Free Amazon DOP-C02 Practice Questions

Shared by Knox on 12-08-2026

For More Free Questions and Preparation Resources

Check the Links on Last Page

Question 1

Question Type: MultipleChoice

A development team uses AWS CodeCommit, AWS CodePipeline, and AWS CodeBuild to develop and deploy an application. Changes to the code are submitted by pull requests. The development team reviews and merges the pull requests, and then the pipeline builds and tests the application.

Over time, the number of pull requests has increased. The pipeline is frequently blocked because of failing tests. To prevent this blockage, the development team wants to run the unit and integration tests on each pull request before it is merged.

Which solution will meet these requirements?

Options:

A- Create a CodeBuild project to run the unit and integration tests. Create a CodeCommit approval rule template. Configure the template to require the successful invocation of the CodeBuild project. Attach the approval rule to the project's CodeCommit repository.
B- Create an Amazon EventBridge rule to match pullRequestCreated events from CodeCommit Create a CodeBuild project to run the unit and integration tests. Configure the CodeBuild project as a target of the EventBridge rule that includes a custom event payload with the CodeCommit repository and branch information from the event.
C- Create an Amazon EventBridge rule to match pullRequestCreated events from CodeCommit. Modify the existing CodePipeline pipeline to not run the deploy steps if the build is started from a pull request. Configure the EventBridge rule to run the pipeline with a custom payload that contains the CodeCommit repository and branch information from the event.
D- Create a CodeBuild project to run the unit and integration tests. Create a CodeCommit notification rule that matches when a pull request is created or updated. Configure the notification rule to invoke the CodeBuild project.

Answer:

B

Explanation:

CodeCommit generates events in CloudWatch, CloudWatch triggers the CodeBuild
https://aws.amazon.com/es/blogs/devops/complete-ci-cd-with-aws-codecommit-aws-codebuild-aws-codedeploy-and-aws-codepipeline/

Question 2

Question Type: MultipleChoice

A DevOps engineer manages a Java-based application that runs in an Amazon Elastic Container Service (Amazon ECS) cluster on AWS Fargate. Auto scaling has not been configured for the application. The DevOps engineer has determined that the Java Virtual Machine (JVM) thread count is a good indicator of when to scale the application. The application serves customer traffic on port 8080 and makes JVM metrics available on port 9404. Application use has recently increased. The DevOps engineer needs to configure auto scaling for the application. Which solution will meet these requirements with the LEAST operational overhead?

Options:

A- Deploy the Amazon CloudWatch agent as a container sidecar. Configure the CloudWatch agent to retrieve JVM metrics from port 9404. Create CloudWatch alarms on the JVM thread count metric to scale the application. Add a step scaling policy in Fargate to scale up and scale down based on the CloudWatch alarms.
B- Deploy the Amazon CloudWatch agent as a container sidecar. Configure a metric filter for the JVM thread count metric on the CloudWatch log group for the CloudWatch agent. Add a target tracking policy in Fargate. Select the metric from the metric filter as a scale target.
C- Create an Amazon Managed Service for Prometheus workspace. Deploy AWS Distro for OpenTelemetry as a container sidecar to publish the JVM metrics from port 9404 to the Prometheus workspace. Configure rules for the workspace to use the JVM thread count metric to scale the application. Add a step scaling policy in Fargate. Select the Prometheus rules to scale up and scaling down.
D- Create an Amazon Managed Service for Prometheus workspace. Deploy AWS Distro for OpenTelemetry as a container sidecar to retrieve JVM metrics from port 9404 to publish the JVM metrics from port 9404 to the Prometheus workspace. Add a target tracking policy in Fargate. Select the Prometheus metric as a scale target.

Answer:

A

Question 3

Question Type: MultipleChoice

A company's application development team uses Linux-based Amazon EC2 instances as bastion hosts. Inbound SSH access to the bastion hosts is restricted to specific IP addresses, as defined in the associated security groups. The company's security team wants to receive a notification if the security group rules are modified to allow SSH access from any IP address.

What should a DevOps engineer do to meet this requirement?

