB's listener configuration. This ensures that the ALB can successfully send health check requests and receive valid responses from the EC2 instances.

For further research, consider these authoritative links:

AWS Documentation on ALB Health Checks:
https://docs.aws.amazon.com/elasticloadbalancing/latest/application/target-group-health-checks.html **AWS Documentation on ALB Listeners:**
https://docs.aws.amazon.com/elasticloadbalancing/latest/application/load-balancer-listeners.html

Question: 41

A SysOps administrator has created an AWS Service Catalog portfolio and has shared the portfolio with a second AWS account in the company. The second account is controlled by a different administrator.
Which action will the administrator of the second account be able to perform?

A. Add a product from the imported portfolio to a local portfolio.

B. Add new products to the imported portfolio.

C. Change the launch role for the products contained in the imported portfolio.

D. Customize the products in the imported portfolio.

Answer: A

Explanation:

The correct answer is A. An administrator in the second AWS account, after the Service Catalog portfolio has been shared with them, can add products from the imported portfolio to a new, locally managed portfolio within their own account. This allows them to curate and organize the shared products according to their own specific needs and governance policies within their account. They are essentially referencing the products from the original, shared portfolio.

Option B is incorrect because the administrator of the second account cannot modify the imported portfolio itself. They do not have permissions to add new products directly to the portfolio that originates from the first account. The original administrator of the first account retains control over its portfolio's content.

Option C is incorrect because the ability to change the launch role, which defines the AWS Identity and Access Management (IAM) role used to launch the provisioned products, remains with the administrator of the account where the Service Catalog portfolio was initially created. The second account administrator is consuming the catalog, not managing its core operational aspects.

Option D is incorrect. While the administrator of the second account can customize the provisioned products launched from the shared catalog, they cannot directly customize the products within the imported portfolio itself. Customization after provisioning involves adjusting settings and parameters specific to the launched instance of the product. The product definition in the originating portfolio remains unchanged by the second account.

In summary, sharing a Service Catalog portfolio gives the second account administrator the ability to consume the products and organize them within their own account structure, but it doesn't grant them the ability to modify the original portfolio or its underlying product definitions.

Relevant documentation:

AWS Service Catalog documentation:
https://docs.aws.amazon.com/servicecatalog/latest/adminguide/whatis.html
Sharing portfolios: https://docs.aws.amazon.com/servicecatalog/latest/adminguide/portfolios-share.html

Question: 42

A company has migrated its application to AWS. The company will host the application on Amazon EC2 instances of multiple instance families.
During initial testing, a SysOps administrator identifies performance issues on selected EC2 instances. The company has a strict budget allocation policy, so the
SysOps administrator must use the right resource types with the performance characteristics to match the workload.
What should the SysOps administrator do to meet this requirement?

A. Purchase regional Reserved Instances (RIs) for immediate cost savings. Review and take action on the EC2 rightsizing recommendations in Cost Explorer. Exchange the RIs for the optimal instance family after rightsizing.

B. Purchase zonal Reserved Instances (RIs) for the existing instances. Monitor the RI utilization in the AWS Billing and Cost Management console. Make adjustments to instance sizes to optimize utilization.

C. Review and take action on AWS Compute Optimizer recommendations. Purchase Compute Savings Plans to reduce the cost that is required to run the compute resources.

D. Review resource utilization metrics in the AWS Cost and Usage Report. Rightsize the EC2 instances. Create On-Demand Capacity Reservations for the rightsized resources.

Answer: C

Explanation:

The correct answer is C because it directly addresses the problem of performance issues and budget constraints while ensuring optimal resource utilization. Here's a detailed justification:

Why C is the best approach:

AWS Compute Optimizer: This service analyzes the historical utilization of your EC2 instances (CPU, memory, I/O, network) and provides recommendations on the optimal instance types for your workloads. This addresses the performance issues by suggesting instances that are better suited to the application's needs. (https://aws.amazon.com/compute-optimizer/)

Compute Savings Plans: These plans offer significant cost savings in exchange for a commitment to a consistent amount of compute usage (measured in $/hour) for a 1- or 3-year term. They provide flexibility across instance families, operating systems, and AWS Regions, allowing you to change instance types while still benefiting from the savings. This aligns with the company's strict budget allocation policy. Savings Plans are preferred over RIs as they provide more flexibility. (https://aws.amazon.com/savingsplans/)

Why other options are less suitable:

A: Purchasing Regional Reserved Instances (RIs) before rightsizing is premature. While RIs offer cost savings, purchasing them without understanding the optimal instance type can lead to wasted money if the instances are not the right size for the workload. Additionally, exchanging RIs can be complex and might not always be possible without financial penalties. Cost Explorer rightsizing recommendations are helpful, but Compute Optimizer provides more comprehensive guidance on the type of instance, not just the size.

B: Purchasing Zonal Reserved Instances (RIs) also locks you into specific instance types and availability zones before rightsizing, which is inefficient and potentially wasteful. Monitoring RI utilization is important, but it doesn't proactively address the performance issues stemming from potentially incorrect instance types. Adjusting instance sizes is a reactive approach, and might not solve the underlying problem of needing a different instance family.

D: Reviewing the AWS Cost and Usage Report (CUR) for resource utilization metrics is a good practice, but it requires manual analysis and might not provide the same level of insight as Compute Optimizer. While rightsizing is necessary, creating On-Demand Capacity Reservations after rightsizing doesn't address cost optimization as directly as Savings Plans, and reservations are only beneficial if you are concerned about the availability of capacity in a specific Availability Zone. It's more expensive than Savings Plans.

In summary, answer C provides a proactive, data-driven approach to both resolving performance issues and optimizing costs, making it the most effective solution given the company's requirements. Compute Optimizer ensures the application runs on the most appropriate instance types for its workload, and Savings Plans guarantee cost savings while retaining the flexibility to adapt to changing application needs.

Question: 43

A SysOps administrator is tasked with deploying a company's infrastructure as code. The SysOps administrator want to write a single template that can be reused for multiple environments.
How should the SysOps administrator use AWS CloudFormation to create a solution?

A. Use Amazon EC2 user data in a CloudFormation template.
B. Use nested stacks to provision resources.
C. Use parameters in a CloudFormation template.
D. Use stack policies to provision resources.

Answer: C

Explanation:

The correct answer is C: Use parameters in a CloudFormation template.

Here's why:

CloudFormation parameters allow you to pass values into your template when you create or update a stack. These values can be used to customize resource properties, making the same template deploy differently across multiple environments (e.g., development, staging, production). By defining parameters, you create a flexible and reusable template that adapts to specific environment requirements. This is ideal for infrastructure as code, where consistency and reusability are paramount.

Option A, using EC2 user data, primarily focuses on configuring instances after they're launched. While helpful for initial setup, it doesn't inherently facilitate reusing the same template across multiple environments with varying configurations. User data scripts could use parameters passed from the CloudFormation template, but it's not the core method for reusability.

Option B, nested stacks, breaks down a large infrastructure into smaller, manageable stacks. This improves modularity and organization but doesn't inherently solve the problem of making a single template adaptable to different environments. While you could pass parameters to nested stacks, it's still parameters that are driving the environment-specific configuration, not the nested stack structure itself. Nested stacks are more about logical grouping of related resources.

Option D, stack policies, control what actions users can perform on resources within a stack. While important for governance and security, they don't enable environment-specific customization. Stack policies are designed to prevent accidental or unauthorized changes, not to vary resource properties based on the environment.

In conclusion, parameters are the most direct and efficient way to create a single CloudFormation template that can be reused for multiple environments by allowing environment-specific values to be passed in during stack creation or update.

For further research:

AWS CloudFormation Parameters:
https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/parameters-section-structure.html

Question: 44

A SysOps administrator is responsible for a large fleet of Amazon EC2 instances and must know whether any instances will be affected by upcoming hardware maintenance.
Which option would provide this information with the LEAST administrative overhead?

A. Deploy a third-party monitoring solution to provide real-time EC2 instance monitoring.
B. List any instances with failed system status checks using the AWS Management Console.
C. Monitor AWS CloudTrail for StopInstances API calls.
D. Review the AWS Personal Health Dashboard.

Answer: D

Explanation:

The AWS Personal Health Dashboard (PHD) is the most efficient solution because it's a dedicated service providing alerts and remediation guidance when AWS experiences events that might affect you. It proactively informs users about planned maintenance activities, scheduled outages, and potential performance

degradation related to AWS services impacting their resources. This includes EC2 instances.

Option A involves significant overhead. Deploying and managing a third-party monitoring solution requires initial setup, configuration, ongoing maintenance, and potentially complex integrations. It's overkill just to determine maintenance schedules.

Option B, listing instances with failed system status checks, is reactive. System status checks fail only after an issue occurs. It doesn't provide advance warning of scheduled maintenance. While useful for incident response, it's not a preventative measure.

Option C, monitoring CloudTrail for StopInstances API calls, is also reactive and incomplete. While a StopInstances API call could indicate maintenance, it doesn't necessarily mean scheduled AWS maintenance.

It could be due to other reasons like scaling down, user error, or automated scripts. Furthermore, AWS maintenance doesn't always involve stopping instances. A live migration might be possible, resulting in no StopInstances API call.

Therefore, PHD is the best option as it provides proactive notifications specifically related to planned AWS maintenance impacting your EC2 instances, thereby requiring the least administrative effort. The administrator doesn't need to actively monitor logs or implement complex monitoring solutions.

Authoritative links for further research:

AWS Personal Health Dashboard:https://aws.amazon.com/premiumsupport/technology/personal-health-dashboard/
AWS Health User Guide:https://docs.aws.amazon.com/health/latest/ug/what-is-aws-health.html

Question: 45

A SysOps administrator is attempting to deploy resources by using an AWS CloudFormation template. An Amazon EC2 instance that is defined in the template fails to launch and produces an InsufficientInstanceCapacity error.
Which actions should the SysOps administrator take to resolve this error? (Choose two.)

A. Create a separate AWS CloudFormation template for the EC2 instance.
B. Modify the AWS CloudFormation template to not specify an Availability Zone for the EC2 instance.
C. Modify the AWS CloudFormation template to use a different EC2 instance type.
D. Use a different Amazon Machine Image (AMI) for the EC2 instance.
E. Use the AWS CLI's validate-template command before creating a stack from the template.

Answer: BC

Explanation:

The InsufficientInstanceCapacity error in AWS CloudFormation indicates that the requested instance type is not currently available in the specified Availability Zone due to high demand. To address this, the administrator needs to modify the template to either reduce the strain on the capacity or let AWS select an available zone.

Option B, "Modify the AWS CloudFormation template to not specify an Availability Zone for the EC2 instance," is a valid solution. By removing the explicit Availability Zone specification, AWS will automatically choose an Availability Zone with sufficient capacity during instance launch. This allows AWS to find a zone where the requested instance type is available, circumventing the insufficient capacity error.

Option C, "Modify the AWS CloudFormation template to use a different EC2 instance type," is also a reasonable approach. Different instance types have varying resource requirements and are housed within different hardware pools. Selecting a different, potentially smaller or less popular, instance type may allow

the instance to launch successfully if the requested instance type is experiencing capacity constraints. For example, switching from m5.xlarge to m5.large might resolve the issue.

Option A is not an ideal solution. Creating a separate CloudFormation template doesn't inherently solve the insufficient capacity problem; it simply isolates the failing resource. The error will likely persist unless accompanied by other changes.

Option D, using a different AMI, is unlikely to solve the issue. AMIs primarily define the operating system and software, not the underlying hardware requirements that contribute to instance capacity issues.

Option E, using the AWS CLI's validate-template command, only checks the template for syntax and structural errors. It doesn't assess resource availability or capacity constraints. It is a good practice to validate the template to avoid syntax errors but will not address insufficient instance capacity.

In summary, the two most effective actions are to either let AWS choose the Availability Zone by removing explicit specification in the template (B), or to try using a different instance type (C) that might have available capacity.

Relevant Documentation:

AWS EC2 Instance Types:https://aws.amazon.com/ec2/instance-types/
AWS CloudFormation Get started:https://aws.amazon.com/cloudformation/getting-started/

Question: 46

A company hosts a web application on Amazon EC2 instances behind an Application Load Balancer (ALB). The company uses Amazon Route 53 to route traffic.
The company also has a static website that is configured in an Amazon S3 bucket.
A SysOps administrator must use the static website as a backup to the web application. The failover to the static website must be fully automated.
Which combination of actions will meet these requirements? (Choose two.)

A. Create a primary failover routing policy record. Configure the value to be the ALB.

B. Create an AWS Lambda function to switch from the primary website to the secondary website when the health check fails.

C. Create a primary failover routing policy record. Configure the value to be the ALB. Associate the record with a Route 53 health check.

