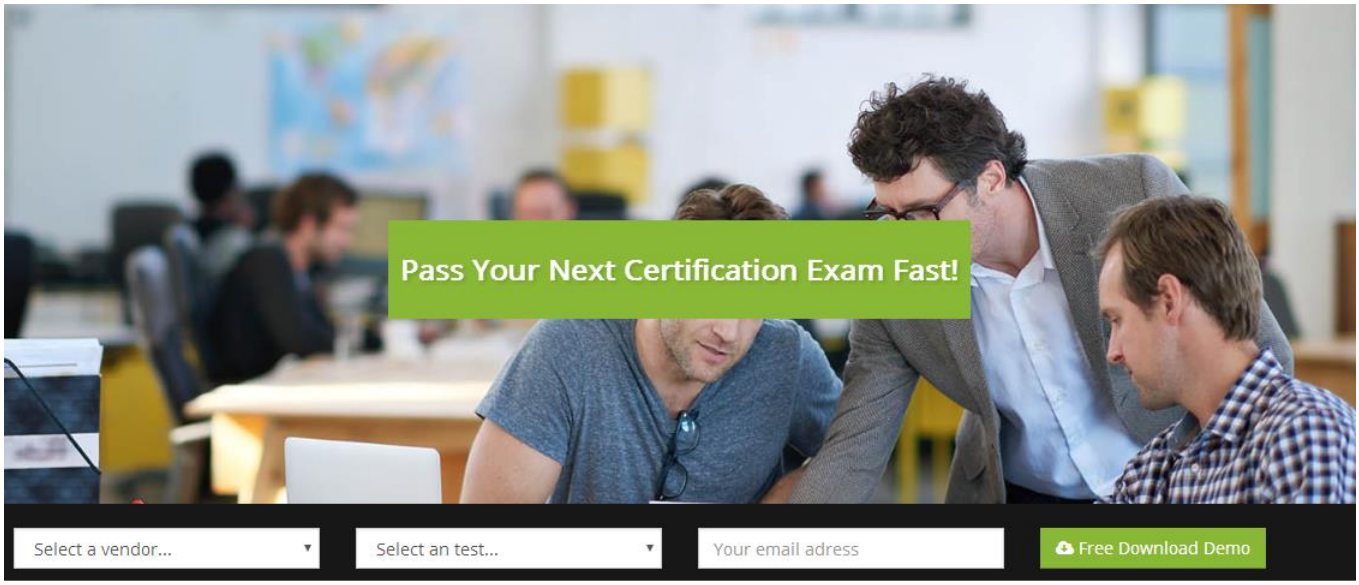
ITPassLeader



Instant Download | 365 Days Free Updates | Money Back Guarantee | Security & Privacy

Choose the version that fits your needs	PDF Version	Desktop Test Engine	Online Test Engine
Latest and Up-to-Date exam dumps with real exam questions answers.	✓	✓	✓
Get 12-Months free updates without any extra charges.	✓	✓	✓
Experience same exam environment before appearing in the certification exam.	✗	✓	✓
100% exam passing guarante in the first attempt.	✓	✓	✓
20% discount on more than one license and 30% discount on 5+ license purchases.	✗	✓	✓
100% secure purchase on SSL.	✓	✓	✓
Completely private purchase without sharing your personal info with anyone.	✓	✓	✓

http://www.itpassleader.com

High-praise Exam Dumps Questions grant you success by high pass rate - ITPassLeader

Exam : **SOA-C03**

Title : AWS Certified CloudOps Engineer - Associate

Vendor : Amazon

Version : DEMO

NO.1 A company uses multiple Amazon RDS databases to support an application. The application receives all its traffic during weekdays and is idle during weekends. The company wants a solution to automatically manage the RDS DB instances during idle periods to optimize costs.
Which solution will meet these requirements?

A. Use a cron job to automatically scale down the RDS DB instance type during weekends.

B. Configure Instance Scheduler on AWS to stop the RDS DB instances at the beginning of each weekend and to start the instances at the end of each weekend.

