Amazon
SOA-C03
AWS Certified CloudOps Engineer - Associate

Latest Version: 6.3

Question: 1

A company's ecommerce application is running on Amazon EC2 instances that are behind an Application Load Balancer (ALB). The instances are in an Auto Scaling group. Customers report that the website is occasionally down. When the website is down, it returns an HTTP 500 (server error) status code to customer browsers.
The Auto Scaling group's health check is configured for EC2 status checks, and the instances appear healthy.
Which solution will resolve the problem?

A. Replace the ALB with a Network Load Balancer.
B. Add Elastic Load Balancing (ELB) health checks to the Auto Scaling group.
C. Update the target group configuration on the ALB. Enable session affinity (sticky sessions).
D. Install the Amazon CloudWatch agent on all instances. Configure the agent to reboot the instances.

Answer: B

Explanation:
In this scenario, the EC2 instances pass their EC2 status checks, indicating that the operating system is responsive. However, the application hosted on the instance is failing intermittently, returning HTTP 500 errors. This demonstrates a discrepancy between the instance-level health and the application-level health.
According to AWS CloudOps best practices under Monitoring, Logging, Analysis, Remediation and Performance Optimization (SOA-C03 Domain 1), Auto Scaling groups should incorporate Elastic Load Balancing (ELB) health checks instead of relying solely on EC2 status checks. The ELB health check probes the application endpoint (for example, HTTP or HTTPS target group health checks), ensuring that the application itself is functioning correctly.
When an instance fails an ELB health check, Amazon EC2 Auto Scaling will automatically mark the instance as unhealthy and replace it with a new one, ensuring continuous availability and performance optimization.
Extract from AWS CloudOps (SOA-C03) Study Guide – Domain 1:
"Implement monitoring and health checks using ALB and EC2 Auto Scaling integration. Application Load Balancer health checks allow Auto Scaling to terminate and replace instances that fail application-level health checks, ensuring consistent application performance."
Extract from AWS Auto Scaling Documentation:
"When you enable the ELB health check type for your Auto Scaling group, Amazon EC2 Auto Scaling considers both EC2 status checks and Elastic Load Balancing health checks to determine instance health. If an instance fails the ELB health check, it is automatically replaced."
Therefore, the correct answer is B, as it ensures proper application-level monitoring and remediation using ALB-integrated ELB health checks—a core CloudOps operational practice for proactive incident response and availability assurance.
References (AWS CloudOps Verified Source Extracts):

AWS Certified CloudOps Engineer – Associate (SOA-C03) Exam Guide: Domain 1 – Monitoring, Logging, and Remediation.
AWS Auto Scaling User Guide: Health checks for Auto Scaling instances (Elastic Load Balancing integration).
AWS Well-Architected Framework – Operational Excellence and Reliability Pillars.
AWS Elastic Load Balancing Developer Guide – Target group health checks and monitoring.

Question: 2

A company hosts a critical legacy application on two Amazon EC2 instances that are in one Availability Zone. The instances run behind an Application Load Balancer (ALB). The company uses Amazon CloudWatch alarms to send Amazon Simple Notification Service (Amazon SNS) notifications when the ALB health checks detect an unhealthy instance. After a notification, the company's engineers manually restart the unhealthy instance. A CloudOps engineer must configure the application to be highly available and more resilient to failures. Which solution will meet these requirements?

A. Create an Amazon Machine Image (AMI) from a healthy instance. Launch additional instances from the AMI in the same Availability Zone. Add the new instances to the ALB target group.
B. Increase the size of each instance. Create an Amazon EventBridge rule. Configure the EventBridge rule to restart the instances if they enter a failed state.
C. Create an Amazon Machine Image (AMI) from a healthy instance. Launch an additional instance from the AMI in the same Availability Zone. Add the new instance to the ALB target group. Create an AWS Lambda function that runs when an instance is unhealthy. Configure the Lambda function to stop and restart the unhealthy instance.
D. Create an Amazon Machine Image (AMI) from a healthy instance. Create a launch template that uses the AMI. Create an Amazon EC2 Auto Scaling group that is deployed across multiple Availability Zones. Configure the Auto Scaling group to add instances to the ALB target group.

