Free Questions for DOP-C02

Shared by Colon on 04-04-2023

For More Free Questions and Preparation Resources

Check the Links on Last Page

Question 1

Question Type: MultipleChoice

A DevOps engineer is setting up a container-based architecture. The engineer has decided to use AWS CloudFormation to automatically provision an Amazon ECS cluster and an Amazon EC2 Auto Scaling group to launch the EC2 container instances. After successfully creating the CloudFormation stack, the engineer noticed that, even though the ECS cluster and the EC2 instances were created successfully and the stack finished the creation, the EC2 instances were associating with a different cluster.

How should the DevOps engineer update the CloudFormation template to resolve this issue?

Options:

A- Reference the EC2 instances in the AWS: ECS: Cluster resource and reference the ECS cluster in the AWS: ECS: Service resource.
B- Reference the ECS cluster in the AWS: AutoScaling: LaunchConfiguration resource of the UserData property.
C- Reference the ECS cluster in the AWS:EC2: Instance resource of the UserData property.
D- Reference the ECS cluster in the AWS: CloudFormation: CustomResource resource to trigger an AWS Lambda function that registers the EC2 instances with the appropriate ECS cluster.

Answer:

B

Explanation:

The UserData property of the AWS: AutoScaling: LaunchConfiguration resource can be used to specify a script that runs when the EC2 instances are launched. This script can include the ECS cluster name as an environment variable for the ECS agent running on the EC2 instances. This way, the EC2 instances will register with the correct ECS cluster. Option A is incorrect because the AWS: ECS: Cluster resource does not have a property to reference the EC2 instances. Option C is incorrect because the EC2 instances are launched by the Auto Scaling group, not by the AWS: EC2: Instance resource. Option D is incorrect because using a custom resource and a Lambda function is unnecessary and overly complex for this scenario.Reference:AWS::AutoScaling::LaunchConfiguration,Amazon ECS Container Agent Configuration

Question 2

Question Type: MultipleChoice

A company has a single developer writing code for an automated deployment pipeline. The developer is storing source code in an Amazon S3 bucket for each project. The company wants to add more developers to the team but is concerned about code conflicts and lost work The company also wants to build a test environment to deploy newer versions of code for testing and allow developers to automatically deploy to both environments when code is changed in the repository.

What is the MOST efficient way to meet these requirements?

Options:

A- Create an AWS CodeCommit repository tor each project, use the mam branch for production code: and create a testing branch for code deployed to testing Use feature branches to develop new features and pull requests to merge code to testing and main branches.
B- Create another S3 bucket for each project for testing code, and use an AWS Lambda function to promote code changes between testing and production buckets Enable versioning on all buckets to prevent code conflicts.
C- Create an AWS CodeCommit repository for each project, and use the main branch for production and test code with different deployment pipelines for each environment Use feature branches to develop new features.
D- Enable versioning and branching on each S3 bucket, use the main branch for production code, and create a testing branch for code deployed to testing. Have developers use each branch for developing in each environment.

Answer:

A

Explanation:

Creating an AWS CodeCommit repository for each project, using the main branch for production code, and creating a testing branch for code deployed to testing will meet the requirements.AWS CodeCommit is a managed revision control service that hosts Git repositories and works with all Git-based tools1. By using feature branches to develop new features and pull requests to merge code to testing and main branches, the developers can avoid code conflicts and lost work, and also implement code reviews and approvals. Option B is incorrect because creating another S3 bucket for each project for testing code and using an AWS Lambda function to promote code changes between testing and production buckets will not provide the benefits of revision control, such as tracking changes, branching, merging, and collaborating. Option C is incorrect because

using the main branch for production and test code with different deployment pipelines for each environment will not allow the developers to test their code changes before deploying them to production. Option D is incorrect because enabling versioning and branching on each S3 bucket will not work with Git-based tools and will not provide the same level of revision control as AWS CodeCommit.Reference:

AWS CodeCommit

Certified DevOps Engineer - Professional (DOP-C02) Study Guide(page 182)

Question 3

Question Type: MultipleChoice

A company's application uses a fleet of Amazon EC2 On-Demand Instances to analyze and process dat

a. The EC2 instances are in an Auto Scaling group. The Auto Scaling group is a target group for an Application Load Balancer (ALB). The application analyzes critical data that cannot tolerate interruption. The application also analyzes noncritical data that can withstand interruption.