D. Create a secondary failover routing policy record. Configure the value to be the static website. Associate the record with a Route 53 health check.

E. Create a secondary failover routing policy record. Configure the value to be the static website.

Answer: CE

Explanation:

The requirement is to automatically failover from a web application hosted on EC2 instances behind an ALB to a static website hosted in an S3 bucket when the primary application is unavailable. Route 53 is used for DNS resolution, and the failover needs to be automated.

Option C is correct because it establishes the primary routing mechanism using Route 53's failover routing policy. By creating a primary failover record pointing to the ALB and associating it with a health check, Route 53 can monitor the ALB's health. If the health check fails (meaning the application behind the ALB is unavailable), Route 53 will automatically stop routing traffic to the ALB. This sets up the initial "primary" destination.

Option E is also correct as it defines the fallback destination. Creating a secondary failover routing policy

record pointing to the S3 static website ensures that when the primary record (associated with the ALB) is deemed unhealthy, Route 53 will automatically route traffic to the static website. Associating this record with the same health check isn't necessary; in fact, it would be counterproductive. The secondary record activates when the primary health check fails. Without the secondary record, traffic would simply be dropped when the ALB is unhealthy, defeating the purpose of the failover.

Option A is incorrect because it only defines the primary record but doesn't associate it with a health check for automated failover. Without the health check, Route 53 would continue to route traffic to the ALB even if the underlying application is failing.

Option B is incorrect because using a Lambda function to switch DNS records is a more complex and less efficient solution compared to using Route 53's built-in failover routing policies. Route 53 failover is designed specifically for this purpose and offers better availability and lower latency. Additionally, managing a Lambda function adds operational overhead.

Option D is incorrect because, while it defines the secondary record and associates it with a health check, it does not define primary record which must be present to set the correct flow. The failover relies on the primary health check failing before the secondary record becomes active. If the secondary is independently evaluated by its own health check, that wouldn't achieve the desired failover behavior.

Therefore, the combination of options C and E correctly sets up a Route 53 failover configuration where traffic is initially routed to the web application behind the ALB, and if the health check associated with the ALB fails, traffic is automatically rerouted to the static website in S3.

Further reading:

Route 53 Failover:https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/dns-failover.html **Route 53 Health Checks:**https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/dns-failover-configuring-health-checks.html

Question: 47

A data analytics application is running on an Amazon EC2 instance. A SysOps administrator must add custom dimensions to the metrics collected by the Amazon
CloudWatch agent.
How can the SysOps administrator meet this requirement?

A. Create a custom shell script to extract the dimensions and collect the metrics using the Amazon CloudWatch agent.

B. Create an Amazon EventBridge (Amazon CloudWatch Events) rule to evaluate the required custom dimensions and send the metrics to Amazon Simple Notification Service (Amazon SNS).

C. Create an AWS Lambda function to collect the metrics from AWS CloudTrail and send the metrics to an Amazon CloudWatch Logs group.

D. Create an append_dimensions field in the Amazon CloudWatch agent configuration file to collect the metrics.

Answer: D

Explanation:

The correct answer is **D. Create an append_dimensions field in the Amazon CloudWatch agent configuration file to collect the metrics.**

Here's a detailed justification:

The Amazon CloudWatch agent is specifically designed to collect system metrics and logs from EC2

instances and on-premises servers. It allows for customization to collect specific metrics and add dimensions, which are key-value pairs that provide context and granularity to the collected data.

The append_dimensions field in the CloudWatch agent configuration file allows you to add custom dimensions to all metrics collected by the agent on that specific instance. This is the most straightforward and efficient way to fulfill the requirement of adding custom dimensions to existing metrics. The agent handles the metric collection and submission to CloudWatch after being configured.

Option A is incorrect because writing a custom shell script to extract dimensions and collect metrics would be a more complex and less maintainable solution. The CloudWatch agent is already designed to handle metric collection and provides the required configuration options.

Option B is incorrect because Amazon EventBridge is used for event routing and triggering actions based on events, not for collecting and processing metrics directly. Sending events to SNS would be an indirect and inefficient way to try and accomplish this goal.

Option C is incorrect because AWS Lambda functions, while versatile, are not the optimal solution for collecting system-level metrics. CloudTrail is an audit service for AWS API calls, not a source of granular system metrics. Furthermore, sending metrics to CloudWatch Logs for later extraction is less efficient than directly sending them as metrics using the CloudWatch agent.

The append_dimensions setting in the CloudWatch agent configuration directly addresses the problem, providing a simple and integrated way to add the required custom dimensions. It leverages the existing capabilities of the agent, making it the most efficient and correct approach.

Supporting Documentation:

Amazon CloudWatch Agent Configuration File: General Information:
https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/CloudWatch-Agent-Configuration-File-Sections.html
Specify Which Metrics to Collect:
https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/metrics-collected-by-CloudWatch-agent.html

Question: 48

A company stores its data in an Amazon S3 bucket. The company is required to classify the data and find any sensitive personal information in its S3 files.
Which solution will meet these requirements?

A. Create an AWS Config rule to discover sensitive personal information in the S3 files and mark them as noncompliant.

B. Create an S3 event-driven artificial intelligence/machine learning (AI/ML) pipeline to classify sensitive personal information by using Amazon Rekognition.

C. Enable Amazon GuardDuty. Configure S3 protection to monitor all data inside Amazon S3.

D. Enable Amazon Macie. Create a discovery job that uses the managed data identifier.

Answer: D

Explanation:

The correct answer is D, enabling Amazon Macie and creating a discovery job using a managed data identifier. Here's why:

Macie is designed specifically for discovering and protecting sensitive data stored in Amazon S3. It uses machine learning and pattern matching to identify personally identifiable information (PII), protected health

information (PHI), financial data, and other sensitive data types. This aligns perfectly with the requirement to classify data and find sensitive personal information within the S3 bucket.

Creating a discovery job in Macie allows you to scan the S3 bucket and identify these sensitive data elements. Managed data identifiers are pre-defined patterns and machine learning models that Macie uses to detect common sensitive data types. Macie then provides reports on the discovered sensitive data, including its location within the S3 bucket and the type of sensitive data identified.

Option A, using AWS Config, is primarily focused on configuration compliance and governance. While Config can monitor S3 bucket properties, it's not designed for deep content inspection and identification of sensitive data within files.

Option B, an S3 event-driven AI/ML pipeline using Amazon Rekognition, is not the right solution. Rekognition is designed for image and video analysis and not suited to identify personally sensitive information in data files.

Option C, enabling Amazon GuardDuty and configuring S3 protection, is for threat detection. While GuardDuty monitors S3 activity for malicious behavior, it doesn't classify data or identify sensitive information within the S3 files themselves.

Therefore, Amazon Macie is the only solution directly addressing the requirement to classify data and find sensitive personal information within S3 buckets. It offers pre-built classifiers and automated discovery capabilities that are specifically designed for this purpose.

Authoritative Links:

Amazon Macie:https://aws.amazon.com/macie/
AWS Config:https://aws.amazon.com/config/
Amazon GuardDuty:https://aws.amazon.com/guardduty/
Amazon Rekognition:https://aws.amazon.com/rekognition/

Question: 49

A company hosts a web portal on Amazon EC2 instances. The web portal uses an Elastic Load Balancer (ELB) and Amazon Route 53 for its public DNS service.
The ELB and the EC2 instances are deployed by way of a single AWS CloudFormation stack in the us-east-1 Region.
The web portal must be highly available across multiple Regions.
Which configuration will meet these requirements?

A. Deploy a copy of the stack in the us-west-2 Region. Create a single start of authority (SOA) record in Route 53 that includes the IP address from each ELB. Configure the SOA record with health checks. Use the ELB in us-east-1 as the primary record and the ELB in us-west-2 as the secondary record.

B. Deploy a copy of the stack in the us-west-2 Region. Create an additional A record in Route 53 that includes the ELB in us-west-2 as an alias target. Configure the A records with a failover routing policy and health checks. Use the ELB in us-east-1 as the primary record and the ELB in us-west-2 as the secondary record.

C. Deploy a new group of EC2 instances in the us-west-2 Region. Associate the new EC2 instances with the existing ELB, and configure load balancer health checks on all EC2 instances. Configure the ELB to update Route 53 when EC2 instances in us-west-2 fail health checks.

D. Deploy a new group of EC2 instances in the us-west-2 Region. Configure EC2 health checks on all EC2 instances in each Region. Configure a peering connection between the VPCs. Use the VPC in us-east-1 as the primary record and the VPC in us-west-2 as the secondary record.

Answer: B

Explanation:

The correct answer is B. Here's why:

Option B offers a multi-region active/passive setup using Route 53's failover routing policy, which directly addresses the requirement for high availability across multiple regions. Deploying a copy of the CloudFormation stack in us-west-2 ensures that you have a complete, independent environment in the second region.

Creating an additional A record in Route 53 with the us-west-2 ELB as an alias target alongside the original record for us-east-1 allows Route 53 to manage traffic distribution based on the health of the ELBs. Crucially, configuring a failover routing policy with health checks ensures that Route 53 will automatically redirect traffic to the us-west-2 ELB if the us-east-1 ELB becomes unhealthy. The us-east-1 ELB is designated as the primary record, while the us-west-2 ELB is the secondary (failover) record. This configuration adheres to a standard active/passive approach for disaster recovery.

Option A is incorrect because a Start of Authority (SOA) record is not used for routing traffic. SOA records contain administrative information about a DNS zone, not IP addresses for servers.

Option C is incorrect because an ELB is regional. You cannot associate EC2 instances from different regions with a single ELB. ELBs are designed to distribute traffic within a single region.

Option D is incorrect because VPC peering does not automatically provide failover capabilities for web traffic routing. Even with a peering connection, Route 53 would still need to be configured to direct traffic to the backup VPC. Using VPCs directly in Route53 is generally not ideal.

Refer to the AWS documentation for more information on Route 53 failover routing policies and multi-region architectures:

Route 53 Health Checks
Route 53 Failover Routing
Multi-Region Architectures

Question: 50

A SysOps administrator is investigating why a user has been unable to use RDP to connect over the internet from their home computer to a bastion server running on an Amazon EC2 Windows instance.
Which of the following are possible causes of this issue? (Choose two.)

A. A network ACL associated with the bastion's subnet is blocking the network traffic.
B. The instance does not have a private IP address.
C. The route table associated with the bastion's subnet does not have a route to the internet gateway.
D. The security group for the instance does not have an inbound rule on port 22.
E. The security group for the instance does not have an outbound rule on port 3389.

Answer: AC

Explanation:

Here's a detailed justification for why options A and C are the correct answers, along with supporting concepts and authoritative links:

Understanding the Scenario

The core problem is that a user can't connect to an EC2 Windows instance (bastion server) via RDP (port 3389) from their home computer (over the internet). This implies a network connectivity issue preventing traffic from reaching the instance.

Justification for Option A: Network ACL Blocking Traffic

Network Access Control Lists (NACLs) act as a stateless firewall at the subnet level. They control inbound and outbound traffic. If the NACL associated with the subnet where the bastion server resides does not have an inbound rule allowing traffic on port 3389 (RDP) from the user's home IP address range (or 0.0.0.0/0 for any source, which isn't ideal for security), the connection will be blocked. NACLs are evaluated before security groups, so even if the security group is configured correctly, the NACL can still prevent access.

Justification for Option C: Route Table Missing Route to Internet Gateway

For the EC2 instance to be accessible from the internet, the subnet it's in must have a route to an Internet Gateway. A route table dictates how network packets are routed from within the subnet. If the route table associated with the bastion server's subnet doesn't contain a route that directs traffic destined for the internet (typically 0.0.0.0/0) to an Internet Gateway, the instance effectively has no path to send traffic to or receive traffic from the internet, including RDP connection attempts.

Why other options are incorrect:

B: The instance does not have a private IP address. An EC2 instance must have a private IP address. It's fundamental for internal communication within the VPC. The instance may have a public IP address (or use an Elastic IP), which is important for direct internet access but not the core issue here.

D: The security group for the instance does not have an inbound rule on port 22. Port 22 is for SSH, not RDP. The question specifies RDP (port 3389), making this irrelevant.

E: The security group for the instance does not have an outbound rule on port 3389. While outbound rules can restrict traffic, typically security groups are configured to allow all outbound traffic (the default behavior). The primary concern is usually inbound traffic for incoming connections like RDP. Even if an outbound rule blocked 3389, the issue would likely be on the response from the server to the client, not preventing the initial connection attempt. The core issue is the inability to establish the connection to the server.