C. Purchase Reserved Instances for the RDS DB instances.

D. Use the auto scaling feature of Amazon RDS to automatically adjust the DB instance type based on CPU utilization.

Answer: B

Explanation:
The Instance Scheduler on AWS is an AWS-provided solution designed specifically to start and stop AWS resources such as Amazon RDS instances on a defined schedule. This directly aligns with the requirement to automatically manage RDS instances during predictable idle periods, such as weekends, to reduce costs.
RDS instances incur compute charges while running, even if idle. Stopping them during weekends eliminates those charges while retaining storage and backups. Instance Scheduler supports tag-based scheduling, centralized management, and automated start/stop workflows without custom scripting.
Option A introduces custom automation and ongoing maintenance overhead. Option C (Reserved Instances) is unsuitable because the databases are idle for long, predictable periods and Reserved Instances charge regardless of usage. Option D is incorrect because RDS does not support auto scaling of DB instance classes based on utilization.
Instance Scheduler is the most cost-effective and operationally efficient solution for this use case.

NO.2 A company uses an Amazon Simple Queue Service (Amazon SQS) queue and Amazon EC2 instances in an Auto Scaling group with target tracking for a web application. The company collects the ASGAverageNetworkIn metric but notices that instances do not scale fast enough during peak traffic. There are a large number of SQS messages accumulating in the queue.
A CloudOps engineer must reduce the number of SQS messages during peak periods.
Which solution will meet this requirement?

A. Define and use a new custom Amazon CloudWatch metric based on the SQS ApproximateNumberOfMessagesDelayed metric in the target tracking policy.

B. Define and use Amazon CloudWatch metric math to calculate the SQS queue backlog for each instance in the target tracking policy.

C. Define and use step scaling by specifying a ChangeInCapacity value for the EC2 instances.

D. Define and use simple scaling by specifying a ChangeInCapacity value for the EC2 instances.

Answer: B

Explanation:
According to the AWS Cloud Operations and Auto Scaling documentation, scaling applications that consume Amazon SQS messages should be driven by queue backlog per instance, not by general system metrics such as network traffic or CPU.
The correct approach is to calculate a custom metric using CloudWatch metric math that divides the SQS metric ApproximateNumberOfMessagesVisible by the number of active EC2 instances in the Auto Scaling group. This "backlog per instance" value represents the average number of messages

waiting to be processed by each instance.
Then, the CloudOps engineer can create a target tracking policy that automatically scales out or in based on maintaining a desired backlog threshold. This approach ensures dynamic, workload-driven scaling behavior that reacts in near real time to message volume.
Step and simple scaling (Options C and D) require manual thresholds and do not automatically balance the load per instance.
Thus, Option B-using CloudWatch metric math to define queue backlog per instance for target tracking-is the most effective and AWS-recommended CloudOps practice.
Reference: AWS Cloud Operations & Scaling Guide - Scaling Based on Amazon SQS Queue Backlog Per Instance

NO.3 A company has a non-production application that runs on an Amazon EC2 instance. The Amazon CloudWatch agent is installed on the EC2 instance. The application includes a process that randomly overuses temporary disk space and fills disks to 100% capacity.
A CloudOps engineer needs to automate a reboot of the EC2 instance after the disks reach 100% capacity.
Which solution will meet this requirement in the MOST operationally efficient way?

A. Create a CloudWatch alarm for the EC2 instance. Create an Amazon EventBridge event rule that reacts to the CloudWatch alarm and reboots the EC2 instance.

B. Create a CloudWatch alarm for the EC2 instance. Create an Amazon SES notification that reacts to the CloudWatch alarm and reboots the EC2 instance.

C. Create an AWS Lambda function to reboot the EC2 instance. Create a CloudWatch alarm that uses Amazon EventBridge to invoke the Lambda function.

D. Create an AWS Lambda function to reboot the EC2 instance. Use EC2 health checks to invoke the Lambda function.

Answer: A
Explanation:
The CloudWatch agent can publish custom disk usage metrics from the EC2 instance. A CloudWatch alarm can monitor the disk usage metric and enter the ALARM state when disk usage reaches 100%. The most operationally efficient approach is to connect the alarm state change to an Amazon EventBridge rule and use the rule to trigger an EC2 reboot action. This avoids custom polling and reduces manual intervention. Option B is wrong because Amazon SES sends email and does not reboot EC2 instances. Option C can work, but it adds unnecessary Lambda code and maintenance when EventBridge can directly react to CloudWatch alarm state changes. Option D is wrong because EC2 health checks do not detect application-level temporary disk exhaustion in this way. Therefore, CloudWatch alarm plus EventBridge automation is the cleanest CloudOps solution.