Options:

A- Create an Amazon EventBridge rule with a source of aws.cloudtrail and the event name AuthorizeSecurityGroupIngress. Define an Amazon Simple Notification Service (Amazon SNS) topic as the target.
B- Enable Amazon GuardDuty and check the findings for security groups in AWS Security Hub. Configure an Amazon EventBridge rule with a custom pattern that matches GuardDuty events with an output of NON_COMPLIANT. Define an Amazon Simple Notification Service (Amazon SNS) topic as the target.
C- Create an AWS Config rule by using the restricted-ssh managed rule to check whether security groups disallow unrestricted incoming SSH traffic. Configure automatic remediation to publish a message to an Amazon Simple Notification Service (Amazon SNS) topic.
D- Enable Amazon Inspector. Include the Common Vulnerabilities and Exposures-1.1 rules package to check the security groups that are associated with the bastion hosts. Configure Amazon Inspector to publish a message to an Amazon Simple Notification Service (Amazon SNS) topic.

Answer:

A

Explanation:

https://aws.amazon.com/premiumsupport/knowledge-center/monitor-security-group-changes-ec2/

Question 4

Question Type: MultipleChoice

A DevOps engineer successfully creates an Amazon Elastic Kubernetes Service (Amazon EKS) cluster that includes managed node groups. When the DevOps engineer tries to add node groups to the cluster, the cluster returns an error that states, "NodeCreationFailure: Instances failed to join the Kubernetes cluster." The DevOps engineer confirms that the EC2 worker nodes are running and that the EKS cluster is in an active state. How should the DevOps engineer troubleshoot this issue?

Options:

A- Ensure that the EKS cluster's VPC subnets do not overlap with the 172.17.0.0/16 CIDR range.
B- Use kubectl to update the kubeconfig file to use the credentials that created the cluster.

C- Run the AWSSupport-TroubleshootEKSWorkerNode runbook.
D- Create an AWS Identity and Access Management (IAM) OpenID Connect (OIDC) provider for the cluster.

Answer:

C

Question 5

Question Type: MultipleChoice

A company uses an organization in AWS Organizations with all features enabled to manage a fleet of AWS accounts. The company expects to create many new accounts for an upcoming project.

The company wants to ensure that the new accounts will not have default VPCs and that users can develop only in specific AWS Regions. The company must monitor the new accounts for compliance with the Center for Internet Security (CIS) AWS Foundations Benchmark framework.

Which combination of solutions will meet these requirements with the LEAST operational effort? (Select TWO.)

Options:

A- Activate AWS Control Tower. Configure AWS Control Tower to disable internet-accessible subnets. Set the maximum number of private subnets to zero. Configure Region denies, and ensure that users can access only the specified Regions.
B- Activate AWS Control Tower. Install Customizations for AWS Control Tower (CfCT). Develop a custom AWS CloudFormation template to delete default VPCs. Configure Region denies, and ensure that users can access only the specified Regions.
C- Write an SCP that denies access to all Regions except the specified Regions. Create an AWS Lambda function that assumes an IAM role by using the Organizations default service role in each member account to identify and delete default VPCs. Create an Amazon EventBridge rule that invokes the Lambda function when the company creates a new AWS account.
D- Activate AWS Security Hub at the organization level. Select the CIS AWS Foundations Benchmark framework, and apply the framework to the organization.
E- Activate the CIS AWS Foundations Benchmark framework on the Control Library panel in AWS Control Tower.

Answer:

B, D

Explanation:

The company's requirements span preventive governance, account baseline configuration, and continuous compliance monitoring, all with minimal operational overhead. AWS-native, organization-level services are the most efficient way to meet these goals.

To ensure that new accounts do not retain default VPCs and that development is restricted to specific Regions, AWS Control Tower is the correct foundation. However, Control Tower alone does not remove default VPCs. By installing Customizations for AWS Control Tower (CfCT), the company can automatically deploy OU-scoped CloudFormation templates during account provisioning. A simple CloudFormation template can delete default VPCs in all Regions, while Control Tower-managed Region deny guardrails (SCPs) restrict access to only approved Regions. This approach is declarative, repeatable, and tightly integrated with account creation workflows, resulting in low operational overhead.

For CIS AWS Foundations Benchmark compliance monitoring, AWS Security Hub is the purpose-built service. Enabling Security Hub at the organization level and selecting the CIS benchmark automatically evaluates all member accounts against CIS controls and continuously reports findings. This provides centralized visibility and compliance reporting without custom rule development.