Authoritative Links:

Amazon VPC Network ACLs:https://docs.aws.amazon.com/vpc/latest/userguide/vpc-network-acls.html
Amazon VPC Route Tables:https://docs.aws.amazon.com/vpc/latest/userguide/VPC_Route_Tables.html
Amazon EC2 Security Groups:https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-security-groups.html

Question: 51

A SysOps administrator is examining the following AWS CloudFormation template:

```
AWSTemplateFormatVersion: '2010-09-09'
Description: 'Creates an EC2 Instance'
Resources:
    EC2Instance:
      Type: AWS::EC2::Instance
      Properties:
        ImageId: ami-79fd7eee
        InstanceType: m5n.large
        SubnetId: subnet-1abc3d3fg
        PrivateDnsName: ip-10-24-34-0.ec2.internal
        Tags:
          - Key: Name
            Value: !Sub "${AWS::StackName} Instance"
```

Why will the stack creation fail?

A. The Outputs section of the CloudFormation template was omitted. B. The Parameters section of the CloudFormation template was omitted. C. The PrivateDnsName cannot be set from a CloudFormation template. D. The VPC was not specified in the CloudFormation template.

Answer: C

Explanation:

PrivateDNSEndpoint is an output, not a parameter

Question: 52

A new application runs on Amazon EC2 instances and accesses data in an Amazon RDS database instance. When fully deployed in production, the application fails. The database can be queried from a console on a bastion host.
When looking at the web server logs, the following error is repeated multiple times: ***
Error Establishing a Database Connection
Which of the following may be causes of the connectivity problems? (Choose two.)

A. The security group for the database does not have the appropriate egress rule from the database to the web server.

B. The certificate used by the web server is not trusted by the RDS instance.

C. The security group for the database does not have the appropriate ingress rule from the web server to the database.

D. The port used by the application developer does not match the port specified in the RDS configuration. E. The database is still being created and is not available for connectivity.

Answer: CD

Explanation:

Here's a detailed justification for why options C and D are the correct answers, and why the others are incorrect, regarding the EC2 application's inability to connect to the RDS database:

Option C: The security group for the database does not have the appropriate ingress rule from the web server to the database.

This is a highly probable cause. Security groups act as virtual firewalls that control inbound and outbound traffic for your EC2 instances and RDS databases. For the web servers to connect to the RDS database, the database's security group must allow ingress (inbound) traffic from the web servers, specifically on the database port (typically 3306 for MySQL or MariaDB, 5432 for PostgreSQL, etc.). Without this ingress rule, the database will reject connection attempts from the web servers, resulting in the "Error Establishing a Database Connection" message. The fact that the database can be queried from the bastion host indicates that the database instance itself is functioning and accessible, but the security group configuration is blocking the web servers.

Option D: The port used by the application developer does not match the port specified in the RDS configuration.

This is also a valid cause. The application needs to connect to the database on the correct port number. If the application is configured to use a different port than the one the RDS database is listening on, the connection will fail. For example, if the RDS database is configured to listen on port 3306 and the application is trying to connect to port 3307, the connection will fail. Misconfiguration on the application side regarding the port can prevent it from connecting even if security groups are configured correctly.

Why the other options are incorrect:

Option A: The security group for the database does not have the appropriate egress rule from the database to the web server. Egress rules control outbound traffic. While important for other communication scenarios, they are not the primary cause of the "Error Establishing a Database Connection" from the web servers to the database. The web server initiates the connection, so the ingress rule on the database matters the most.

Option B: The certificate used by the web server is not trusted by the RDS instance. Certificates are primarily used for SSL/TLS encryption to secure the connection. While certificate issues can prevent a connection, they typically manifest as SSL-related errors rather than a general "Error Establishing a Database Connection." The question doesn't specify that SSL/TLS is enabled or that any certificate-related errors are occurring.

Option E: The database is still being created and is not available for connectivity. The question states that the database can be queried from the bastion host. This implies that the database creation has completed, and the database is available to accept connections. Therefore, this is not a valid root cause.

Authoritative Links:

AWS Security Groups:https://docs.aws.amazon.com/vpc/latest/userguide/VPC_SecurityGroups.html
Connecting to an RDS instance:
https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_ConnectToInstance.html

Question: 53

A compliance team requires all administrator passwords for Amazon RDS DB instances to be changed at least annually. Which solution meets this requirement in the MOST operationally efficient manner?

A. Store the database credentials in AWS Secrets Manager. Configure automatic rotation for the secret every 365 days.

B. Store the database credentials as a parameter in the RDS parameter group. Create a database trigger to rotate the password every 365 days.

C. Store the database credentials in a private Amazon S3 bucket. Schedule an AWS Lambda function to generate a new set of credentials every 365 days.

D. Store the database credentials in AWS Systems Manager Parameter Store as a secure string parameter. Configure automatic rotation for the parameter every 365 days.

Answer: A

Explanation:

The correct answer is A. Here's why:

Detailed Justification:

Option A leverages AWS Secrets Manager's automatic rotation capability. AWS Secrets Manager is designed specifically for managing secrets (like database credentials), rotating them, and managing access. The automatic rotation feature allows you to set a schedule (in this case, every 365 days) for the password to be automatically changed. This involves writing a Lambda function that Secrets Manager invokes at the specified intervals to generate a new password, update the database, and update the secret in Secrets Manager. This is the most operationally efficient method because it automates the entire process, removing the need for manual intervention or custom scripting to handle the rotation.

Option B is incorrect because storing database credentials in an RDS parameter group is not a secure practice. Parameter groups are meant for configuration parameters, not sensitive information. Furthermore, using a database trigger for password rotation is complex, potentially error-prone, and tightly couples the security logic to the database itself. Also, parameter groups do not have automated rotation features.

Option C is suboptimal because storing credentials in a private S3 bucket requires you to build and manage the password rotation mechanism entirely from scratch using Lambda. While feasible, it's more work than leveraging the built-in functionality of Secrets Manager, increasing the operational burden. It lacks the built-in auditability and integration with other AWS services that Secrets Manager provides.

Option D is not the best solution. While AWS Systems Manager Parameter Store secure strings can store secrets, it does not have native automatic rotation capabilities like Secrets Manager. While you can implement rotation using custom scripts and scheduled events, this adds unnecessary complexity compared to the out-of-the-box automation provided by Secrets Manager. Parameter Store is generally designed for configuration data and not specifically optimized for frequent secret rotation, auditing and access control in the same way as Secrets Manager.

Therefore, AWS Secrets Manager (Option A) provides the most operationally efficient and secure method for automatically rotating RDS database administrator passwords annually. It reduces manual effort, minimizes the risk of human error, and aligns with AWS best practices for secret management.

Authoritative Links:

AWS Secrets Manager:https://aws.amazon.com/secrets-manager/
Rotating AWS Secrets Manager secrets:
https://docs.aws.amazon.com/secretsmanager/latest/userguide/rotating-secrets.html

Question: 54

A SysOps administrator is responsible for managing a fleet of Amazon EC2 instances. These EC2 instances upload build artifacts to a third-party service. The third-party service recently implemented a strict IP allow list that requires all build uploads to come from a single IP address.
What change should the systems administrator make to the existing build fleet to comply with this new requirement?

A. Move all of the EC2 instances behind a NAT gateway and provide the gateway IP address to the service.

B. Move all of the EC2 instances behind an internet gateway and provide the gateway IP address to the service.

C. Move all of the EC2 instances into a single Availability Zone and provide the Availability Zone IP address to the service.

D. Move all of the EC2 instances to a peered VPC and provide the VPC IP address to the service.

Answer: A

Explanation:

The correct answer is **A. Move all of the EC2 instances behind a NAT gateway and provide the gateway IP address to the service.**

Here's why:

The problem requires all build uploads to originate from a single IP address for the third-party service's IP allow list. EC2 instances in a VPC, by default, have dynamic public IP addresses (if assigned) or no public IP addresses if launched without. To comply with the IP allow list requirement, a static and consistent IP address is needed.

A NAT (Network Address Translation) gateway allows instances in a private subnet to connect to services outside the VPC (like the third-party service) without exposing their individual public IP addresses. All outbound traffic from the EC2 instances passes through the NAT gateway, which translates the private IP addresses of the instances to the NAT gateway's public IP address. This single, public IP address is then used for all communication with the external service.

Therefore, by placing the EC2 instances behind a NAT gateway, the SysOps administrator can provide the NAT gateway's public IP address (or Elastic IP associated with it) to the third-party service. This satisfies the requirement of having all build uploads originate from a single, known IP address.

Option B is incorrect because an Internet Gateway allows direct internet access for instances. While it enables communication, it doesn't provide a single, consistent source IP for all instances, defeating the purpose of the allow list.

Option C is incorrect because moving instances to a single Availability Zone does not guarantee a single public IP address for all outbound traffic. Each instance within the Availability Zone would still need its own public IP or a NAT instance. Also, it does not provide IP address of AZ but it provides better network latency between instances

Option D is incorrect because VPC peering is used to connect two VPCs, and it does not solve the problem of providing a single IP address for outbound traffic to the third-party service. VPC peering manages internal network connection within AWS environment.

Here are some authoritative links for further research:

NAT Gateway:https://docs.aws.amazon.com/vpc/latest/userguide/vpc-nat-gateway.html
Internet Gateway:https://docs.aws.amazon.com/vpc/latest/userguide/VPC_Internet_Gateway.html **VPC Peering:**https://docs.aws.amazon.com/vpc/latest/peering/what-is-vpc-peering.html

Question: 55

A company uses an Amazon CloudFront distribution to deliver its website. Traffic logs for the website must be centrally stored, and all data must be encrypted at rest.
Which solution will meet these requirements?

A. Create an Amazon OpenSearch Service (Amazon Elasticsearch Service) domain with internet access and server-side encryption that uses the default AWS managed customer master key (CMK). Configure CloudFront to use the Amazon OpenSearch Service (Amazon Elasticsearch Service) domain as a log destination.

B. Create an Amazon OpenSearch Service (Amazon Elasticsearch Service) domain with VPC access and server-side encryption that uses AES-256. Configure CloudFront to use the Amazon OpenSearch Service (Amazon Elasticsearch Service) domain as a log destination.

C. Create an Amazon S3 bucket that is configured with default server-side encryption that uses AES-256. Configure CloudFront to use the S3 bucket as a log destination.

D. Create an Amazon S3 bucket that is configured with no default encryption. Enable encryption in the CloudFront distribution, and use the S3 bucket as a log destination.

Answer: C

Explanation:

The correct answer is C because it directly addresses the requirements of central storage and encryption at rest for CloudFront access logs in the most straightforward and cost-effective manner.

Here's a breakdown:

Requirement 1: Centrally Stored Traffic Logs: Amazon S3 is a highly scalable, durable, and cost-effective object storage service ideal for storing large volumes of data, such as website traffic logs. CloudFront can be configured to directly write logs to an S3 bucket, satisfying the central storage requirement.

Requirement 2: Data Encryption at Rest: S3 offers server-side encryption (SSE) with AES-256 as a configuration option. By enabling default server-side encryption with AES-256 on the S3 bucket, all logs stored within the bucket are automatically encrypted at rest, meeting the encryption requirement.

Let's analyze why the other options are less suitable:

Option A (OpenSearch with Internet Access): While OpenSearch can store and analyze logs, exposing it directly to the internet poses security risks. Furthermore, it adds unnecessary complexity and cost for simply storing the raw log data. The question requires storing, not analysis.

Option B (OpenSearch with VPC Access): While more secure than Option A, using OpenSearch within a VPC is still an overkill solution for simply storing logs. It involves more infrastructure and configuration complexity than simply using S3.

Option D (CloudFront Encryption): CloudFront does not encrypt access logs written to the S3 bucket itself. CloudFront encryption relates to content delivered to the end-users, not the log files. Not enabling server-side encryption on the S3 bucket directly violates the encryption requirement.

Therefore, configuring CloudFront to write logs to an S3 bucket with default server-side encryption (SSE) using AES-256 is the most efficient and appropriate solution to satisfy the prompt's requirements.

Relevant Links:

Amazon S3 Server-Side Encryption:https://docs.aws.amazon.com/AmazonS3/latest/userguide/serv-side-encryption.html
CloudFront Access Logs:
https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/AccessLogs.html

Question: 56

An organization created an Amazon Elastic File System (Amazon EFS) volume with a file system ID of fs-85ba41fc, and it is actively used by 10 Amazon EC2 hosts. The organization has become concerned that the file system is not encrypted. How can this be resolved?

A. Enable encryption on each host's connection to the Amazon EFS volume. Each connection must be recreated

for encryption to take effect.

B. Enable encryption on the existing EFS volume by using the AWS Command Line Interface.

C. Enable encryption on each host's local drive. Restart each host to encrypt the drive.

D. Enable encryption on a newly created volume and copy all data from the original volume. Reconnect each host to the new volume.