Answer: D

Explanation:
High availability requires removing single-AZ risk and eliminating manual recovery. The AWS Reliability best practices state to design for multi-AZ and automatic healing: Auto Scaling “helps maintain application availability and allows you to automatically add or remove EC2 instances” (AWS Auto Scaling User Guide). The Reliability Pillar recommends to “distribute workloads across multiple Availability Zones” and to “automate recovery from failure” (AWS Well-Architected Framework – Reliability Pillar). Attaching the Auto Scaling group to an ALB target group enables health-based replacement: instances failing load balancer health checks are replaced and traffic is routed only to healthy targets. Using an AMI in a launch template ensures consistent, repeatable instance configuration (AWS EC2 Launch Templates).
Options A and C keep all instances in a single Availability Zone and rely on manual or ad-hoc restarts, which do not meet high-availability or resiliency goals. Option B only scales vertically and adds a restart rule; it neither removes the single-AZ failure domain nor provides automated replacement. Therefore, creating a multi-AZ EC2 Auto Scaling group with a launch template and attaching it to the ALB target group (Option D) is the CloudOps-aligned solution for resilience and business continuity.
References (AWS CloudOps Documents / Study Guide):

• AWS Certified CloudOps Engineer – Associate (SOA-C03) Exam Guide: Domain 2 – Reliability and Business Continuity
• AWS Well-Architected Framework – Reliability Pillar
• Amazon EC2 Auto Scaling User Guide – Health checks and replacement
• Elastic Load Balancing User Guide – Target group health checks and ALB integration
• Amazon EC2 Launch Templates – Reproducible instance configuration

Question: 3

An Amazon EC2 instance is running an application that uses Amazon Simple Queue Service (Amazon SQS) queues. A CloudOps engineer must ensure that the application can read, write, and delete messages from the SQS queues.
Which solution will meet these requirements in the MOST secure manner?

A. Create an IAM user with an IAM policy that allows the sqs:SendMessage permission, the sqs:ReceiveMessage permission, and the sqs:DeleteMessage permission to the appropriate queues. Embed the IAM user's credentials in the application's configuration.
B. Create an IAM user with an IAM policy that allows the sqs:SendMessage permission, the sqs:ReceiveMessage permission, and the sqs:DeleteMessage permission to the appropriate queues. Export the IAM user's access key and secret access key as environment variables on the EC2 instance.
C. Create and associate an IAM role that allows EC2 instances to call AWS services. Attach an IAM policy to the role that allows sqs:* permissions to the appropriate queues.
D. Create and associate an IAM role that allows EC2 instances to call AWS services. Attach an IAM policy to the role that allows the sqs:SendMessage permission, the sqs:ReceiveMessage permission, and the sqs:DeleteMessage permission to the appropriate queues.

Answer: D

Explanation:
The most secure pattern is to use an IAM role for Amazon EC2 with the minimum required permissions. AWS guidance states: “Use roles for applications that run on Amazon EC2 instances” and “grant least privilege by allowing only the actions required to perform a task.” By attaching a role to the instance, short-lived credentials are automatically provided through the instance metadata service; this removes the need to create long-term access keys or embed secrets. Granting only sqs:SendMessage, sqs:ReceiveMessage, and sqs:DeleteMessage against the specific SQS queues enforces least privilege and aligns with CloudOps security controls. Options A and B rely on IAM user access keys, which contravene best practices for workloads on EC2 and increase credential-management risk. Option C uses a role but grants sqs:*, violating least-privilege principles. Therefore, Option D meets the security requirement with scoped, temporary credentials and precise permissions.
References (AWS CloudOps Documents / Study Guide):
• AWS Certified CloudOps Engineer – Associate (SOA-C03) Exam Guide – Security & Compliance
• IAM Best Practices – “Use roles instead of long-term access keys,” “Grant least privilege”
• IAM Roles for Amazon EC2 – Temporary credentials for applications on EC2
• Amazon SQS – Identity and access management for Amazon SQS

Question: 4

A company runs an application that logs user data to an Amazon CloudWatch Logs log group. The company discovers that personal information the application has logged is visible in plain text in the CloudWatch logs.
The company needs a solution to redact personal information in the logs by default. Unredacted information must be available only to the company's security team. Which solution will meet these requirements?

A. Create an Amazon S3 bucket. Create an export task from appropriate log groups in CloudWatch. Export the logs to the S3 bucket. Configure an Amazon Macie scan to discover personal data in the S3 bucket. Invoke an AWS Lambda function to move identified personal data to a second S3 bucket. Update the S3 bucket policies to grant only the security team access to both buckets.
B. Create a customer managed AWS KMS key. Configure the KMS key policy to allow only the security team to perform decrypt operations. Associate the KMS key with the application log group.
C. Create an Amazon CloudWatch data protection policy for the application log group. Configure data identifiers for the types of personal information that the application logs. Ensure that the security team has permission to call the unmask API operation on the application log group.
D. Create an OpenSearch domain. Create an AWS Glue workflow that runs a Detect PII transform job and streams the output to the OpenSearch domain. Configure the CloudWatch log group to stream the logs to AWS Glue. Modify the OpenSearch domain access policy to allow only the security team to access the domain.