NO.4 A company uses Amazon ElastiCache (Redis OSS) to cache application data. A CloudOps engineer must implement a solution to increase the resilience of the cache and minimize the recovery time objective (RTO).
Which solution will meet these requirements?

A. Replace ElastiCache (Redis OSS) with ElastiCache (Memcached).

B. Create an Amazon EventBridge rule to initiate a backup every hour.

C. Create a read replica in a second Availability Zone and enable Multi-AZ for the Redis replication group.

D. Enable automatic backups and restore the backups when necessary.

Answer: C

Explanation:

Amazon ElastiCache for Redis supports Multi-AZ replication groups, which provide high availability by automatically promoting a replica in another Availability Zone if the primary node fails. This architecture significantly reduces recovery time because failover occurs automatically without manual intervention.

Creating a read replica in a second AZ ensures redundancy and resilience against AZ-level failures. Enabling Multi-AZ allows Redis to maintain availability during infrastructure issues or maintenance events.

Option A removes persistence and high availability features. Options B and D rely on backups, which increase RTO because restore operations take time and require manual steps.

Therefore, Multi-AZ Redis with replicas provides the best combination of resilience and minimal RTO.

NO.5 A company runs its applications on a large number of Amazon EC2 instances. A CloudOps engineer must implement a solution to notify the operations team whenever an EC2 instance state changes.

What is the MOST operationally efficient solution that meets these requirements?

A. Create a script that captures instance state changes and publishes a notification to an Amazon SNS topic. Use AWS Systems Manager Run Command to run the script on all EC2 instances.

B. Create an Amazon EventBridge event rule that captures EC2 instance state changes. Set an Amazon SNS topic as the target.

C. Create an Amazon EventBridge event rule that captures EC2 instance state changes. Set as the target an AWS Lambda function that publishes a notification to an Amazon SNS topic.

D. Create an AWS Config custom rule that evaluates instance state changes with automatic remediation.Use the rule to invoke an AWS Lambda function that publishes a notification to an Amazon SNS topic.

Answer: B

Explanation:

Amazon EventBridge receives EC2 instance state-change events and can route matching events directly to a target such as an Amazon SNS topic. This is the most operationally efficient solution because it uses native event-driven integration and does not require scripts, agents, polling, or custom Lambda code. Option A is poor operational design because every instance would need script execution and maintenance. Option C adds an unnecessary Lambda function; EventBridge can publish to SNS directly. Option D misuses AWS Config, which is better suited to configuration compliance and resource-state evaluation, not simple near-real-time notification of every EC2 instance state transition. For CloudOps event monitoring, EventBridge rules are the standard approach for reacting to AWS service events and notifying operators.

NO.6 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 CloudOps engineer must find the process ID (PID) of the service or process that is consuming more CPU.

What should the CloudOps engineer 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 each night and run the top command.
D. Use the default Amazon CloudWatch CPUUtilization metric.
Answer: A
Explanation:
The CloudWatch agent procstat plugin is specifically designed to collect per-process metrics such as CPU usage by process ID. It allows continuous, automated monitoring without manual intervention and integrates directly with CloudWatch.
Default CloudWatch metrics provide only instance-level CPU utilization and do not expose process-level details. Manual logins and Lambda-based polling introduce unnecessary operational overhead and are not scalable.
Therefore, configuring the procstat plugin is the most efficient solution.