Answer: D

Explanation:

The correct answer is D because encrypting an existing EFS volume at rest is not directly supported. EFS encryption at rest is a configuration that must be specified during volume creation. You cannot simply enable it on an already existing EFS file system.

Option A is incorrect. While EFS supports encryption in transit, which encrypts data as it moves between the EC2 instances and the EFS volume, this is separate from encryption at rest. Enabling encryption in transit doesn't address the organization's concern about the file system itself not being encrypted when stored. This option also does not mention encryption at rest.

Option B is incorrect because there is no command or method to enable encryption at rest on an existing EFS volume. EFS encryption at rest must be configured during file system creation. Changing the encryption status afterward isn't supported by AWS.

Option C is incorrect because encrypting the EC2 host's local drives is irrelevant to encrypting the EFS volume. The EFS volume is a network file system, and its encryption status is independent of the encryption status of the clients accessing it. Encrypting local drives helps with the local security of the instances, not the data stored on the shared EFS volume.

Therefore, the only way to resolve the organization's concern about the unencrypted EFS volume is to create a new EFS volume with encryption at rest enabled. Once the new volume is created, the data from the original, unencrypted volume needs to be copied over. Finally, the EC2 instances must be reconfigured to mount the new, encrypted EFS volume instead of the old one. This ensures the data is encrypted at rest going forward.

Here are some authoritative links for further research:

EFS Encryption at Rest and in Transit
Creating EFS File Systems

Question: 57

A company uses an AWS Service Catalog portfolio to create and manage resources. A SysOps administrator must create a replica of the company's existing AWS infrastructure in a new AWS account.
What is the MOST operationally efficient way to meet this requirement?

A. Create an AWS CloudFormation template to use the AWS Service Catalog portfolio in the new AWS account.

B. In the new AWS account, manually create an AWS Service Catalog portfolio that duplicates the original portfolio.

C. Run an AWS Lambda function to create a new AWS Service Catalog portfolio based on the output of the DescribePortfolio API operation.

D. Share the AWS Service Catalog portfolio with the new AWS account. Import the portfolio into the new AWS account.

Answer: D

Explanation:

The most operationally efficient way to replicate an AWS Service Catalog portfolio in a new AWS account is option D: Share the AWS Service Catalog portfolio with the new AWS account and then import it.

Here's why:

AWS Service Catalog Portfolio Sharing: AWS Service Catalog supports sharing portfolios across AWS accounts using AWS Resource Access Manager (RAM). This is specifically designed for this type of scenario. Rather than recreating the portfolio and all its products, you can directly leverage the existing configuration. **Efficiency:** This approach avoids manual recreation or complex scripting. Sharing a portfolio directly transfers all the defined products, constraints, and tags to the new account, saving considerable time and effort. It also reduces the risk of errors associated with manual configuration.

Consistency: Sharing ensures a consistent configuration between the original and the new account. This is critical for maintaining a standardized environment and avoiding discrepancies.

Avoidance of Duplication: Manual creation (option B) and programmatic creation (option C) lead to duplication of resources and potential inconsistencies.

Reduced Complexity: CloudFormation (option A) could be used, but it would necessitate a thorough understanding of the original portfolio's configuration and would be significantly more complex than the direct sharing approach.

After sharing, importing the portfolio within the new AWS account makes it accessible and usable. This whole process ensures an operationally streamlined process.

Authoritative Links:

AWS Service Catalog Portfolio Sharing
AWS Resource Access Manager (RAM)

Question: 58

A SysOps administrator must manage the security of an AWS account. Recently, an IAM user's access key was mistakenly uploaded to a public code repository.
The SysOps administrator must identify anything that was changed by using this access key. How should the SysOps administrator meet these requirements?

A. Create an Amazon EventBridge (Amazon CloudWatch Events) rule to send all IAM events to an AWS Lambda function for analysis.

B. Query Amazon EC2 logs by using Amazon CloudWatch Logs Insights for all events initiated with the compromised access key within the suspected timeframe.

C. Search AWS CloudTrail event history for all events initiated with the compromised access key within the suspected timeframe.

D. Search VPC Flow Logs for all events initiated with the compromised access key within the suspected timeframe.

Answer: C

Explanation:

The correct answer is C: Search AWS CloudTrail event history for all events initiated with the compromised access key within the suspected timeframe.

Here's a detailed justification:

AWS CloudTrail is a service that enables governance, compliance, operational auditing, and risk auditing of your AWS account. CloudTrail logs API calls made to AWS services. This means it captures who made the API call, from where (source IP), when, and what action was performed. In this scenario, the SysOps administrator needs to identify all actions performed using the compromised access key. CloudTrail is perfectly suited for

this purpose because it records every API call along with the associated access key used to make that call. By filtering the CloudTrail event history by the compromised access key ID and the timeframe during which it was exposed, the administrator can obtain a complete audit trail of all actions performed using that key.

Option A is incorrect because while EventBridge and Lambda can be used for real-time monitoring of IAM events, they don't provide a historical record of all API calls made. Setting up an EventBridge rule now wouldn't help in identifying what happened before the rule was created. EventBridge is more for proactive monitoring and automation based on events occurring now or in the future.

Option B is incorrect because Amazon EC2 logs, analyzed through CloudWatch Logs Insights, primarily capture instance-level logs, such as application logs or system logs. EC2 logs generally do not include the specific IAM access key used to make API calls to AWS services. While certain actions inside an EC2 instance might generate logs showing API activity, it's not the central and comprehensive record like CloudTrail, and relies on detailed logging configuration within the instance itself.

Option D is incorrect because VPC Flow Logs capture information about the IP traffic going to and from network interfaces in your VPC. While VPC Flow Logs can help identify suspicious network activity that resulted from actions performed with the compromised key (like an EC2 instance initiating connections to a malicious IP), they won't directly tell you which IAM key was used to initiate the action that caused the traffic. VPC Flow Logs offer network-level visibility, not API call-level auditing.

Therefore, CloudTrail is the only service that provides the required visibility into all API calls made with a specific access key, making option C the correct solution.

Relevant Links:

AWS CloudTrail Documentation:https://docs.aws.amazon.com/awscloudtrail/latest/userguide/cloudtrail-user-guide.html
Amazon EventBridge Documentation:https://docs.aws.amazon.com/eventbridge/latest/userguide/eb-what-is.html
Amazon CloudWatch Logs Insights Documentation:
https://docs.aws.amazon.com/AmazonCloudWatch/latest/logs/AnalyzingLogData.html
VPC Flow Logs Documentation:https://docs.aws.amazon.com/vpc/latest/userguide/flow-logs.html

Question: 59

A company runs a retail website on multiple Amazon EC2 instances behind an Application Load Balancer (ALB). The company must secure traffic to the website over an HTTPS connection.
Which combination of actions should a SysOps administrator take to meet these requirements? (Choose two.)

A. Attach the certificate to each EC2 instance.
B. Attach the certificate to the ALB.
C. Create a private certificate in AWS Certificate Manager (ACM).
D. Create a public certificate in AWS Certificate Manager (ACM).
E. Export the certificate, and attach it to the website.

Answer: BD

Explanation:

The correct answer is BD. Here's a detailed justification:

To secure traffic to a website over HTTPS, a Secure Sockets Layer/Transport Layer Security (SSL/TLS) certificate is required. This certificate verifies the identity of the website and encrypts communication between the client (browser) and the server. An Application Load Balancer (ALB) can handle the SSL/TLS

termination, meaning it decrypts the traffic before forwarding it to the backend EC2 instances. This simplifies management and improves performance.

Option D is correct because a public certificate is needed to establish trust with clients connecting to the website. AWS Certificate Manager (ACM) can be used to provision, manage, and deploy SSL/TLS certificates for use with AWS services, including ALBs. A public certificate from ACM is automatically trusted by browsers.

Option B is correct because the SSL/TLS certificate needs to be associated with the ALB. The ALB is the entry point for incoming HTTPS traffic, and it needs the certificate to decrypt the traffic and establish a secure connection. By attaching the certificate to the ALB, you delegate the SSL/TLS termination to the load balancer, reducing the workload on the EC2 instances.

Option A is incorrect because attaching the certificate to each EC2 instance is unnecessary and less efficient when using an ALB for SSL/TLS termination. This would require managing certificates on each instance, increasing administrative overhead.

Option C is incorrect because a private certificate is typically used for internal communication or services within a VPC that do not need to be publicly trusted. For a public website, a publicly trusted certificate is necessary.

Option E is incorrect because exporting the certificate and attaching it to the website directly (bypassing the ALB) defeats the purpose of using the ALB for SSL/TLS termination. Furthermore, directly managing the certificate on the web server would increase operational complexity. The ALB simplifies certificate management.

In summary, a public certificate from ACM is created (D), and then this certificate is attached to the ALB (B) to enable HTTPS and secure traffic to the website, taking advantage of the ALB's SSL/TLS termination capabilities.

Supporting Links:

AWS Certificate Manager: https://aws.amazon.com/certificate-manager/
Application Load Balancer Listener Configuration:
https://docs.aws.amazon.com/elasticloadbalancing/latest/application/load-balancer-listeners.html

Question: 60

A company runs a web application on three Amazon EC2 instances behind an Application Load Balancer (ALB). The company notices that random periods of increased traffic cause a degradation in the application's performance. A SysOps administrator must scale the application to meet the increased traffic.
Which solution meets these requirements?

A. Create an Amazon CloudWatch alarm to monitor application latency and increase the size of each EC2 instance if the desired threshold is reached.

B. Create an Amazon EventBridge (Amazon CloudWatch Events) rule to monitor application latency and add an EC2 instance to the ALB if the desired threshold is reached.

C. Deploy the application to an Auto Scaling group of EC2 instances with a target tracking scaling policy. Attach the ALB to the Auto Scaling group.

D. Deploy the application to an Auto Scaling group of EC2 instances with a scheduled scaling policy. Attach the ALB to the Auto Scaling group.

Answer: C

Explanation:

The correct answer is C because it provides a dynamic and automated scaling solution that directly addresses the fluctuating traffic patterns. Let's break down why:

Auto Scaling Group (ASG): An ASG automatically manages the desired capacity of EC2 instances. This ensures that the application has the necessary resources available based on demand.
https://docs.aws.amazon.com/autoscaling/ec2/userguide/auto-scaling-groups.html

Target Tracking Scaling Policy: This type of scaling policy monitors a specified metric (e.g., average CPU utilization of the EC2 instances) and automatically adjusts the number of instances in the ASG to maintain the target value. This reactive scaling is perfect for unpredictable traffic spikes.
https://docs.aws.amazon.com/autoscaling/ec2/userguide/as-scaling-target-tracking.html

Application Load Balancer (ALB): The ALB distributes incoming traffic evenly across the healthy EC2 instances within the ASG. This ensures high availability and efficient resource utilization.
https://docs.aws.amazon.com/elasticloadbalancing/latest/application/introduction.html

Option A is incorrect because vertically scaling instances (changing their size) is disruptive and takes time. It's also less efficient than scaling horizontally (adding more instances). CloudWatch Alarms can trigger actions, but resizing instances isn't the best scaling strategy in this scenario.

Option B is not optimal because EventBridge is typically used for event-driven architectures and may not be the best choice for directly reacting to real-time application latency. Manually adding instances via EventBridge triggered actions would be complex and less automated than using an ASG.

Option D is also not a great fit because scheduled scaling is better suited for predictable traffic patterns. The problem description specifically mentions "random periods of increased traffic," making a scheduled approach inappropriate. Scheduled Scaling is useful if you know beforehand when the traffic would increase (eg: everyday at 9am).

Question: 61

A company has a stateful, long-running workload on a single xlarge general purpose Amazon EC2 On-Demand Instance
Metrics show that the service is always using 80% of its available memory and 40% of its available CPU.
A SysOps administrator must reduce the cost of the service without negatively affecting performance. Which change in instance type will meet these requirements?

A. Change to one large compute optimized On-Demand Instance.
B. Change to one large memory optimized On-Demand Instance.
C. Change to one xlarge general purpose Spot Instance.
D. Change to two large general purpose On-Demand Instances.

Answer: B

Explanation:

The problem states the existing EC2 instance is memory-constrained, utilizing 80% of its available memory, while CPU utilization is only at 40%. The goal is to reduce cost without impacting performance.

Option A, a compute-optimized instance, addresses high CPU usage, which isn't the bottleneck here.
Therefore, this would be ineffective and potentially wasteful.

Option B, a memory-optimized instance, directly addresses the memory bottleneck. Switching to a large memory-optimized instance provides more RAM while remaining an On-Demand Instance, ensuring consistent performance. Sizing it to "large" allows for potentially saving money while adequately addressing memory needs.