Answer: C

Explanation:
CloudWatch Logs data protection provides native redaction/masking of sensitive data at ingestion and query. AWS documentation states it can “detect and protect sensitive data in logs” using data identifiers, and that authorized users can “use the unmask action to view the original data.” Creating a data protection policy on the log group masks PII by default for all viewers, satisfying the requirement to redact personal information. Granting only the security team permission to invoke the unmask API operation ensures that unredacted content is restricted. Option B (KMS) encrypts at rest but does not redact fields; encryption alone does not prevent plaintext visibility to authorized readers. Options A and D add complexity and latency, move data out of CloudWatch, and do not provide default inline redaction/unmask controls in CloudWatch itself. Therefore, the CloudOps-aligned, managed solution is to use CloudWatch Logs data protection with appropriate data identifiers and unmask permissions limited to the security team.
References (AWS CloudOps Documents / Study Guide):
• AWS Certified CloudOps Engineer – Associate (SOA-C03) Exam Guide – Monitoring & Logging
• Amazon CloudWatch Logs – Data Protection (masking/redaction with data identifiers)
• CloudWatch Logs – Permissions for masking and unmasking sensitive data
• AWS Well-Architected Framework – Security and Operational Excellence (sensitive data handling)

Question: 5

A multinational company uses an organization in AWS Organizations to manage over 200 member accounts across multiple AWS Regions. The company must ensure that all AWS resources meet specific security requirements.
The company must not deploy any EC2 instances in the ap-southeast-2 Region. The company must completely block root user actions in all member accounts. The company must prevent any user from deleting AWS CloudTrail logs, including administrators. The company requires a centrally managed solution that the company can automatically apply to all existing and future accounts. Which solution will meet these requirements?

A. Create AWS Config rules with remediation actions in each account to detect policy violations. Implement IAM permissions boundaries for the account root users.
B. Enable AWS Security Hub across the organization. Create custom security standards to enforce the security requirements. Use AWS CloudFormation StackSets to deploy the standards to all the accounts in the organization. Set up Security Hub automated remediation actions.
C. Use AWS Control Tower for account governance. Configure Region deny controls. Use Service Control Policies (SCPs) to restrict root user access.
D. Configure AWS Firewall Manager with security policies to meet the security requirements. Use an AWS Config aggregator with organization-wide conformance packs to detect security policy violations.

Answer: C

Explanation:
AWS CloudOps governance best practices emphasize centralized account management and preventive guardrails. AWS Control Tower integrates directly with AWS Organizations and provides “Region deny controls” and “Service Control Policies (SCPs)” that apply automatically to all existing and newly created member accounts. SCPs are organization-wide guardrails that define the maximum permissions for accounts. They can explicitly deny actions such as launching EC2 instances in a specific Region, or block root user access.
To prevent CloudTrail log deletion, SCPs can also include denies on cloudtrail:DeleteTrail and s3:DeleteObject actions targeting the CloudTrail log S3 bucket. These SCPs ensure that no user, including administrators, can violate the compliance requirements.
AWS documentation under the Security and Compliance domain for CloudOps states:
“Use AWS Control Tower to establish a secure, compliant, multi-account environment with preventive guardrails through service control policies and detective controls through AWS Config.”
This approach meets all stated needs: centralized enforcement, automatic propagation to new accounts, region-based restrictions, and immutable audit logs. Options A, B, and D either detect violations reactively or lack complete enforcement and automation across future accounts.
References (AWS CloudOps Documents / Study Guide):
• AWS Certified CloudOps Engineer – Associate (SOA-C03) Exam Guide – Domain 4: Security and Compliance
• AWS Control Tower – Preventive and Detective Guardrails
• AWS Organizations – Service Control Policies (SCPs)
• AWS Well-Architected Framework – Security Pillar (Governance and Centralized Controls)

Thank You for Trying Our Product

For More Information – Visit link below:

https://www.examsboost.com/

15 USD Discount Coupon Code:

G74JA8UF

FEATURES

- ✓ 90 Days Free Updates
- ✓ Money Back Pass Guarantee
- ✓ Instant Download or Email Attachment
- ✓ 24/7 Live Chat Support
- ✓ PDF file could be used at any Platform
- ✓ 50,000 Happy Customer