aws training and certification

AWS Certified DevOps Engineer – Professional (DOP-C01) Exam Guide

Introduction

The AWS Certified DevOps Engineer – Professional (DOP-C01) exam is intended for individuals who perform a DevOps engineer role. The exam validates a candidate's technical expertise in provisioning, operating, and managing distributed application systems on the AWS platform.

The exam also validates a candidate's ability to complete the following tasks:

- Implement and manage continuous delivery systems and methodologies on AWS
- Implement and automate security controls, governance processes, and compliance validation
- Define and deploy monitoring, metrics, and logging systems on AWS
- Implement systems that are highly available, scalable, and self-healing on the AWS platform
- Design, manage, and maintain tools to automate operational processes

Target candidate description

The target candidate is expected to have 2 or more years of experience provisioning, operating, and managing AWS environments. The target candidate must also have experience developing code in at least one high-level programming language.

Recommended AWS knowledge

The target candidate should have the following knowledge:

- Experience building highly automated infrastructures
- Experience administering operating systems
- An understanding of modern development and operations processes and methodologies

What is considered out of scope for the target candidate?

The following is a non-exhaustive list of related job tasks that the target candidate is not expected to be able to perform. These items are considered out of scope for the exam:

- Advanced networking (advanced routing algorithms, failover techniques)
- Deep-level security recommendations for developers
- Database query and performance optimization
- Full-stack application code development
- Normalization of data schemes

For a detailed list of specific tools and technologies that might be covered on the exam, as well as lists of in-scope AWS services, refer to the Appendix.

Exam content

Response types

There are two types of questions on the exam:

- **Multiple choice:** Has one correct response and three incorrect responses (distractors)
- **Multiple response:** Has two or more correct responses out of five or more response options

Select one or more responses that best complete the statement or answer the question. Distractors, or incorrect answers, are response options that a candidate with incomplete knowledge or skill might choose. Distractors are generally plausible responses that match the content area.

Unanswered questions are scored as incorrect; there is no penalty for guessing. The exam includes 65 questions that will affect your score.

Unscored content

The exam includes 10 unscored questions that do not affect your score. AWS collects information about candidate performance on these unscored questions to evaluate these questions for future use as scored questions. These unscored questions are not identified on the exam.

Exam results

The AWS Certified DevOps Engineer – Professional (DOP-C01) exam is a pass or fail exam. The exam is scored against a minimum standard established by AWS professionals who follow certification industry best practices and guidelines.

Your results for the exam are reported as a scaled score of 100–1,000. The minimum passing score is 750. Your score shows how you performed on the exam as a whole and whether or not you passed. Scaled scoring models help equate scores across multiple exam forms that might have slightly different difficulty levels.

Your score report may contain a table of classifications of your performance at each section level. This information is intended to provide general feedback about your exam performance. The exam uses a compensatory scoring model, which means that you do not need to achieve a passing score in each section. You need to pass only the overall exam.

Each section of the exam has a specific weighting, so some sections have more questions than other sections have. The table contains general information that highlights your strengths and weaknesses. Use caution when interpreting section-level feedback.

Content outline

This exam guide includes weightings, test domains, and objectives for the exam. It is not a comprehensive listing of the content on the exam. However, additional context for each of the objectives is available to help guide your preparation for the exam. The following table lists the main content domains and their weightings. The table precedes the complete exam content outline, which includes the additional context. The percentage in each domain represents only scored content.

Domain	% of Exam
Domain 1: SDLC Automation	22%
Domain 2: Configuration Management and Infrastructure as Code	19%
Domain 3: Monitoring and Logging	15%
Domain 4: Policies and Standards Automation	10%
Domain 5: Incident and Event Response	18%
Domain 6: High Availability, Fault Tolerance, and Disaster Recover	16%
TOTAL	**100%**

Domain 1: SDLC Automation

1.1 Apply concepts required to automate a CI/CD pipeline
- Set up repositories
- Set up build services
- Integrate automated testing (e.g., unit tests, integrity tests)
- Set up deployment products/services
- Orchestrate multiple pipeline stages

1.2 Determine source control strategies and how to implement them
- Determine a workflow for integrating code changes from multiple contributors
- Assess security requirements and recommend code repository access design
- Reconcile running application versions to repository versions (tags)
- Differentiate different source control types

1.3 Apply concepts required to automate and integrate testing
- Run integration tests as part of code merge process
- Run load/stress testing and benchmark applications at scale
- Measure application health based on application exit codes (robust Health Check)
- Automate unit tests to check pass/fail, code coverage
    - CodePipeline, CodeBuild, etc.
- Integrate tests with pipeline

1.4 Apply concepts required to build and manage artifacts securely
- Distinguish storage options based on artifacts security classification
- Translate application requirements into Operating System and package configuration (build specs)
- Determine the code/environment dependencies and required resources
    - Example: CodeDeploy AppSpec, CodeBuild buildspec
- Run a code build process