Option C, a Spot Instance, aims to reduce cost, but the workload is stateful and long-running. Spot Instances are prone to interruption, potentially disrupting the application, which is against the requirements. Moreover, this does not address the memory bottleneck directly.

Option D, two large general purpose instances, distributes the load across instances. While it could alleviate the memory pressure on each individual instance, it also complicates the architecture and doubles the instance count, potentially increasing costs and complexity.

Therefore, option B is the most appropriate solution. By selecting a memory-optimized instance type, the application receives the necessary memory resources to function optimally. Remaining with On-Demand instances ensures that the workload will not be interrupted. Using a "large" instance allows for potential cost savings as compared to "xlarge".

Relevant link for further research:

Amazon EC2 Instance Types
Amazon EC2 Pricing

Question: 62

A company asks a SysOps administrator to ensure that AWS CloudTrail files are not tampered with after they are created. Currently, the company uses AWS
Identity and Access Management (IAM) to restrict access to specific trails. The company's security team needs the ability to trace the integrity of each file.
What is the MOST operationally efficient solution that meets these requirements?

A. Create an Amazon EventBridge (Amazon CloudWatch Events) rule that invokes an AWS Lambda function when a new file is delivered. Configure the Lambda function to compute an MD5 hash check on the file and store the result in an Amazon DynamoDB table. The security team can use the values that are stored in DynamoDB to verify the integrity of the delivered files.

B. Create an AWS Lambda function that is invoked each time a new file is delivered to the CloudTrail bucket. Configure the Lambda function to compute an MD5 hash check on the file and store the result as a tag in an Amazon 53 object. The security team can use the information in the tag to verify the integrity of the delivered files.

C. Enable the CloudTrail file integrity feature on an Amazon S3 bucket. Create an IAM policy that grants the security team access to the file integrity logs that are stored in the S3 bucket.

D. Enable the CloudTrail file integrity feature on the trail. The security team can use the digest file that is created by CloudTrail to verify the integrity of the delivered files.

Answer: D

Explanation:

The correct answer is **D**, enabling the CloudTrail file integrity feature on the trail. This is the most operationally efficient solution because it leverages a built-in AWS feature designed specifically for this purpose. CloudTrail's file integrity validation (FIV) helps detect unintentional or malicious modifications to CloudTrail log files.

When FIV is enabled, CloudTrail creates digest files containing hash values for the log files. These digest files are also signed digitally, providing an auditable chain of evidence. The digest files are stored in the same S3 bucket as the CloudTrail logs, making it easy to locate and access them. The security team can then use these digest files to verify the integrity of the log files using tools provided by AWS or third-party libraries.

Options A and B, while technically feasible, involve creating and managing Lambda functions, DynamoDB tables, and potentially managing tags on S3 objects. These options add unnecessary complexity and operational overhead compared to using the built-in FIV feature. They require writing and maintaining code, handling potential errors, and managing resource scaling.

Option C mentions enabling file integrity on an S3 bucket, which is not a direct feature in AWS. File integrity validation is a feature of CloudTrail, not S3.

Therefore, utilizing CloudTrail's built-in file integrity validation feature is the most straightforward and efficient way to meet the requirements of ensuring CloudTrail file integrity and providing the security team with the means to verify it. This approach minimizes operational burden and maximizes the use of AWS-managed services.

Relevant documentation:

Working with CloudTrail Log File Integrity Validation

Question: 63

A SysOps Administrator is required to monitor free space on Amazon EBS volumes attached to Microsoft Windows-based Amazon EC2 instances within a company's account. The administrator must be alerted to potential issues.
What should the administrator do to receive email alerts before low storage space affects EC2 instance performance?

A. Use built-in Amazon CloudWatch metrics, and configure CloudWatch alarms and an Amazon SNS topic for email notifications.

B. Use AWS CloudTrail logs and configure the trail to send notifications to an Amazon SNS topic.

C. Use the Amazon CloudWatch agent to send disk space metrics, then set up CloudWatch alarms using an Amazon SNS topic.

D. Use AWS Trusted Advisor and enable email notification alerts for EC2 disk space.

Answer: C

Explanation:

The correct answer is C because it outlines the appropriate steps for monitoring disk space on Windows-based EC2 instances and triggering email alerts.

Here's a detailed justification:

Built-in CloudWatch metrics are insufficient for granular disk monitoring on Windows: While CloudWatch provides basic EC2 instance metrics, it doesn't natively offer detailed disk space utilization metrics for guest operating systems like Windows. We need finer-grained information such as the percentage of free space on specific volumes.

CloudWatch agent for custom metrics: The CloudWatch agent allows you to collect custom metrics from EC2 instances. Specifically, it can gather disk space usage data within the Windows OS. The agent can be configured to monitor specific drives and the available free space.

CloudWatch Alarms for threshold breaches: Once the CloudWatch agent is installed and reporting disk space metrics, we can create CloudWatch Alarms. These alarms are configured to trigger when the disk space falls below a predefined threshold (e.g., 10% free space).

SNS for Email Notifications: CloudWatch alarms can be configured to send notifications via Amazon SNS (Simple Notification Service). By configuring an SNS topic with email subscriptions, alerts are sent directly to the administrator's email address when the alarm state changes (e.g., goes into an "Alarm" state due to low disk space).

Here's why the other options are incorrect:

A: Using built-in CloudWatch metrics: As mentioned, built-in metrics lack the necessary disk space granularity for Windows instances.
B: AWS CloudTrail: CloudTrail primarily tracks API calls and user activity, not system resource utilization like

disk space. It's designed for auditing and security monitoring, not performance alerting.

D: AWS Trusted Advisor: Trusted Advisor checks for cost optimization, security, fault tolerance, and performance improvements. While it can offer recommendations related to EC2 instance configuration, it's not designed for real-time disk space monitoring and alerting. It's also important to note that disk utilization checks are not part of the Trusted Advisor checks by default.

Authoritative Links:

CloudWatch Agent:https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/Install-CloudWatch-Agent.html
CloudWatch Alarms:
https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/AlarmThatSendsEmail.html
Amazon SNS:https://aws.amazon.com/sns/

Question: 64

A company is using an AWS KMS customer master key (CMK) with imported key material. The company references the CMK by its alias in the Java application to encrypt data. The CMK must be rotated every 6 months.
What is the process to rotate the key?

A. Enable automatic key rotation for the CMK, and specify a period of 6 months.
B. Create a new CMK with new imported material, and update the key alias to point to the new CMK.
C. Delete the current key material, and import new material into the existing CMK.
D. Import a copy of the existing key material into a new CMK as a backup, and set the rotation schedule for 6 months.

Answer: B

Explanation:

The correct answer is B: Create a new CMK with new imported material, and update the key alias to point to the new CMK. Here's why:

Automatic key rotation is not supported for CMKs with imported key material. AWS KMS allows automatic key rotation only for CMKs that are created and managed by AWS KMS itself. Since the company is using imported key material, automatic rotation (option A) is not a viable solution.
https://docs.aws.amazon.com/kms/latest/developerguide/rotate-keys.html

Deleting the current key material and importing new material into the existing CMK (option C) would render any data encrypted with the previous key material inaccessible. Key rotation is intended to protect data from compromise, but this method would make the previously encrypted data unusable, which defeats the purpose and introduces data loss risk.

Importing a copy of the existing key material into a new CMK and setting a rotation schedule (option D) is incorrect because it doesn't actually rotate the key material being used for encryption. It merely creates a backup of the same key material, and automatic rotation is not possible for imported keys.

To properly rotate a CMK with imported key material, the process involves creating a new CMK and importing new key material into it. Then, critically, the key alias used by the application needs to be updated to point to this new CMK. The application, which is currently using the alias to encrypt data, will then automatically begin using the new CMK without any code changes. This is a seamless process. After validating the new CMK is functioning correctly, the older CMK can eventually be disabled and deleted to minimize confusion and potential misuse. This approach fulfills the company's requirement for key rotation every 6 months without causing data loss or requiring application code modifications. The alias acts as an abstraction layer, decoupling the application from the specific CMK ID.

Question: 65

The security team is concerned because the number of AWS Identity and Access Management (IAM) policies being used in the environment is increasing. The team tasked a SysOps administrator to report on the current number of IAM policies in use and the total available IAM policies.
Which AWS service should the administrator use to check how current IAM policy usage compares to current service limits?

A. AWS Trusted Advisor
B. Amazon Inspector
C. AWS Config
D. AWS Organizations

Answer: A

Explanation:

The correct answer is **A. AWS Trusted Advisor**.

AWS Trusted Advisor is the best tool for this task because it provides insights into your AWS environment based on best practices. Specifically, it includes checks on service limits, security, cost optimization, fault tolerance, and performance. In the context of IAM policies, Trusted Advisor monitors the utilization of IAM resources against AWS service limits. It identifies if you are approaching or exceeding the maximum allowed number of IAM policies, roles, users, groups, and instance profiles. This is precisely what the security team requires – a comparison between the current IAM policy usage and the total available IAM policies, allowing them to proactively manage IAM policy growth.

Amazon Inspector (B) is a vulnerability management service that automates security vulnerability assessments. While useful for identifying security issues within applications and infrastructure, it doesn't directly address IAM policy usage against service limits. AWS Config (C) assesses, audits, and evaluates the configurations of your AWS resources. It provides detailed configuration information and supports compliance audits, but it doesn't directly report on service limit usage in the same way as Trusted Advisor. AWS Organizations (D) is used for managing multiple AWS accounts from a central location. Although
Organizations can help with governance and compliance across accounts, it does not directly report on IAM policy usage relative to service limits within an individual account. Trusted Advisor's dedicated checks for service limits make it the most suitable tool for this specific request. The Trusted Advisor provides the necessary information to keep the security team informed about the growth of IAM Policies and compare them to current service limits.

AWS Trusted Advisor DocumentationAWS IAM Limits

Question: 66

A SysOps administrator is trying to set up an Amazon Route 53 domain name to route traffic to a website hosted on Amazon S3. The domain name of the website is www.example.com and the S3 bucket name DOC-EXAMPLE-BUCKET. After the record set is set up in Route 53, the domain name www.anycompany.com does not seem to work, and the static website is not displayed in the browser.
Which of the following is a cause of this?

A. The S3 bucket must be configured with Amazon CloudFront first.
B. The Route 53 record set must have an IAM role that allows access to the S3 bucket.
C. The Route 53 record set must be in the same region as the S3 bucket.

D. The S3 bucket name must match the record set name in Route 53.

Answer: D

Explanation:

The correct answer is **D. The S3 bucket name must match the record set name in Route 53.**

Here's a detailed justification:

When hosting a static website directly on Amazon S3 and using Amazon Route 53 to route traffic to it, a specific naming convention is crucial. Route 53 needs to know where to direct incoming traffic. To achieve this, the S3 bucket name **must** exactly match the domain or subdomain name you're using in your Route 53 record. In the scenario, the domain being used is www.example.com. Therefore, your S3 bucket MUST be named www.example.com.

If the S3 bucket name is different (e.g., DOC-EXAMPLE-BUCKET as in the question), Route 53 won't be able to correctly map the domain name to the S3 bucket endpoint, resulting in the website being inaccessible. Route 53 uses the bucket name to construct the correct S3 website endpoint.

Option A is incorrect because using CloudFront is an optimization strategy, not a requirement for basic S3 static website hosting with Route 53. CloudFront improves performance and caching but isn't mandatory for the initial setup.

Option B is incorrect because IAM roles govern access to AWS resources for entities like EC2 instances or Lambda functions, not directly for Route 53 record sets. Route 53 primarily deals with DNS resolution.

Option C is incorrect because Route 53 is a global service and doesn't need to be in the same region as the S3 bucket. S3 buckets, however, are region-specific. Route 53 directs users to the globally available endpoint for the S3 website, and the S3 service handles redirection within its infrastructure. The region of S3 is not considered when pointing a domain to a static S3 website.

Therefore, the most direct and critical reason for the website failing to load is the bucket name mismatch with the domain name in Route 53.

Relevant Links:

Hosting a Static Website on Amazon S3
Configuring a Static Website Using a Custom Domain Registered With Route 53

Question: 67

A SysOps administrator has used AWS CloudFormation to deploy a serverless application into a production VPC. The application consists of an AWS Lambda function, an Amazon DynamoDB table, and an Amazon API Gateway API. The SysOps administrator must delete the AWS CloudFormation stack without deleting the DynamoDB table. Which action should the SysOps administrator take before deleting the AWS CloudFormation stack?

A. Add a Retain deletion policy to the DynamoDB resource in the AWS CloudFormation stack.
B. Add a Snapshot deletion policy to the DynamoDB resource in the AWS CloudFormation stack.
C. Enable termination protection on the AWS CloudFormation stack.
D. Update the application's IAM policy with a Deny statement for the dynamodb:DeleteTable action.