NO.7 A medical research company uses an Amazon Bedrock powered AI assistant with agents and knowledge bases to provide physicians quick access to medical study protocols. The company needs to generate audit reports that contain user identities, usage data for Bedrock agents, access data for knowledge bases, and interaction parameters.
Which solution will meet these requirements?
A. Use AWS CloudTrail to log API events from generative AI workloads. Store the events in CloudTrail Lake. Use SQL-like queries to generate reports.
B. Use Amazon CloudWatch to capture generative AI application logs. Stream the logs to Amazon OpenSearch Service. Use an OpenSearch dashboard visualization to generate reports.
C. Use Amazon CloudWatch to log API events from generative AI workloads. Send the events to an Amazon S3 bucket. Use Amazon Athena queries to generate reports.
D. Use AWS CloudTrail to capture generative AI application logs. Stream the logs to Amazon Managed Service for Apache Flink. Use SQL queries to generate reports.
Answer: A
Explanation:
As per AWS Cloud Operations, Bedrock, and Governance documentation, AWS CloudTrail is the authoritative service for capturing API activity and audit trails across AWS accounts. For Amazon Bedrock, CloudTrail records all user-initiated API calls, including interactions with agents, knowledge bases, and generative AI model parameters.
Using CloudTrail Lake, organizations can store, query, and analyze CloudTrail events directly without needing to export data. CloudTrail Lake supports SQL-like queries for generating audit and compliance reports, enabling the company to retrieve information such as user identity, API usage, timestamp, model or agent ID, and invocation parameters.
In contrast, CloudWatch focuses on operational metrics and log streaming, not API-level identity data.
OpenSearch or Flink would add unnecessary complexity and cost for this use case.
Thus, the AWS-recommended CloudOps best practice is to leverage CloudTrail with CloudTrail Lake to maintain auditable, queryable API activity for Bedrock workloads, fulfilling governance and compliance requirements.
Reference: AWS Cloud Operations & Governance Guide - Section: Auditing and Governance for

Generative AI Workloads Using AWS CloudTrail and CloudTrail Lake

NO.8 A CloudOps engineer needs to control access to groups of Amazon EC2 instances using AWS Systems Manager Session Manager. Specific tags on the EC2 instances have already been added. Which additional actions should the CloudOps engineer take to control access? (Select TWO.)

A. Attach an IAM policy to the users or groups that require access to the EC2 instances.

B. Attach an IAM role to control access to the EC2 instances.

C. Create a placement group for the EC2 instances and add a specific tag.

D. Create a service account and attach it to the EC2 instances that need to be controlled.

E. Create an IAM policy that grants access to any EC2 instances with a tag specified in the Condition element.

Answer: A E

Explanation:

AWS Systems Manager Session Manager allows secure, auditable instance access without SSH keys or inbound ports. To control access based on instance tags, CloudOps best practices require two configurations:

* Attach an IAM policy to users or groups granting ssm:StartSession, ssm:DescribeInstanceInformation, and ssm:DescribeSessions.
* Include a Condition element in the IAM policy referencing instance tags, such as Condition: { " StringEquals " : { " ssm:resourceTag/Environment " : " Production " }}.

This ensures users can start sessions only with instances that have matching tags, providing fine-grained access control.

AWS CloudOps documentation under Security and Compliance states:

"Use IAM policies with resource tags in the Condition element to restrict which managed instances users can access using Session Manager." Options B and D incorrectly suggest attaching roles or service accounts that are not relevant to user-level access control. Option C (placement groups) pertains to networking and performance, not access management. Therefore, A and E together provide tag-based, least-privilege access as required.

References:* AWS Certified CloudOps Engineer - Associate (SOA-C03) Exam Guide - Domain 4: Securit y and Compliance* AWS Systems Manager User Guide - Controlling Access to Session Manager Using Tags* AWS IAM Policy Reference - Condition Keys for AWS Systems Manager* AWS Well-Architected Framework - Security Pillar

NO.9 A SysOps administrator creates a custom Amazon Machine Image (AMI) in the eu-west-2 Region and uses the AMI to launch Amazon EC2 instances. The SysOps administrator needs to use the same AMI to launch EC2 instances in two other Regions: us-east-1 and us-east-2.

What must the SysOps administrator do to use the custom AMI in the additional Regions?