The critical data analysis requires quick scalability in response to real-time application demand. The noncritical data analysis involves memory consumption. A DevOps engineer must implement a solution that reduces scale-out latency for the critical data. The solution also must process the noncritical data.

Which combination of steps will meet these requirements? (Select TWO.)

Options:

A- For the critical data, modify the existing Auto Scaling group. Create a warm pool instance in the stopped state. Define the warm pool size. Create a new
B- For the critical data, modify the existing Auto Scaling group. Create a warm pool instance in the stopped state. Define the warm pool size. Create a new
C- For the critical data. modify the existing Auto Scaling group. Create a lifecycle hook to ensure that bootstrap scripts are completed successfully. Ensure that the application on the instances is ready to accept traffic before the instances are registered. Create a new version of the launch template that has detailed monitoring enabled.
D- For the noncritical data, create a second Auto Scaling group that uses a launch template. Configure the launch template to install the unified Amazon
CloudWatch agent and to configure the CloudWatch agent with a custom memory utilization metric. Use Spot Instances. Add the new Auto Scaling group as
the target group for the ALB. Modify the application to use two target groups for critical data and

noncritical data.
E- For the noncritical data, create a second Auto Scaling group. Choose the predefined memory utilization metric type for the target tracking scaling policy. Use Spot Instances. Add the new Auto Scaling group as the target group for the ALB. Modify the application to use two target groups for critical data and noncritical data.

Answer:

B, D

Explanation:

For the critical data, using a warm pool1can reduce the scale-out latency by having pre-initialized EC2 instances ready to serve the application traffic.Using On-Demand Instances can ensure that the instances are always available and not interrupted by Spot interruptions2.

For the noncritical data, using a second Auto Scaling group with Spot Instances can reduce the cost and leverage the unused capacity of EC23.Using a launch template with the CloudWatch agent4can enable the collection of memory utilization metrics, which can be used to scale the group based on the memory demand. Adding the second group as a target group for the ALB and modifying the application to use two target groups can enable routing the traffic based on the data type.

Question 4

Question Type: MultipleChoice

A company has an AWS CodeDeploy application. The application has a deployment group that uses a single tag group to identify instances for the deployment of Application

Options:

A- The single tag group configuration identifies instances that have Environment=Production and Name=ApplicattonA tags for the deployment of ApplicationA.
The company launches an additional Amazon EC2 instance with Department=Marketing Environment^Production. and Name=ApplicationB tags. On the next CodeDeploy deployment of ApplicationA. the additional instance has ApplicationA installed on it. A DevOps engineer needs to configure the existing deployment group to prevent ApplicationA from being installed on the additional instance
Which solution will meet these requirements?
A- Change the current single tag group to include only the Environment=Production tag Add

another single tag group that includes only the Name=ApplicationA tag.
B- Change the current single tag group to include the Department=Marketmg Environment=Production and Name=ApplicationAtags
C- Add another single tag group that includes only the Department=Marketing tag. Keep the Environment=Production and Name=ApplicationA tags with the current single tag group
D- Change the current single tag group to include only the Environment=Production tag Add another single tag group that includes only the Department=Marketing tag

Answer:

A, A

Explanation:

To prevent ApplicationA from being installed on the additional instance, the deployment group configuration needs to be more specific. By changing the current single tag group to include only theEnvironment=Productiontag and adding another single tag group that includes only theName=ApplicationAtag, the deployment process will target only the instances that match both tag groups. This ensures that only instances intended for ApplicationA with the correct environment and name tags will receive the deployment, thus excluding the additional instance with theDepartment=MarketingandName=ApplicationBtags.

AWS CodeDeploy Documentation:Working with instances for CodeDeploy

AWS CodeDeploy Documentation:Stop a deployment with CodeDeploy

Stack Overflow Discussion:CodeDeploy Deployment failed to stop Application

Question 5

Question Type: MultipleChoice

A company has a mission-critical application on AWS that uses automatic scaling The company wants the deployment lilecycle to meet the following parameters.