Answer: A

Explanation:

The correct answer is A: Add a Retain deletion policy to the DynamoDB resource in the AWS CloudFormation stack.

When deleting a CloudFormation stack, CloudFormation attempts to delete all resources created by the stack. To prevent the deletion of a specific resource like a DynamoDB table, a deletion policy can be applied.
The Retain deletion policy instructs CloudFormation to retain the resource even when the stack is deleted.
This is crucial in scenarios where data persistence is required beyond the lifecycle of the CloudFormation stack.

Option B, adding a Snapshot deletion policy, applies to resources like EBS volumes. While it preserves data, it does so by creating a snapshot and then deleting the original resource. This doesn't achieve the goal of retaining the existing DynamoDB table.

Option C, enabling termination protection on the CloudFormation stack, prevents the entire stack from being accidentally deleted. While useful, it doesn't allow for selective deletion or retention of individual resources within the stack. It would prevent the stack from being deleted at all, which is not the desired outcome.

Option D, updating the application's IAM policy with a Deny statement for the dynamodb:DeleteTable action, prevents the application itself (i.e., the Lambda function) from deleting the DynamoDB table. However, it doesn't prevent CloudFormation from attempting to delete the table during stack deletion if no deletion policy is specified on the resource within the stack. The CloudFormation service principal, which performs the deletion, likely has sufficient privileges to override any application-level IAM policies.

Therefore, using the Retain deletion policy on the DynamoDB resource in the CloudFormation template ensures the table persists even after the CloudFormation stack is deleted, achieving the desired outcome without affecting the functionality of the application itself.

Refer to the AWS documentation for more information:

CloudFormation DeletionPolicy Attribute

Question: 68

A SysOps administrator is notified that an Amazon EC2 instance has stopped responding. The AWS Management Console indicates that the system checks are failing.
What should the administrator do first to resolve this issue?

A. Reboot the EC2 instance so it can be launched on a new host.
B. Stop and then start the EC2 instance so that it can be launched on a new host.
C. Terminate the EC2 instance and relaunch it.
D. View the AWS CloudTrail log to investigate what changed on the EC2 instance.

Answer: B

Explanation:

The correct answer is **B. Stop and then start the EC2 instance so that it can be launched on a new host.**

Here's a detailed justification:

When system checks are failing on an EC2 instance, it indicates a problem with the underlying host or the instance's hardware. System checks monitor the health of the EC2 hypervisor, network, and physical host. A failing system check suggests that the instance is unable to communicate with the network or that there's an issue with the host hardware.

Why not reboot? (Option A) A simple reboot might temporarily resolve software-related issues within the

instance, but if the problem is with the underlying host, a reboot will not solve the root cause, and the issue will likely reoccur after the instance reboots on the same problematic host. A reboot also maintains the instance on the same host, which is the source of the problem.

Why stop and start? (Option B) Stopping and starting an EC2 instance (that is not EBS-backed root volume) forces the instance to migrate to a new, potentially healthy, host. This is because when an instance is stopped, the underlying physical host is released, and upon starting, AWS will allocate the instance to a new host. This provides a chance for the instance to run on healthy infrastructure. This approach doesn't lose the underlying data if EBS is used.

Why not terminate and relaunch? (Option C) Terminating the instance would result in the loss of any data stored on the instance store (ephemeral) volumes. While a valid option if the data is backed up elsewhere, it is a more drastic step than stopping and starting and should be considered only if other options fail or if the instance's data is already persisted elsewhere, such as to EBS, S3 or other persistent data stores. In most cases, stopping and starting is preferred.

Why not immediately check CloudTrail? (Option D) While reviewing CloudTrail logs is important for investigating the root cause of an issue in the long run, it's not the first action to take when an instance is unresponsive and system checks are failing. The immediate priority is to restore service availability. CloudTrail analysis is more suited for post-incident analysis to prevent recurrence.

Therefore, stopping and starting the instance is the quickest and least disruptive approach to potentially resolve the problem by moving it to a healthy host. Following recovery, CloudTrail can be consulted for root cause analysis.

Supporting Documentation:

EC2 Instance System Checks
Troubleshooting Instances with Failed Status Checks

Question: 69

A software development company has multiple developers who work on the same product. Each developer must have their own development environments, and these development environments must be identical. Each development environment consists of Amazon EC2 instances and an Amazon RDS DB instance. The development environments should be created only when necessary, and they must be terminated each night to minimize costs. What is the MOST operationally efficient solution that meets these requirements?

A. Provide developers with access to the same AWS CloudFormation template so that they can provision their development environment when necessary. Schedule a nightly cron job on each development instance to stop all running processes to reduce CPU utilization to nearly zero.

B. Provide developers with access to the same AWS CloudFormation template so that they can provision their development environment when necessary. Schedule a nightly Amazon EventBridge (Amazon CloudWatch Events) rule to invoke an AWS Lambda function to delete the AWS CloudFormation stacks.

C. Provide developers with CLI commands so that they can provision their own development environment when necessary. Schedule a nightly Amazon EventBridge (Amazon CloudWatch Events) rule to invoke an AWS Lambda function to terminate all EC2 instances and the DB instance.

D. Provide developers with CLI commands so that they can provision their own development environment when necessary. Schedule a nightly Amazon EventBridge (Amazon CloudWatch Events) rule to cause AWS CloudFormation to delete all of the development environment resources.

Answer: B

Explanation:

The correct answer is B because it offers the most operationally efficient solution for managing development environments. CloudFormation templates enable developers to consistently provision identical environments, reducing configuration drift and potential errors. By using CloudFormation, the infrastructure is defined as code, promoting reproducibility and version control.

Scheduling an EventBridge rule to trigger a Lambda function to delete the CloudFormation stacks is efficient for nightly cleanup. Deleting the entire stack ensures that all resources, including EC2 instances and RDS instances, are properly terminated, preventing unnecessary costs. This approach avoids leaving dangling resources, which can happen if individual EC2 or RDS instances are terminated manually or through other means. Managing the infrastructure as a stack through CloudFormation provides a single point of
management for creation and deletion.

Option A is less efficient because stopping processes doesn't eliminate the cost of running EC2 and RDS instances. You still pay for the resources allocated to the stopped instances.

Option C is problematic because terminating individual EC2 and RDS instances outside of CloudFormation can lead to inconsistencies if some resources fail to terminate properly. This can create orphaned resources and complicate management. CLI commands, while workable, don't enforce the uniformity of the template.

Option D is similar to option C in that it relies on CloudFormation deleting individual resources. It's better to delete the entire stack as in Option B, because it's a more atomic operation and reduces chances of leaving resources behind. Furthermore, the question specifies using EventBridge to cause CloudFormation to delete the resources, which isn't the primary way EventBridge is used to interact with CloudFormation. EventBridge is better suited to invoke a Lambda function that then manages the CloudFormation stack lifecycle.

Therefore, CloudFormation with EventBridge and Lambda for stack deletion offers the best combination of consistency, cost optimization, and operational efficiency.

Relevant links for further research:

AWS CloudFormation:https://aws.amazon.com/cloudformation/
Amazon EventBridge:https://aws.amazon.com/eventbridge/ **AWS Lambda:**https://aws.amazon.com/lambda/

Question: 70

A company is partnering with an external vendor to provide data processing services. For this integration, the vendor must host the company's data in an Amazon
S3 bucket in the vendor's AWS account. The vendor is allowing the company to provide an AWS Key Management Service (AWS KMS) key to encrypt the company's data. The vendor has provided an IAM role Amazon Resources Name (ARN) to the company for this integration.
What should a SysOps administrator do to configure this integration?

A. Create a new KMS key. Add the vendor's IAM role ARN to the KMS key policy. Provide the new KMS key ARN to the vendor.

B. Create a new KMS key. Create a new IAM key. Add the vendor's IAM role ARN to an inline policy that is attached to the IAM user. Provide the new IAM user ARN to the vendor.

C. Configure encryption using the KMS managed S3 key. Add the vendor's IAM role ARN to the KMS key policy. Provide the KMS managed S3 key ARN to the vendor.

D. Configure encryption using the KMS managed S3 key. Create an S3 bucket. Add the vendor's IAM role ARN to the S3 bucket policy. Provide the S3 bucket ARN to the vendor.

Answer: A

Explanation:

The correct answer is **A: Create a new KMS key. Add the vendor's IAM role ARN to the KMS key policy. Provide the new KMS key ARN to the vendor.**

Here's a detailed justification:

1. **Control over Encryption:** The company wants to use its own KMS key to encrypt data hosted in the

vendor's S3 bucket. This ensures the company retains control over the encryption and decryption of its data. Creating a new KMS key specifically for this integration is the best practice.

2. **Granting Vendor Access:** The vendor needs permission to use the KMS key to encrypt and decrypt the data. This is achieved by adding the vendor's IAM role ARN to the KMS key policy. The KMS key policy acts as a resource-based policy that grants permissions to specific AWS principals (in this case, the vendor's IAM role) to use the key.
3. **Minimum Required Permissions:** By granting the vendor access through the KMS key policy, you are adhering to the principle of least privilege. The vendor will only be able to use the KMS key for encrypting and decrypting data, and nothing else.
4. **IAM Role for Vendor Access:** The vendor's IAM role is designed to be assumed by the vendor's services. The vendor uses this role to gain necessary permissions. The company doesn't need to create new IAM users or keys for the vendor's services.
5. **Avoiding S3 Bucket Policies for KMS Access:** Option D is incorrect because while S3 bucket policies control access to the S3 bucket itself, they don't directly grant access to the KMS key used for encryption. The KMS key policy governs who can use the key.
6. **Why not KMS Managed key:** KMS managed keys (option C & D) do not allow customer control, which is an important requirement in this question.
7. **No need for IAM User creation:** Option B is incorrect because there is no need to create an IAM user. The vendor already has an IAM role. Creating a new IAM user would introduce complexity.
8. **Sharing the KMS Key ARN:** The vendor needs to know which KMS key to use. Providing the KMS key ARN allows them to specify this key when uploading data to the S3 bucket.
9. **Security Best Practices:** Using a dedicated KMS key for this integration allows for better auditing and simplifies key rotation if needed.
10. **Cloud Computing and Security:** This solution aligns with cloud computing security best practices, emphasizing the importance of encryption, identity and access management (IAM), and the principle of least privilege.

Authoritative links:

AWS KMS documentation: https://docs.aws.amazon.com/kms/latest/developerguide/overview.html IAM Roles: https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles.html
S3 Encryption: https://docs.aws.amazon.com/AmazonS3/latest/userguide/UsingEncryption.html Key Policies: https://docs.aws.amazon.com/kms/latest/developerguide/key-policies.html

Question: 71

A SysOps administrator is using AWS Systems Manager Patch Manager to patch a fleet of Amazon EC2 instances. The SysOps administrator has configured a patch baseline and a maintenance window. The SysOps administrator also has used an instance tag to identify which instances to patch.
The SysOps administrator must give Systems Manager the ability to access the EC2 instances.
Which additional action must the SysOps administrator perform to meet this requirement?

A. Add an inbound rule to the instances' security group.
B. Attach an IAM instance profile with access to Systems Manager to the instances.
C. Create a Systems Manager activation. Then activate the fleet of instances.
D. Manually specify the instances to patch instead of using tag-based selection.

Answer: B

Explanation:

The correct answer is B: Attach an IAM instance profile with access to Systems Manager to the instances.

Here's a detailed justification:

AWS Systems Manager (SSM) needs permissions to perform actions on your EC2 instances, such as patching them. This requires providing the instances with a role that grants these permissions. An IAM instance profile is the standard and recommended way to grant these permissions to EC2 instances. The instance profile is essentially a container for an IAM role that can be assumed by EC2 instances.

Option A is incorrect because adding an inbound rule to the instance's security group only allows network traffic to reach the instance. While necessary for connectivity, it does not grant SSM the permissions it needs to manage the instances. SSM uses the EC2's outbound connectivity to communicate with the SSM service endpoints.

Option C is incorrect because Systems Manager activations are typically used for hybrid environments (non-EC2 instances running on-premises or in other clouds). For EC2 instances, the IAM role approach is the appropriate and simpler method. Activations register a machine as a managed instance, but for EC2 instances, this is automatically handled if the instance has the correct IAM role.

Option D is incorrect because tag-based selection is a perfectly valid and preferred way to target instances for patching. The problem isn't the use of tags; it's the lack of permissions for SSM to act on the instances identified by the tag.