1.5 Determine deployment/delivery strategies (e.g., A/B, Blue/green, Canary, Red/black) and how to implement them using AWS services
- Determine the correct delivery strategy based on business needs
- Critique existing deployment strategies and suggest improvements
- Recommend DNS/routing strategies (e.g., Route 53, ELB, ALB, load balancer) based on business continuity goals
- Verify deployment success/failure and automate rollbacks

Domain 2: Configuration Management and Infrastructure as Code

2.1 Determine deployment services based on deployment needs
- Demonstrate knowledge of process flows of deployment models
- Given a specific deployment model, classify and implement relevant AWS services to meet requirements
    - Given the requirement to have DynamoDB choose CloudFormation instead of OpsWorks
    - Determine what to do with rolling updates

2.2 Determine application and infrastructure deployment models based on business needs
- Balance different considerations (cost, availability, time to recovery) based on business requirements to choose the best deployment model
- Determine a deployment model given specific AWS services
- Analyze risks associated with deployment models and relevant remedies

2.3 Apply security concepts in the automation of resource provisioning
- Choose the best automation tool given requirements
- Demonstrate knowledge of security best practices for resource provisioning (e.g., encrypting data bags, generating credentials on the fly)
- Review IAM policies and assess if sufficient but least privilege is granted for all lifecycle stages of a deployment (e.g., create, update, promote)
- Review credential management solutions (e.g., EC2 parameter store, third party)
- Build the automation
    - CloudFormation template, Chef Recipe, Cookbooks, Code pipeline, etc.

2.4 Determine how to implement lifecycle hooks on a deployment
- Determine appropriate integration techniques to meet project requirements
- Choose the appropriate hook solution (e.g., implement leader node selection after a node failure) in an Auto Scaling group
- Evaluate hook implementation for failure impacts (if a remote call fails, if a dependent service is temporarily unavailable (i.e., Amazon S3), and recommend resiliency improvements
- Evaluate deployment rollout procedures for failure impacts and evaluate rollback/recovery processes

2.5 Apply concepts required to manage systems using AWS configuration management tools and services
- Identify pros and cons of AWS configuration management tools
- Demonstrate knowledge of configuration management components
- Show the ability to run configuration management services end to end with no assistance while adhering to industry best practices

Domain 3: Monitoring and Logging

3.1 Determine how to set up the aggregation, storage, and analysis of logs and metrics
- Implement and configure distributed logs collection and processing (e.g., agents, syslog, flumed, CW agent)
- Aggregate logs (e.g., Amazon S3, CW Logs, intermediate systems (EMR), Kinesis FH – Transformation, ELK/BI)
- Implement custom CW metrics, Log subscription filters
- Manage Log storage lifecycle (e.g., CW to S3, S3 lifecycle, S3 events)

3.2 Apply concepts required to automate monitoring and event management of an environment
- Parse logs (e.g., Amazon S3 data events/event logs/ELB/ALB/CF access logs) and correlate with other alarms/events (e.g., CW events to AWS Lambda) and take appropriate action
- Use CloudTrail/VPC flow logs for detective control (e.g., CT, CW log filters, Athena, NACL or WAF rules) and take dependent actions (AWS step) based on error handling logic (state machine)
- Configure and implement Patch/inventory/state management using ESM (SSM), Inspector, CodeDeploy, OpsWorks, and CW agents
    - EC2 retirement/maintenance
- Handle scaling/failover events (e.g., ASG, DB HA, route table/DNS update, Application Config, Auto Recovery, PH dashboard, TA)
- Determine how to automate the creation of monitoring

3.3 Apply concepts required to audit, log, and monitor operating systems, infrastructures, and applications
- Monitor end to end service metrics (DDB/S3) using available AWS tools (X-ray with EB and Lambda)
- Verify environment/OS state through auditing (Inspector), Config rules, CloudTrail (process and action), and AWS APIs
- Enable, configure, and analyze custom metrics (e.g., Application metrics, memory, KCL/KPL) and take action
- Ensure container monitoring (e.g., task state, placement, logging, port mapping, LB)
- Distinguish between services that enable service level or OS level monitoring
    - Example: AWS services that use OS agents (e.g., Inspector, SSM)

3.4 Determine how to implement tagging and other metadata strategies
- Segregate authority based on tagging (lifecycle stages – dev/prod) with Condition context keys
- Utilize Amazon S3 system/user-defined metadata for classification and automation
- Design and implement tag-based deployment groups with CodeDeploy
- Best practice for cost allocation/optimization with tagging

Domain 4: Policies and Standards Automation

4.1 Apply concepts required to enforce standards for logging, metrics, monitoring, testing, and security
- Detect, report, and respond to governance and security violations
- Apply logging standards across application, operating system, and infrastructure
- Apply context specific application health and performance monitoring
- Outline standards for delivery models for logs and metrics (e.g., JSON, XML, Data Normalization)