A. Copy the AMI to the additional Regions

B. Make the AMI public in the Community AMIs section of the AWS Management Console

C. Share the AMI to the additional Regions. Assign the required access permissions.

D. Copy the AMI to a new Amazon S3 bucket. Assign access permissions to the AMI for the additional Regions

Answer: A

Explanation:

Comprehensive and Detailed Explanation From Exact Extract of AWS CloudOps Documents:

Amazon Machine Images (AMIs) are Region-specific resources. AWS CloudOps documentation explicitly states that an AMI created in one Region cannot be used to launch instances in another Region unless it is copied to the target Region. Therefore, the SysOps administrator must copy the AMI to both us-east-1 and us- east-2.
The AMI copy process creates a new AMI in each destination Region and automatically copies the underlying snapshots. Once the AMIs exist in the target Regions, they can be referenced in launch templates, Auto Scaling groups, or AWS CloudFormation templates for consistent multi-Region deployments.
Option B is incorrect because making an AMI public does not replicate it across Regions. Option C is incorrect because sharing an AMI only grants account-level access within the same Region. Option D is incorrect because AMIs cannot be launched from Amazon S3 directly.
This approach aligns with AWS CloudOps automation practices for multi-Region application deployment and disaster recovery readiness.
References:
Amazon EC2 User Guide - Copying an AMI across Regions
AWS SysOps Administrator Study Guide - AMI lifecycle management
AWS Well-Architected Framework - Deployment and automation best practices

NO.10 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.

NO.11 A company requires the rotation of administrative credentials for production workloads on a regular basis. A CloudOps engineer must implement this policy for an Amazon RDS DB instance ' s master user password.
Which solution will meet this requirement with the LEAST operational effort?
A. Create an AWS Lambda function to change the RDS master user password. Create an Amazon EventBridge scheduled rule to invoke the Lambda function.
B. Create a new SecureString parameter in AWS Systems Manager Parameter Store. Encrypt the parameter with an AWS Key Management Service (AWS KMS) key. Configure automatic rotation.
C. Create a new String parameter in AWS Systems Manager Parameter Store. Configure automatic rotation.
D. Create a new RDS database secret in AWS Secrets Manager. Apply the secret to the RDS DB instance.Configure automatic rotation.
Answer: D
Explanation:
AWS Secrets Manager natively supports credential management and automatic rotation for Amazon RDS master user passwords. When a secret is associated with an RDS instance, Secrets Manager automatically updates the password both in the secret and on the database, without downtime or manual scripting.
AWS documentation confirms:
"AWS Secrets Manager can automatically rotate the master user password for Amazon RDS databases.
Rotation is fully managed and integrated, requiring no custom code or maintenance." Option A introduces unnecessary Lambda automation. Option B and C use Parameter Store, which does not provide direct RDS password rotation. Therefore, Option D achieves secure, automatic credential rotation with least operational effort, fully aligned with CloudOps security automation principles.
References:* AWS Certified CloudOps Engineer - Associate (SOA-C03) Exam Guide - Domain 4: Securit y and Compliance* AWS Secrets Manager - Rotating Secrets for Amazon RDS* AWS Well-Architected Framework - Security Pillar* Amazon RDS User Guide - Managing Master User Passwords

NO.12 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 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)

NO.13 A company uses AWS CloudFormation to manage a stack of Amazon EC2 instances. A CloudOps engineer needs to keep the EC2 instances and their data even if the stack is deleted. Which solution will meet these requirements?
A. Set the DeletionPolicy attribute to Snapshot.
B. Use Amazon Data Lifecycle Manager (DLM).

C. Create an AWS Backup plan.

D. Set the DeletionPolicy attribute to Retain.

Answer: D

Explanation:

CloudFormation's DeletionPolicy: Retain ensures that resources are not deleted when the stack is deleted.

This preserves both the EC2 instance and any attached storage.

Snapshot does not apply to EC2 instances themselves. Backup solutions do not prevent deletion.

Therefore, Retain is the correct policy.

NO.14 A CloudOps engineer is troubleshooting an AWS CloudFormation stack creation that failed. Before the CloudOps engineer can identify the problem, the stack and its resources are deleted. For future deployments, the CloudOps engineer must preserve any resources that CloudFormation successfully created.

What should the CloudOps engineer do to meet this requirement?

A. Set the value of the DisableRollback parameter to False during stack creation.

B. Set the value of the OnFailure parameter to DO_NOTHING during stack creation.

C. Specify a rollback configuration that has a rollback trigger of DO_NOTHING during stack creation.

D. Set the value of the OnFailure parameter to ROLLBACK during stack creation.

Answer: B

Explanation:

By default, when AWS CloudFormation encounters a failure during stack creation, it automatically rolls back and deletes any resources that were successfully created. This behavior makes troubleshooting difficult because the failed and partially created resources are no longer available for inspection.