Therefore, to give Systems Manager the ability to access and patch the EC2 instances identified by tags, you need to attach an IAM instance profile with the necessary permissions (specifically, the AmazonSSMManagedInstanceCore policy or a similar custom policy granting SSM access) to the instances. This allows the SSM agent running on the instances to authenticate with the SSM service and perform the patching operations as defined in the maintenance window and patch baseline.

Authoritative Links:

IAM roles for Amazon EC2:https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/iam-roles-for-amazon-ec2.html
AWS Systems Manager Prerequisites:https://docs.aws.amazon.com/systems-manager/latest/userguide/systems-manager-prereqs.html
Grant Systems Manager Access to EC2 Instances:https://docs.aws.amazon.com/systems-manager/latest/userguide/security-iam-id-based-policy-examples.html#instance-management-policy-examples

Question: 72

A company hosts its website on Amazon EC2 instances in the us-east-1 Region. The company is preparing to extend its website into the eu-central-1 Region, but the database must remain only in us-east-1. After deployment, the EC2 instances in eu-central-1 are unable to connect to the database in us-east-1.
What is the MOST operationally efficient solution that will resolve this connectivity issue?

A. Create a VPC peering connection between the two Regions. Add the private IP address range of the instances to the inbound rule of the database security group.

B. Create a VPC peering connection between the two Regions. Add the security group of the instances in eu-central-1 to the outbound rule of the database security group.

C. Create a VPN connection between the two Regions. Add the private IP address range of the instances to the outbound rule of the database security group.

D. Create a VPN connection between the two Regions. Add the security group of the instances in eu-central-1 to the inbound rule of the database security group.

Answer: A

Explanation:

Here's a detailed justification for why option A is the most operationally efficient solution, along with

supporting concepts and links:

The problem is that EC2 instances in eu-central-1 cannot connect to the database in us-east-1. The most direct and efficient solution for enabling private network communication between two VPCs in different AWS Regions is to use VPC peering. VPC peering allows network traffic to be routed between two VPCs using private IP addresses.

Option A correctly utilizes VPC peering. Creating a VPC peering connection establishes a private network link between the two Regions' VPCs, allowing traffic to flow directly between them without traversing the public internet. Adding the private IP address range of the EC2 instances in eu-central-1 to the inbound rule of the database security group in us-east-1 specifically allows traffic from those instances to access the database. This acts as a firewall rule, allowing only authorized traffic to reach the database.

Option B is incorrect because you need to add an inbound rule to the database's security group to allow connections to the database from the EC2 instances. Outbound rules control what the database can connect to, not what can connect to the database.

Options C and D propose using a VPN connection. While a VPN would work, it is less operationally efficient than VPC peering for this specific scenario. VPN connections require setting up and managing VPN gateways, which adds complexity and overhead. VPC peering is a fully managed AWS service designed for this purpose. VPN also adds encryption overhead that is not required for peered VPCs.Adding the private IP address range of the instances is better than adding security group to the database sg as it is more explicit and restrictive allowing us to better manage the database.

In summary, VPC peering provides a simple, direct, and managed solution for connecting VPCs across Regions, making it the most operationally efficient choice. Adjusting the security group rules ensures only the intended instances can access the database, maintaining security.

Supporting Concepts:

VPC Peering: Enables you to connect one VPC with another via a direct network route using private IP addresses. It allows instances in either VPC to communicate with each other as if they are within the same network.

Security Groups: Virtual firewalls that control inbound and outbound traffic for EC2 instances. They operate at the instance level and define rules specifying allowed protocols, ports, and source/destination IP addresses or security groups.

Inbound Rules: Govern what traffic is allowed into the instance (or database, in this case).

Outbound Rules: Govern what traffic is allowed out of the instance.

Authoritative Links:

VPC Peering:https://docs.aws.amazon.com/vpc/latest/peering/what-is-vpc-peering.html
Security Groups:https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-security-groups.html

Question: 73

A company wants to create an automated solution for all accounts managed by AWS Organizations to detect any security groups that use 0.0.0.0/0 as the source address for inbound traffic. The company also wants to automatically remediate any noncompliant security groups by restricting access to a specific CIDR block that corresponds with the company's intranet.
Which set of actions should the SysOps administrator take to create a solution?

A. Create an AWS Config rule to detect noncompliant security groups. Set up automatic remediation to change the 0.0.0.0/0 source address to the approved CIDR block.

B. Create an IAM policy to deny the creation of security groups that have 0.0.0.0/0 as the source address. Attach this IAM policy to every user in the company.

C. Create an AWS Lambda function to inspect new and existing security groups. Check for a noncompliant 0.0.0.0/0 source address and change the source address to the approved CIDR block.

D. Create a service control policy (SCP) for the organizational unit (OU) to deny the creation of security groups that have the 0.0.0.0/0 source address. Set up automatic remediation to change the 0.0.0.0/0 source address to the approved CIDR block.

Answer: A

Explanation:

The correct answer is **A**. Here's why:

AWS Config: AWS Config is a service that enables you to assess, audit, and evaluate the configurations of your AWS resources. It continuously monitors and records resource configurations and allows you to automate the evaluation of recorded configurations against desired configurations. This makes it ideal for detecting security groups that violate the rule of not using 0.0.0.0/0 for inbound traffic.
https://aws.amazon.com/config/

AWS Config Rules: You can define AWS Config rules to specify the desired configurations for your AWS resources. In this case, you would create a rule to check if any security group's inbound rules have 0.0.0.0/0 as a source.

Automatic Remediation: AWS Config supports automatic remediation, allowing you to automatically take corrective actions when a resource is found to be noncompliant with a rule. You can configure the Config rule to automatically change the source address of noncompliant security groups to the company's intranet CIDR block.

Why other options are incorrect:

B (IAM Policy): While IAM policies can prevent the creation of resources with certain configurations, they don't help remediate existing non-compliant resources. Also, attaching an IAM policy to every user is not scalable in the long run compared to applying a SCP to prevent this at the organization level.

C (Lambda Function): A Lambda function could accomplish this task, but it would require more custom coding and management compared to using AWS Config. AWS Config provides built-in functionality for configuration monitoring and remediation. While this can be a viable option, it introduces additional operational overhead and maintenance requirements.

D (SCP): SCPs, like IAM policies, can prevent the creation of non-compliant resources but don't remediate existing ones. Also, while SCPs are applied at the OU or Organization level, they prevent the creation of non-compliant resources and do not automatically remediate those. Configuring remediation within an SCP is not standard or supported.

In summary, AWS Config provides a managed and scalable solution for detecting and automatically remediating non-compliant security groups, making option A the most appropriate choice.

Question: 74

A company requires that all activity in its AWS account be logged using AWS CloudTrail. Additionally, a SysOps administrator must know when CloudTrail log files are modified or deleted.
How should the SysOps administrator meet these requirements?

A. Enable log file integrity validation. Use the AWS CLI to validate the log files.

B. Enable log file integrity validation. Use the AWS CloudTrail Processing Library to validate the log files.

C. Use CloudTrail Insights to monitor the log files for modifications.

D. Use Amazon CloudWatch Logs to monitor the log files for modifications.

Answer: A

Explanation:

The correct answer is **A. Enable log file integrity validation. Use the AWS CLI to validate the log files.**

Here's why:

Requirement 1: Logging all activity: AWS CloudTrail is the service that logs API calls made within the AWS account. Enabling CloudTrail ensures this.

Requirement 2: Detecting modifications or deletions: Log file integrity validation is a crucial feature of CloudTrail that addresses this requirement. When enabled, CloudTrail creates a digital signature of each log file using a cryptographic hash function. This signature is then recorded in a digest file. Any modification or deletion of the log file will cause the validation process to fail because the generated hash will not match the signature in the digest file.

Why CLI is the correct validation tool: The AWS Command Line Interface (CLI) provides the necessary tools to validate the integrity of the CloudTrail log files using the information provided in the digest files. The CLI tools can calculate the hash of the log files and compare it with the value recorded in the digest file. If the hashes match, it confirms the log file's integrity.

Why other options are incorrect:

B: AWS CloudTrail Processing Library: While the CloudTrail Processing Library can process and analyze CloudTrail logs, it's not specifically designed for validating log file integrity. Its main purpose is to parse and process the log data for further analysis.

C: CloudTrail Insights: CloudTrail Insights is used to identify unusual operational activity in your AWS account by analyzing CloudTrail events. It doesn't directly validate the integrity of the log files. It detects anomalies based on API call patterns, not alterations to log files themselves.

D: Amazon CloudWatch Logs: While CloudWatch Logs can monitor log files for specific patterns, it's not designed to validate the integrity of the log files. CloudWatch Logs focus on detecting and alerting on specific events within the logs, not whether the logs have been tampered with.

In summary: Log file integrity validation coupled with the AWS CLI for validation provides a direct and effective way to fulfill both logging and tamper-detection requirements for CloudTrail logs.

Authoritative Links:

AWS CloudTrail Log File Integrity Validation: https://docs.aws.amazon.com/awscloudtrail/latest/userguide/cloudtrail-log-file-validation-cli.html Validating CloudTrail Log File Integrity: https://aws.amazon.com/blogs/security/validating-cloudtrail-log-file-integrity/

Question: 75

A company is planning to host its stateful web-based applications on AWS. A SysOps administrator is using an Auto Scaling group of Amazon EC2 instances. The web applications will run 24 hours a day, 7 days a week throughout the year. The company must be able to change the instance type within the same instance family later in the year based on the traffic and usage patterns.

Which EC2 instance purchasing option will meet these requirements MOST cost-effectively?

A. Convertible Reserved Instances
B. On-Demand Instances
C. Spot Instances
D. Standard Reserved Instances

Answer: A

Explanation:

The question requires identifying the most cost-effective EC2 purchasing option for stateful, 24/7 web applications in an Auto Scaling group, with the ability to change instance types within the same family later on.

Option A, Convertible Reserved Instances, is the most suitable choice. Reserved Instances (RIs) offer a significant discount compared to On-Demand instances for long-term usage. Convertible RIs specifically allow changing instance attributes like instance type, OS, and tenancy within the same instance family. This flexibility addresses the requirement to modify the instance type based on future traffic patterns while still benefiting from the cost savings of RIs.

Option B, On-Demand Instances, provides flexibility but lacks cost optimization for consistent 24/7 workloads. The higher price negates cost-effectiveness.

Option C, Spot Instances, offers significant discounts but is unsuitable for stateful applications due to potential interruptions. Spot Instances are reclaimed by AWS when the spot price exceeds the bid price, which can lead to data loss and application downtime, violating the requirement for constant availability.

Option D, Standard Reserved Instances, offers cost savings for long-term use but lacks the flexibility to change instance types. While cheaper than Convertible RIs initially, they lock the user into the same instance type for the duration of the term, failing to meet the changing instance type requirement.

Therefore, Convertible Reserved Instances strike the optimal balance between cost savings for consistent usage and the flexibility needed to adapt to future traffic changes, making them the most cost-effective choice for the scenario.

Further reading:

AWS EC2 Reserved Instances:https://aws.amazon.com/ec2/pricing/reserved-instances/ **AWS EC2 Instance Purchasing Options:**https://aws.amazon.com/ec2/purchasing-options/

Question: 76

An application runs on Amazon EC2 instances in an Auto Scaling group. Following the deployment of a new feature on the EC2 instances, some instances were marked as unhealthy and then replaced by the Auto Scaling group. The EC2 instances terminated before a SysOps administrator could determine the cause of the health status changes.
To troubleshoot this issue, the SysOps administrator wants to ensure that an AWS Lambda function is invoked in this situation.
How should the SysOps administrator meet these requirements?

A. Activate the instance scale-in protection setting for the Auto Scaling group. Invoke the Lambda function through Amazon EventBridge (Amazon CloudWatch Events).

B. Activate the instance scale-in protection setting for the Auto Scaling group. Invoke the Lambda function through Amazon Route 53.

C. Add a lifecycle hook to the Auto Scaling group to invoke the Lambda function through Amazon EventBridge (Amazon CloudWatch Events).

D. Add a lifecycle hook to the Auto Scaling group to invoke the Lambda function through Amazon Route 53.

Answer: C

Explanation:

Here's a detailed justification for why option C is the correct answer and why the other options are incorrect:

Correct Answer: C. Add a lifecycle hook to the Auto Scaling group to invoke the Lambda function through Amazon EventBridge (Amazon CloudWatch Events).

Justification:

1. **Lifecycle Hooks:** Auto Scaling lifecycle hooks allow you to perform custom actions whenever an instance transitions to a specific state during its lifecycle (launching or terminating). In this scenario, we need to intercept the termination process of the unhealthy instances. Lifecycle hooks are designed precisely for this purpose. A lifecycle hook can be configured to pause the instance termination while a specific action, like invoking a Lambda function, is executed.