* The application must be deployed one instance at a time to ensure the remaining fleet continues to serve traffic

* The application is CPU intensive and must be closely monitored

* The deployment must automatically roll back if the CPU utilization of the deployment instance exceeds 85%.

Which solution will meet these requirements?

Options:

A- Use AWS CloudFormalion to create an AWS Step Functions state machine and Auto Scaling hfecycle hooks to move to one instance at a time into a wait state Use AWS Systems Manager automation to deploy the update to each instance and move it back into the Auto Scaling group using the heartbeat timeout
B- Use AWS CodeDeploy with Amazon EC2 Auto Scaling. Configure an alarm tied to the CPU utilization metric. Use the CodeDeployDefault OneAtAtime configuration as a deployment strategy Configure automatic rollbacks within the deployment group to roll back the deployment if the alarm thresholds are breached
C- Use AWS Elastic Beanstalk for load balancing and AWS Auto Scaling Configure an alarm tied to the CPU utilization metric Configure rolling deployments with a fixed batch size of one instance Enable enhanced health to monitor the status of the deployment and roll back based on the alarm previously created.
D- Use AWS Systems Manager to perform a blue/green deployment with Amazon EC2 Auto Scaling Configure an alarm tied to the CPU utilization metric Deploy updates one at a time Configure automatic rollbacks within the Auto Scaling group to roll back the deployment if the alarm thresholds are breached

Answer:

B

Explanation:

https://aws.amazon.com/about-aws/whats-new/2016/09/aws-codedeploy-introduces-deployment-monitoring-with-amazon-cloudwatch-alarms-and-automatic-deployment-rollback/

Question 6

Question Type: MultipleChoice

A company hired a penetration tester to simulate an internal security breach The tester performed port scans on the company's Amazon EC2 instances. The company's security measures did not detect the port scans.

The company needs a solution that automatically provides notification when port scans are performed on EC2 instances. The company creates and subscribes to an Amazon Simple Notification Service (Amazon SNS) topic.

What should the company do next to meet the requirement?

Options:

A- Ensure that Amazon GuardDuty is enabled Create an Amazon CloudWatch alarm for detected EC2 and port scan findings. Connect the alarm to the SNS topic.
B- Ensure that Amazon Inspector is enabled Create an Amazon EventBridge event for detected network reachability findings that indicate port scans Connect the event to the SNS topic.
C- Ensure that Amazon Inspector is enabled. Create an Amazon EventBridge event for detected CVEs that cause open port vulnerabilities. Connect the event to the SNS topic
D- Ensure that AWS CloudTrail is enabled Create an AWS Lambda function to analyze the CloudTrail logs for unusual amounts of traffic from an IP address range Connect the Lambda function to the SNS topic.

Answer:

A

Explanation:

* Ensure that Amazon GuardDuty is Enabled:

Amazon GuardDuty is a threat detection service that continuously monitors for malicious activity and unauthorized behavior.

It can detect port scans and generate findings for these events.

* Create an Amazon CloudWatch Alarm for Detected EC2 and Port Scan Findings:

Configure GuardDuty to monitor for port scans and other threats.

Create a CloudWatch alarm that triggers when GuardDuty detects port scan activities.

* Connect the Alarm to the SNS Topic:

The CloudWatch alarm should be configured to send notifications to the SNS topic subscribed by the security team.

This setup ensures that the security team receives near-real-time notifications when a port scan is detected on the EC2 instances.

Example configuration steps:

Enable GuardDuty and ensure it is monitoring the relevant AWS accounts.

Create a CloudWatch alarm:

{

'AlarmName': 'GuardDutyPortScanAlarm',

'MetricName': 'ThreatIntelIndicator',

'Namespace': 'AWS/GuardDuty',

'Statistic': 'Sum',

'Dimensions': [

{

'Name': 'FindingType',

'Value': 'Recon:EC2/Portscan'

}

],

'Period': 300,

'EvaluationPeriods': 1,

'Threshold': 1,

'ComparisonOperator': 'GreaterThanOrEqualToThreshold',

'AlarmActions': ['arn:aws:sns:region:account-id:SecurityAlerts']

}

Amazon GuardDuty

Creating CloudWatch Alarms for GuardDuty Findings

Question 7

Question Type: MultipleChoice

A company gives its employees limited rights to AWS DevOps engineers have the ability to assume an administrator role. For tracking purposes, the security team wants to receive a near-real-time notification when the administrator role is assumed.