CloudFormation provides the OnFailure parameter to control this behavior. Setting the parameter to DO_NOTHING instructs CloudFormation to stop stack creation when a failure occurs and retain all successfully created resources. This allows the CloudOps engineer to inspect the environment, review logs, and identify the root cause without redeploying resources.

The DisableRollback parameter controls rollback behavior but does not provide the same explicit behavior control during failure scenarios. Rollback triggers are used for monitoring-based rollback, not for preserving resources on failure. Setting OnFailure to ROLLBACK explicitly enforces deletion, which is the opposite of the requirement.

Therefore, setting the OnFailure parameter to DO_NOTHING is the correct solution.

NO.15 A company is using an Amazon Aurora MySQL DB cluster that has point-in-time recovery, backtracking, and automatic backup enabled. A CloudOps engineer needs to roll back the DB cluster to a specific recovery point within the previous 72 hours. Restores must be completed in the same production DB cluster.

Which solution will meet these requirements?

A. Create an Aurora Replica. Promote the replica to replace the primary DB instance.

B. Create an AWS Lambda function to restore an automatic backup to the existing DB cluster.

C. Use backtracking to rewind the existing DB cluster to the desired recovery point.

D. Use point-in-time recovery to restore the existing DB cluster to the desired recovery point.

Answer: C

Explanation:
As documented in AWS Cloud Operations and Database Recovery, Aurora Backtrack allows you to rewind the existing database cluster to a chosen point in time without creating a new cluster. This feature supports fine-grained rollback for accidental data changes, making it ideal for scenarios like table deletions or logical corruption.
Backtracking maintains continuous transaction logs and permits rewinding within a configurable window (up to 72 hours). It does not require creating a new cluster or endpoint, and it preserves the same production environment, fulfilling the operational requirement for in-place recovery.
In contrast, Point-in-Time Recovery (Option D) always creates a new cluster, while replica promotion (Option A) and Lambda restoration (Option B) are unrelated to immediate rollback operations.
Therefore, Option C, using Aurora Backtrack, best meets the requirement for same-cluster restoration and minimal downtime.
Reference: AWS Cloud Operations & Database Management Guide - Section: Using Aurora Backtrack for Fast In-Place Recovery

NO.16 A company is running workloads on premises and on AWS. A CloudOps engineer needs to automate tasks across all servers on premises by using AWS services. The CloudOps engineer must not install long-term credentials on the on-premises servers.
What should the CloudOps engineer do to meet these requirements?
A. Create an IAM role and instance profile that include AWS Systems Manager permissions. Attach the role to the on-premises servers.
B. Create a managed-instance activation in AWS Systems Manager. Install the Systems Manager Agent on the on-premises servers. Register the servers with the activation code and ID from the managed- instance activation.
C. Create an AWS managed IAM policy that includes the appropriate AWS Systems Manager permissions. Download the IAM policy to the on-premises servers.
D. Create an IAM user and an access key. Log on to the on-premises servers and install the AWS CLI.Configure the access key in the AWS credentials file after the AWS CLI is successfully installed.
Answer: B
Explanation:
AWS Systems Manager supports hybrid and multicloud managed nodes through managed-instance activations. The CloudOps engineer creates an activation in Systems Manager, installs SSM Agent on the on- premises servers, and registers the servers by using the activation code and activation ID. This allows Systems Manager to manage those servers without storing long-term IAM user credentials on each machine. Option A is wrong because IAM instance profiles attach to EC2 instances, not on-premises servers. Option C misunderstands IAM policies; policies are permission documents, not credentials that can be downloaded to servers. Option D violates the requirement because IAM access keys are long-term credentials and should not be installed on servers. Managed-instance activation is the secure and operationally correct hybrid management approach.

NO.17 A CloudOps engineer creates an AWS CloudFormation template to define an application stack that can be deployed in multiple AWS Regions. The CloudOps engineer also creates an Amazon CloudWatch dashboard by using the AWS Management Console. Each deployment of the application requires its own CloudWatch dashboard.
How can the CloudOps engineer automate the creation of the CloudWatch dashboard each time the application is deployed?