2. **EventBridge (Amazon CloudWatch Events):** EventBridge is a serverless event bus service that enables you to route events between AWS services, integrated SaaS applications, and your own applications. It's the natural mechanism for Auto Scaling lifecycle hooks to trigger other AWS services. When the lifecycle hook activates, it can send an event to EventBridge. This event triggers a configured rule that then invokes the Lambda function.

3. **Lambda Function Invocation:** The Lambda function receives the event data from EventBridge, which includes information about the terminating instance. This enables the Lambda function to perform actions such as collecting logs, capturing instance details, or creating a snapshot before the instance is fully terminated.

4. **Meeting the Requirements:** Using lifecycle hooks and EventBridge ensures that the Lambda function is reliably invoked during instance termination, allowing the SysOps administrator to investigate the cause of the health status changes.

Why other options are incorrect:

A & B: Instance Scale-In Protection: While instance scale-in protection prevents instances from being terminated during scale-in events, it does not provide a mechanism to trigger a Lambda function or perform custom actions during termination. It only prevents termination altogether. This does not address the requirement of troubleshooting during termination.

https://docs.aws.amazon.com/autoscaling/ec2/userguide/ec2-auto-scaling-instance-protection.html **B & D: Amazon Route 53:** Route 53 is a DNS service. It is used to direct traffic to EC2 instances but is not involved in the lifecycle management or termination process of instances. Using Route 53 would be completely irrelevant to triggering a Lambda function during instance termination. There is no integration to allow Route 53 to be used for triggering actions during instance termination.

In summary, using a lifecycle hook with EventBridge provides the necessary functionality to trigger a Lambda function during the termination of EC2 instances in an Auto Scaling group, fulfilling the requirements for troubleshooting health status changes.

Authoritative Links:

Auto Scaling Lifecycle Hooks:https://docs.aws.amazon.com/autoscaling/ec2/userguide/lifecycle-hooks.html **Amazon EventBridge:**https://aws.amazon.com/eventbridge/

Question: 77

A company runs an application that hosts critical data for several clients. The company uses AWS CloudTrail to track user activities on various AWS resources. To meet new security requirements, the company needs to protect the CloudTrail log files from being modified, deleted, or forged.
Which solution will meet these requirement?

A. Enable CloudTrail log file integrity validation.
B. Use Amazon S3 MFA Delete on the S3 bucket where the CloudTrail log files are stored.
C. Use Amazon S3 Versioning to keep all versions of the CloudTrail log files.
D. Use AWS Key Management Service (AWS KMS) security keys to secure the CloudTrail log files.

Answer: A

Explanation:

The correct answer is **A. Enable CloudTrail log file integrity validation.**

Here's why:

CloudTrail Log File Integrity Validation is specifically designed to protect CloudTrail logs from unauthorized modification or deletion. When enabled, CloudTrail creates digitally signed digests of the log files. These digests are stored separately and can be used to verify the integrity of the log files. Any tampering with the log files will result in a failed integrity validation, immediately alerting the company to potential security breaches or malicious activity. This validation relies on cryptographic hashing to detect alterations.

Why other options are incorrect:

B. Use Amazon S3 MFA Delete on the S3 bucket where the CloudTrail log files are stored: MFA Delete adds an extra layer of security to S3 bucket operations by requiring multi-factor authentication for certain sensitive actions, like permanently deleting a version of an object. While helpful, it doesn't prevent modification of existing log files or alert to tampering. It only makes permanent deletion harder.

C. Use Amazon S3 Versioning to keep all versions of the CloudTrail log files: S3 Versioning is a good practice for data protection as it preserves all versions of an object when modified or deleted. However, by itself, versioning doesn't guarantee integrity validation. A malicious actor could potentially modify a file and create a new version without detection.

D. Use AWS Key Management Service (AWS KMS) security keys to secure the CloudTrail log files: KMS is vital for encrypting CloudTrail logs, ensuring confidentiality. While encryption helps protect the content of the logs, it doesn't inherently validate their integrity. A malicious actor could potentially modify the encrypted log files, and while it would be garbled without the key, the modification itself wouldn't be readily detected. KMS plays a critical role in conjunction with integrity validation, but it does not fulfill the requirement alone. Integrity validation uses KMS encryption for the hash which is used to validate logs.

CloudTrail Log File Integrity Validation provides a verifiable way to ensure that log data hasn't been compromised. It's a critical security control in environments where log integrity is paramount.

Authoritative Links:

AWS CloudTrail Log File Integrity Validation:
https://docs.aws.amazon.com/awscloudtrail/latest/userguide/cloudtrail-log-file-validation-intro.html **Amazon S3 MFA Delete:**
https://docs.aws.amazon.com/AmazonS3/latest/userguide/MultiFactorAuthenticationDelete.html
Amazon S3 Versioning:https://docs.aws.amazon.com/AmazonS3/latest/userguide/versioning-overview.html **AWS Key Management Service (AWS KMS):**https://aws.amazon.com/kms/

Question: 78

A global company operates out of five AWS Regions. A SysOps administrator wants to identify all the company's tagged and untagged Amazon EC2 instances.
The company requires the output to display the instance ID and tags.
What is the MOST operationally efficient way for the SysOps administrator to meet these requirements?

A. Create a tag-based resource group in AWS Resource Groups.

B. Use AWS Trusted Advisor. Export the EC2 On-Demand Instances check results from Trusted Advisor.

C. Use Cost Explorer. Choose a service type of EC2-Instances, and group by Resource.

D. Use Tag Editor in AWS Resource Groups. Select all Regions, and choose a resource type of AWS::EC2::Instance.

Answer: D

Explanation:

The correct answer is **D: Use Tag Editor in AWS Resource Groups. Select all Regions, and choose a resource type of AWS::EC2::Instance.** Here's why:

Tag Editor, a feature within AWS Resource Groups, is designed specifically for managing and identifying resources based on their tags across multiple AWS Regions. It provides a centralized console to search for resources that either have or do not have specific tags. By selecting all regions and the AWS::EC2::Instance resource type, the administrator can effectively retrieve a comprehensive list of all EC2 instances, along with their tags, in a single operation. This fulfills the requirement of identifying both tagged and untagged EC2 instances and displaying the instance ID and tags.

Option A (creating a tag-based resource group) isn't suitable as its primary function is grouping resources based on tags for management purposes, not necessarily identifying all instances and their tag status, particularly untagged ones, across multiple regions.

Option B (AWS Trusted Advisor) focuses on optimization, security, and fault tolerance. While Trusted Advisor can flag untagged resources as a best practice check (particularly in cost optimization checks related to EC2 On-Demand Instances), it's not its primary purpose, and extracting the data and filtering to all instances isn't as efficient as using Tag Editor. Furthermore, it lacks the direct presentation of instance ID and tags in a consolidated view for all instances.

Option C (Cost Explorer) is primarily used for cost analysis. Grouping by resource helps in understanding costs associated with different EC2 instances, but it does not efficiently provide information about the presence or absence of tags, and does not give the direct output the question requires.

Therefore, Tag Editor is the most operationally efficient choice because it's designed for tag management across regions and resource types, enabling the administrator to quickly identify all EC2 instances and their tags in a single, streamlined process.

AWS Resource Groups DocumentationTag Editor Documentation

Question: 79

A company needs to upload gigabytes of files every day. The company need to achieve higher throughput and upload speeds to Amazon S3.
Which action should a SysOps administrator take to meet this requirement?

A. Create an Amazon CloudFront distribution with the GET HTTP method allowed and the S3 bucket as an origin.

B. Create an Amazon ElastiCache cluster and enable caching for the S3 bucket.

C. Set up AWS Global Accelerator and configure it with the S3 bucket.

D. Enable S3 Transfer Acceleration and use the acceleration endpoint when uploading files.

Answer: D

Explanation:

The correct answer is **D. Enable S3 Transfer Acceleration and use the acceleration endpoint when uploading files.**

S3 Transfer Acceleration leverages Amazon CloudFront's globally distributed edge locations to speed up data transfers into S3. When enabled, data is routed to the nearest edge location, which then optimizes the transfer over the AWS network backbone to the S3 bucket. This bypasses the public internet for a significant portion of the journey, reducing latency and increasing throughput, especially for transfers across long distances or from geographically dispersed locations.

Option A is incorrect because CloudFront is primarily for content delivery (downloads), not uploads. While you can configure CloudFront to allow PUT requests to an origin, it's not the primary mechanism for accelerated uploads.

Option B is incorrect because ElastiCache is for caching frequently accessed data, not for accelerating uploads to S3. It's a read-heavy optimization and wouldn't directly improve the throughput of large file uploads.

Option C is incorrect because AWS Global Accelerator improves performance for applications with users distributed globally by routing traffic to optimal endpoints. While it could theoretically help, S3 Transfer Acceleration is a purpose-built solution directly targeting S3 upload speeds, making it the superior and more efficient choice. Global Accelerator addresses a broader range of application performance issues and isn't specifically designed to accelerate S3 uploads like S3 Transfer Acceleration.

In summary, S3 Transfer Acceleration is the most effective and direct approach to achieving higher throughput and upload speeds to S3 when transferring gigabytes of files daily, leveraging the AWS global network for optimized transfer.

Further reading:

S3 Transfer Acceleration Documentation
AWS Global Accelerator Documentation

Question: 80

A SysOps administrator maintains the security and compliance of a company's AWS account. To ensure the company's Amazon EC2 instances are following company policy, a SysOps administrator wants to terminate any EC2 instance that do not contain a department tag. Noncompliant resources must be terminated in near-real time. Which solution will meet these requirements?

A. Create an AWS Config rule with the required-tags managed rule to identify noncompliant resources. Configure automatic remediation to run the AWS- TerminateEC2Instance automation document to terminate noncompliant resources.

B. Create a new Amazon EventBridge (Amazon CloudWatch Events) rule to monitor when new EC2 instances are created. Send the event to a Simple Notification Service (Amazon SNS) topic for automatic remediation.

C. Ensure all users who can create EC2 instances also have the permissions to use the ec2:CreateTags and ec2:DescribeTags actions. Change the instance's shutdown behavior to terminate.

D. Ensure AWS Systems Manager Compliance is configured to manage the EC2 instances. Call the AWS-StopEC2Instances automation document to stop noncompliant resources.

Answer: A

Explanation:

Here's a detailed justification for why option A is the best solution and why the others are less suitable for addressing the requirement of terminating EC2 instances lacking a department tag in near real-time while maintaining security and compliance:

Option A: AWS Config Rule with Automatic Remediation (Correct)

This option directly addresses the requirement through a proactive and automated approach. AWS Config allows you to define rules that evaluate the configuration of your AWS resources. The required-tags managed rule specifically checks for the presence of specific tags.

Compliance Monitoring: AWS Config continuously monitors resource configurations and flags non-compliant resources. This provides near real-time detection of EC2 instances without the required department tag. **Automatic Remediation:** Config supports automatic remediation. By linking the required-tags rule to the AWS-TerminateEC2Instance Systems Manager Automation document, any EC2 instance that violates the rule is automatically terminated. This ensures near real-time enforcement of the tagging policy.

Security: AWS Config operates within your AWS account, leveraging IAM roles to securely access and manage resources. The Systems Manager Automation document executes with defined permissions, ensuring controlled termination.

Why Other Options are Less Suitable:

Option B: EventBridge and SNS: While EventBridge can trigger actions based on EC2 instance creation, it requires more complex custom code to check for the tag and then terminate the instance. SNS would only provide notification; you would need an additional component to perform the termination. This adds overhead and complexity. EventBridge rules might also miss instances created before the rule was in place, leading to incomplete enforcement. https://aws.amazon.com/eventbridge/

Option C: IAM Permissions and Shutdown Behavior: This option relies on user enforcement and setting the instance's shutdown behavior. It is not proactive and provides no guarantee that instances will be correctly tagged. Users may simply forget to tag the instances, circumventing the policy. Furthermore, changing the shutdown behavior to terminate only affects instances that are explicitly shut down through the operating system or the console, and does not address instances that are running without the required tag.
https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies.html

Option D: Systems Manager Compliance and AWS-StopEC2Instances: Using Systems Manager Compliance can check for configuration compliance, but it does not offer the same continuous monitoring and automatic remediation of AWS Config. Stopping instances instead of terminating them leaves non-compliant resources consuming resources (EBS volumes, reserved IPs) which is not ideal. Moreover, stopping an instance might not be the desired action; termination is explicitly requested in the scenario. https://aws.amazon.com/systems-manager/features/

In conclusion, AWS Config with automatic remediation offers the most efficient, secure, and compliant solution for near real-time termination of non-compliant EC2 instances based on missing tags. It aligns perfectly with the requirement for proactive enforcement of tagging policies.