Option C introduces custom Lambda automation and EventBridge logic, increasing maintenance burden. Option E is incorrect because Control Tower does not provide full CIS benchmark monitoring; it only offers related detective guardrails.

Therefore, Option B (Control Tower + CfCT) and Option D (Security Hub with CIS benchmark) together provide the most efficient, scalable, and AWS-recommended solution.

Question 6

Question Type: MultipleChoice

A company is running an application on Amazon EC2 instances in an Auto Scaling group. Recently an issue occurred that prevented EC2 instances from launching successfully and it took several hours for the support team to discover the issue. The support team wants to be notified by email whenever an EC2 instance does not start successfully.

Which action will accomplish this?

Options:

A- Add a health check to the Auto Scaling group to invoke an AWS Lambda function whenever an instance status is impaired.
B- Configure the Auto Scaling group to send a notification to an Amazon SNS topic whenever a failed instance launch occurs.

C- Create an Amazon CloudWatch alarm that invokes an AWS Lambda function when a failed Attachinstances Auto Scaling API call is made.
D- Create a status check alarm on Amazon EC2 to send a notification to an Amazon SNS topic whenever a status check fail occurs.

Answer:

B

Explanation:

https://docs.aws.amazon.com/autoscaling/ec2/userguide/ASGettingNotifications.html#auto-scaling-sns-notifications

P2P exams

Question 7

Question Type: MultipleChoice

A company is using AWS CodeDeploy to automate software deployment. The deployment must meet these requirements:

* A number of instances must be available to serve traffic during the deployment Traffic must be balanced across those instances, and the instances must automatically heal in the event of failure.

* A new fleet of instances must be launched for deploying a new revision automatically, with no manual provisioning.

* Traffic must be rerouted to the new environment to half of the new instances at a time. The deployment should succeed if traffic is rerouted to at least half of the instances; otherwise, it should fail.

* Before routing traffic to the new fleet of instances, the temporary files generated during the deployment process must be deleted.

* At the end of a successful deployment, the original instances in the deployment group must be deleted immediately to reduce costs.

How can a DevOps engineer meet these requirements?

Options:

A- Use an Application Load Balancer and an in-place deployment. Associate the Auto Scaling group with the deployment group. Use the Automatically copy Auto Scaling group option. and

use CodeDeployDefault.OneAtAtime as the deployment configuration. Instruct AWS CodeDeploy to terminate the original instances in the deployment group, and use the AllowTraffic hook within appspec.yml to delete the temporary files.
B- Use an Application Load Balancer and a blue/green deployment. Associate the Auto Scaling group and Application Load Balancer target group with the deployment group. Use the Automatically copy Auto Scaling group option, create a custom deployment configuration with minimum healthy hosts defined as 50%. and assign the configuration to the deployment group. Instruct AWS CodeDeploy to terminate the original instances in the deployment group, and use the BeforeBlockTraffic hook within appspec.yml to delete the temporary files.
C- Use an Application Load Balancer and a blue/green deployment. Associate the Auto Scaling group and the Application Load Balancer target group with the deployment group. Use the Automatically copy Auto scaling group option, and use CodeDeployDefault.HalfAtAtime as the deployment configuration. Instruct AWSCodeDeploy to terminate the original instances in the deployment group, and use the BeforeAlIowTraffic hook within appspec.yml to delete the temporary files.
D- Use an Application Load Balancer and an in-place deployment. Associate the Auto Scaling group and Application Load Balancer target group with the deployment group. Use the Automatically copy Auto Scaling group option, and use CodeDeployDefaulLAIIatOnce as a deployment configuration. Instruct AWS CodeDeploy to terminate the original instances in the deployment group, and use the BlockTraffic hook within appspec.yml to delete the temporary files.

Answer:

C

Explanation:

Step 1: Use a Blue/Green Deployment StrategyA blue/green deployment strategy is necessary to meet the requirement of launching a new fleet of instances for each deployment and ensuring availability. In a blue/green deployment, the new version (green environment) is deployed to a separate set of instances, while the old version (blue environment) remains active. After testing the new version, traffic can be gradually shifted.

Action: Use AWS CodeDeploy's blue/green deployment configuration.

Why: Blue/green deployment minimizes downtime and ensures that traffic is shifted only to healthy instances.