A. Create a script by using the AWS CLI to run the aws cloudformation put-dashboard command with the name of the dashboard. Run the command each time a new CloudFormation stack is created.
B. Export the existing CloudWatch dashboard as JSON. Update the CloudFormation template to define an AWS::CloudWatch::Dashboard resource. Include the exported JSON in the resource ' s DashboardBody property.
C. Update the CloudFormation template to define an AWS::CloudWatch::Dashboard resource. Use the intrinsic Ref function to reference the ID of the existing CloudWatch dashboard.
D. Update the CloudFormation template to define an AWS::CloudWatch::Dashboard resource. Specify the name of the existing dashboard in the DashboardName property.
Answer: B
Explanation:
According to CloudOps automation and monitoring best practices, CloudWatch dashboards should be provisioned as infrastructure-as-code (IaC) resources using AWS CloudFormation to ensure consistency, repeatability, and version control. AWS CloudFormation supports the AWS::CloudWatch::Dashboard resource, where the DashboardBody property accepts a JSON object describing widgets, metrics, and layout.
By exporting the existing dashboard configuration as JSON and embedding it into the CloudFormation template, every deployment of the application automatically creates its corresponding dashboard. This method aligns with the CloudOps requirement for automated deployment and operational visibility within the same stack lifecycle.
AWS documentation explicitly states:
"Use the AWS::CloudWatch::Dashboard resource to create a dashboard from your template. You can include the same JSON you use to define a dashboard in the console." Option A requires manual execution. Options C and D incorrectly reference or reuse existing dashboards, failing to produce unique, deployment-specific dashboards.
References:* AWS Certified CloudOps Engineer - Associate (SOA-C03) Exam Guide - Domain 1: Monitoring and Logging* AWS CloudFormation User Guide - Resource Type: AWS::CloudWatch:: Dashboard* AWS Well-Architected Framework - Operational Excellence Pillar* Amazon CloudWatch - Automating Dashboards with Infrastructure as Code

NO.18 A company moves workloads from public subnets to private subnets to improve security. During testing, the company discovers that servers in the private subnets cannot reach an external API. The VPC has a CIDR block of 10.0.0.0/16. The VPC contains two public subnets and two private subnets. The VPC has one internet gateway and has a NAT gateway in each of the private subnets. The company must ensure that workloads that run in the private subnets can reach the external API. Which solution will meet this requirement?
A. Deploy an outbound-only internet gateway to allow traffic from private subnets to the internet. Edit the route tables to direct outbound traffic through the outbound-only internet gateway.
B. Create and configure an Amazon API Gateway HTTP API as a proxy for the external API. Edit the route tables to direct outbound traffic to the HTTP API.
C. Deploy a new NAT gateway that has an Elastic IP address in each public subnet. Edit the route tables to direct outbound traffic through the NAT gateways.
D. Create a VPC interface endpoint. Edit the route tables to direct outbound traffic through the endpoint.
Answer: C

Explanation:
Instances in private subnets do not have direct routes to an internet gateway, so they cannot reach public internet endpoints unless the VPC provides a managed egress path. The standard AWS architecture for private-subnet outbound internet access is to use a NAT device (typically a NAT gateway) placed in a public subnet, with an Elastic IP address and a route from the private subnet to that NAT gateway for 0.0.0.0/0 traffic. The NAT gateway then forwards traffic to the internet through the VPC's internet gateway while preventing unsolicited inbound connections to the private instances.
The scenario states that the VPC has a NAT gateway in each private subnet. That placement prevents proper egress because a NAT gateway itself must be in a public subnet with a route to the internet gateway to function for internet-bound traffic. Without that, private instances will fail to reach external services such as a third-party API, exactly as observed.
Option C corrects the architecture by deploying NAT gateways in the public subnets with Elastic IPs and updating the private subnet route tables so that outbound internet traffic targets the NAT gateways. This enables private workloads to initiate outbound connections while remaining unreachable from the public internet directly, maintaining the intended security posture.
Option A is incorrect because an outbound-only internet gateway is designed for IPv6 egress, not IPv4 internet access. Option B misunderstands routing: API Gateway is an application-layer proxy service and is not a target for VPC route tables to provide generic internet access. Option D is not applicable because VPC interface endpoints provide private connectivity to supported AWS services via PrivateLink, not to arbitrary external public APIs.
Therefore, deploying NAT gateways in the public subnets and routing private subnet egress through them is the correct solution.