How should this be accomplished?

Options:

A- Configure AWS Config to publish logs to an Amazon S3 bucket Use Amazon Athena to query

the logs and send a notification to the security team when the administrator role is assumed
B- Configure Amazon GuardDuty to monitor when the administrator role is assumed and send a notification to the security team
C- Create an Amazon EventBridge event rule using an AWS Management Console sign-in events event pattern that publishes a message to an Amazon SNS topic if the administrator role is assumed
D- Create an Amazon EventBridge events rule using an AWS API call that uses an AWS CloudTrail event pattern to invoke an AWS Lambda function that publishes a message to an Amazon SNS topic if the administrator role is assumed.

Answer:

D

Explanation:

* Create an Amazon EventBridge Rule Using an AWS CloudTrail Event Pattern:

AWS CloudTrail logs API calls made in your account, including actions performed by roles.

Create an EventBridge rule that matches CloudTrail events where the AssumeRole API call is made to assume the administrator role.

* Invoke an AWS Lambda Function:

Configure the EventBridge rule to trigger a Lambda function whenever the rule's conditions are met.

The Lambda function will handle the logic to send a notification.

* Publish a Message to an Amazon SNS Topic:

The Lambda function will publish a message to an SNS topic to notify the security team.

Subscribe the security team's email address to this SNS topic to receive real-time notifications.

Example EventBridge rule pattern:

{

'source': ['aws.cloudtrail'],

'detail-type': ['AWS API Call via CloudTrail'],

'detail': {

'eventSource': ['sts.amazonaws.com'],

'eventName': ['AssumeRole'],

'requestParameters': {

'roleArn': ['arn:aws:iam:::role/AdministratorRole']

}

}

}

Example Lambda function (Node.js) to publish to SNS:

const AWS = require('aws-sdk');

const sns = new AWS.SNS();

exports.handler = async (event) => {

const params = {

Message: `Administrator role assumed: ${JSON.stringify(event.detail)}`,

TopicArn: 'arn:aws:sns:<region>::<sns-topic>'

};

await sns.publish(params).promise();

};

Creating EventBridge Rules

Using AWS Lambda with Amazon SNS

Question 8

Question Type: MultipleChoice

A company uses AWS Organizations to manage its AWS accounts. A DevOps engineer must ensure that all users who access the AWS Management Console are authenticated through the company's corporate identity provider (IdP).

Which combination of steps will meet these requirements? (Select TWO.)

Options:

A- Use Amazon GuardDuty with a delegated administrator account. Use GuardDuty to enforce

denial of 1AM user logins
B- Use AWS 1AM Identity Center to configure identity federation with SAML 2.0.
C- Create a permissions boundary in AWS 1AM Identity Center to deny password logins for 1AM users.
D- Create 1AM groups in the Organizations management account to apply consistent permissions for all 1AM users.
E- Create an SCP in Organizations to deny password creation for 1AM users.
* Step 1: Using AWS IAM Identity Center for SAML-based Identity Federation
To ensure that all users accessing the AWS Management Console are authenticated via the corporate identity provider (IdP), the best approach is to set up identity federation with AWS IAM Identity Center (formerly AWS SSO) using SAML 2.0.
Action: Use AWS IAM Identity Center to configure identity federation with the corporate IdP that supports SAML 2.0.
Why: SAML 2.0 integration enables single sign-on (SSO) for users, allowing them to authenticate through the corporate IdP and gain access to AWS resources.

Answer:

B, E

Explanation:

This corresponds to Option B: Use AWS IAM Identity Center to configure identity federation with SAML 2.0.

* Step 2: Creating an SCP to Deny Password Logins for IAM Users To enforce that IAM users do not create passwords or access the Management Console directly without going through the corporate IdP, you can create a Service Control Policy (SCP) in AWS Organizations that denies password creation for IAM users.

Action: Create an SCP that denies password creation for IAM users.

Why: This ensures that users cannot set passwords for their IAM user accounts, forcing them to use federated access through the corporate IdP for console login.

This corresponds to Option E: Create an SCP in Organizations to deny password creation for IAM users.