Step 2: Use an Application Load Balancer and Auto Scaling GroupThe Application Load Balancer (ALB) is essential to balance traffic across multiple instances, and Auto Scaling ensures the deployment scales automatically to meet demand.

Action: Associate the Auto Scaling group and Application Load Balancer target group with the deployment group.

Why: This configuration ensures that traffic is evenly distributed and that instances automatically scale based on traffic load.

Step 3: Use Custom Deployment ConfigurationThe company requires that traffic be rerouted to at least half of the instances to succeed. AWS CodeDeploy allows you to configure custom deployment settings with specific thresholds for healthy hosts.

Action: Create a custom deployment configuration where 50% of the instances must be healthy.

Why: This ensures that the deployment continues only if at least 50% of the new instances are healthy.

Step 4: Clean Temporary Files Using HooksBefore routing traffic to the new environment, the temporary files generated during the deployment must be deleted. This can be achieved using the BeforeAllowTraffic hook in the appspec.yml file.

Action: Use the BeforeAllowTraffic lifecycle event hook to clean up temporary files before routing traffic to the new environment.

Why: This ensures that the environment is clean before the new instances start serving traffic.

Step 5: Terminate Original Instances After DeploymentAfter a successful deployment, AWS CodeDeploy can automatically terminate the original instances (blue environment) to save costs.

Action: Instruct AWS CodeDeploy to terminate the original instances after the new instances are healthy.

Why: This helps in cost reduction by removing unused instances after the deployment.

This corresponds to Option C: Use an Application Load Balancer and a blue/green deployment. Associate the Auto Scaling group and the Application Load Balancer target group with the deployment group. Use the Automatically copy Auto Scaling group option, and use CodeDeployDefault.HalfAtATime as the deployment configuration. Instruct AWS CodeDeploy to terminate the original instances in the deployment group, and use the BeforeAllowTraffic hook within appspec.yml to delete the temporary files.

Question 8

Question Type: MultipleChoice

A company operates a fleet of Amazon EC2 instances that host critical applications and handle sensitive dat

a. The EC2 instances must have up-to-date security patches to protect against vulnerabilities and ensure compliance with industry standards and regulations. The company needs an automated solution to monitor and enforce security patch compliance across the EC2 fleet.

Which solution will meet these requirements?

Options:

A- Configure AWS Systems Manager Patch Manager and AWS Config with defined patch baselines and compliance rules that run Systems Manager Automation documents.
B- Access each EC2 instance by using SSH keys. Check for and apply security updates by using package managers. Verify the installations.
C- Configure Auto Scaling groups that have scaling policies based on Amazon CloudWatch metrics. Configure Auto Scaling launch templates that launch new instances by using the latest AMIs that contain new security patches.
D- Use AWS CloudFormation to recreate EC2 instances with the latest AMI every time a new patch becomes available. Use AWS CloudTrail logs to monitor patch compliance and to send alerts for non-compliant instances.

Answer:

A

Explanation:

Option A is the most correct because it provides both: (1) automated patching and (2) compliance monitoring/enforcement across a fleet, using AWS-native services built for exactly this purpose.

AWS Systems Manager Patch Manager is designed to automate patching of managed instances using patch baselines, maintenance windows (or on-demand), and it produces compliance status for patching. It's the standard AWS service to apply OS/security patches at scale without SSH'ing into instances.

AWS Config can be used to evaluate and track compliance over time against defined rules, giving centralized visibility and continuous compliance assessment. With remediation, Config can invoke Systems Manager Automation documents to correct non-compliant resources or trigger patch actions (depending on the rule/remediation design). This meets the ''monitor and enforce'' requirement.

Why the other options don't meet requirements as well:

B is manual, doesn't scale well, and increases operational risk (key management, human error). It's not ''automated monitoring and enforcement.''

C (replacing instances with new AMIs) can be part of an immutable infrastructure strategy, but by itself it does not provide compliance monitoring across the current fleet, and scaling policies based on CloudWatch metrics are unrelated to patch compliance. Also, patch cadence would depend on AMI pipelines and instance rotation rather than direct compliance enforcement.

To Get Premium Files for DOP-C02 Visit

https://www.p2pexams.com/products/dop-c02

For More Free Questions Visit

https://www.p2pexams.com/amazon/pdf/dop-c02