NO.19 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 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

NO.20 A company uses memory-optimized Amazon EC2 instances behind a Network Load Balancer (NLB) to run an application. The company launched the EC2 instances from an AWS-provided Red Hat Enterprise Linux (RHEL) AMI.
A CloudOps engineer must monitor RAM utilization in 5-minute intervals. The CloudOps engineer must ensure that the EC2 instances scale in and out appropriately based on incoming load.
Which solution will meet these requirements?
A. Configure detailed monitoring for the EC2 instances. Configure the Amazon CloudWatch agent on the EC2 instances. Create an EC2 Auto Scaling group and Auto Scaling policy that is based on the mem_active metric.
B. Configure detailed monitoring for the EC2 instances. Use the mem_used_percent metric that the detailed monitoring feature provides. Create an IAM role that allows the CloudWatch agent to upload data. Create an EC2 Auto Scaling group and Auto Scaling policy that is based on the mem_used_percent metric.
C. Configure basic monitoring for the EC2 instances. Configure the Amazon CloudWatch agent on the EC2 instances. Create an IAM role that allows the CloudWatch agent to upload data. Create an EC2 Auto Scaling group and Auto Scaling policy that is based on the mem_used_percent metric.
D. Configure basic monitoring for the EC2 instances. Use the standard mem_used_percent metric for monitoring. Create an EC2 Auto Scaling group and Auto Scaling policy that is based on the mem_used_percent metric.
Answer: C
Explanation:
EC2 does not publish RAM utilization as a native CloudWatch metric by default. Memory metrics such as mem_used_percent are typically collected by the CloudWatch Agent, which runs on the instance and publishes custom metrics to CloudWatch. Because the requirement is RAM utilization at 5-minute intervals, the CloudWatch Agent can be configured to emit metrics at that cadence (or faster)
.
"Detailed monitoring" for EC2 mainly affects EC2-provided metrics (like CPU) by changing the period from
5 minutes (basic) to 1 minute (detailed). It does not magically provide memory utilization. Therefore, the key requirement is installing/configuring the CloudWatch Agent and ensuring it has permissions to publish metrics (via an IAM role attached to the instance / instance profile).
Option C correctly combines: (1) basic monitoring (fine for the ask), (2) CloudWatch Agent to publish

mem_used_percent, (3) IAM role permissions to allow publishing, and (4) Auto Scaling policy that scales based on the memory metric.
Option B incorrectly implies detailed monitoring provides mem_used_percent (it does not). Option D assumes a "standard" memory metric exists without the agent, which is not correct. Option A references mem_active, which is not the typical metric name exposed by CloudWatch Agent's standard memory measurements for scaling policies, and also omits the IAM role requirement needed for publishing custom metrics.
Thus, C is the AWS-correct path for memory-based scaling using CloudWatch custom metrics.

NO.21 A CloudOps engineer has successfully deployed a VPC with an AWS CloudFormation template. The CloudOps engineer 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:
CloudFormation StackSets are designed specifically to deploy and manage the same CloudFormation stack across multiple AWS accounts and Regions from a centralized administration point. When accounts are managed under AWS Organizations, StackSets can integrate directly with Organizations to target Organizational Units (OUs) or specific accounts, which minimizes operational overhead. This removes the need to manually assume roles and deploy stacks one-by-one, and avoids building custom automation (Lambda + cross-account role assumption + account enumeration + error handling + retries + idempotency).
With StackSets, you define the template once and then create stack instances across many accounts in a controlled and consistent manner. StackSets provide built-in features for: (1) centralized rollout and updates, (2) drift detection, (3) automatic deployment to new accounts (when using Organizations integration), and (4) standardized execution roles. This is operationally simpler and safer than scripting because StackSets handle orchestration and track deployment state per account/Region.
Option A increases manual work and is error-prone at scale. Options B and C require custom Lambda orchestration, cross-account permissions, and ongoing maintenance for failure modes and change management. StackSets provide the native "least ops" approach to multi-account CloudFormation deployments under Organizations.

NO.22 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 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