4.2 Determine how to optimize cost through automation
- Prioritize automation effort to reduce labor costs
- Implement right sizing of workload based on metrics
- Assess ways to improve time to market through automating process orchestration and repeatable tasks
- Diagnose outliers to determine use case fit
    - Example: Configuration drift
- Measure and automate cost optimization through events
    - Example: Trusted Advisor

4.3 Apply concepts required to implement governance strategies
- Generalize governance standards across CI/CD pipeline
- Outline and measure the real-time status of compliance with governance strategies
- Report on compliance with governance strategies
- Deploy governance policies related to self-service capabilities
    - Example: Service Catalog, CFN Nag

Domain 5: Incident and Event Response

5.1 Troubleshoot issues and determine how to restore operations
- Given an issue, evaluate how to narrow down the unhealthy components as quickly as possible
- Given an increase in load, determine what steps to take to mitigate the impact
- Determine the causes and impacts of a failure
    - Example: Deployment, operations
- Determine the best way to restore operations after a failure occurs
- Investigate and correlate logged events with application components
    - Example: application source code

5.2 Determine how to automate event management and alerting
- Set up automated restores from backup in the event of a catastrophic failure
- Set up methods to deliver alerts and notifications that are appropriate for different types of events
- Assess the quality/actionability of alerts
- Configure metrics appropriate to an application's SLAs
- Proactively update limits

5.3 Apply concepts required to implement automated healing
- Set up the correct scaling strategy to enable auto-healing when a failure occurs (e.g., with Auto Scaling policies)
- Use the correct rollback strategy to avoid impact from failed deployments
- Configure Route 53 to ensure cross-Region failover
- Detect and respond to maintenance or Spot termination events

5.4 Apply concepts required to set up event-driven automated actions
- Configure Lambda functions or CloudWatch actions to implement automated actions
- Set up CloudWatch event rules and/or Config rules and targets
- Use AWS Systems Manager or Step Functions to coordinate components (e.g., Lambda, use maintenance windows)
- Configure a build/roll-out process to automatically respond to critical software updates

Domain 6: High Availability, Fault Tolerance, and Disaster Recovery

6.1 Determine appropriate use of multi-AZ versus multi-Region architectures
- Determine deployment strategy based on HA/DR requirements
- Determine data replication strategy based on cost and durability requirements
- Determine infrastructure, platform, and services based on HA/DR requirements
- Design for HA/FT/DR based on service availability (i.e., global/regional/single AZ)

6.2 Determine how to implement high availability, scalability, and fault tolerance
- Design deployment strategy to support HA/FT/scalability
- Assess statefulness of application infrastructure components
- Use load balancing to distribute traffic across multiple AZ/ASGs/instance types (spot/M4 vs C4) /targets
- Use appropriate caching solutions to improve availability and performance

6.3 Determine the right services based on business needs (e.g., RTO/RPO, cost)
- Determine cost-effective storage solution for your application
  - Example: tiered, archival, EBS type, hot/cold
- Choose a database platform and configuration to meet business requirements
- Choose a cost-effective compute platform based on business requirements
  - Example: Spot
- Choose a deployment service/model based on business requirements
  - Example: Code Deploy, Blue/Green deployment
- Determine when to use managed service vs. self-managed infrastructure (Docker on EC2 vs. ECS)

6.4 Determine how to design and automate disaster recovery strategies
- Automate failure detection
- Automate components/environment recovery
- Choose appropriate deployment strategy for environment recovery
- Design automation to support failover in hybrid environment

6.5 Evaluate a deployment for points of failure
- Determine appropriate deployment-specific health checks
- Implement failure detection during deployment
- Implement failure event handling/response
- Ensure that resources/components/processes exist to react to failures during deployment
- Look for exit codes on each event of the deployment
- Map errors to different points of deployment

Appendix

Which key tools, technologies, and concepts might be covered on the exam?

The following is a non-exhaustive list of the tools and technologies that could appear on the exam. This list is subject to change and is provided to help you understand the general scope of services, features, or technologies on the exam. The general tools and technologies in this list appear in no particular order. AWS services are grouped according to their primary functions. While some of these technologies will likely be covered more than others on the exam, the order and placement of them in this list is no indication of relative weight or importance:

- Application deployment
- Application integration
- Application pipelines
- Automation
- Code repository best practices
- Cost optimization
- Deployment requirements
- Hybrid deployments
- IAM policies
- Metrics, monitoring, alarms, and logging
- Network ACL and security group design and implementation
- Operational best practices
- Rollback procedures

AWS services and features

Analytics:

- Amazon Athena
- Amazon EMR
- Amazon Kinesis Data Firehose
- Amazon Kinesis Data Streams
- Amazon QuickSight

Compute:

- Amazon EC2
- Amazon EC2 Auto Scaling

Containers:

- AWS App Runner
- Amazon Elastic Container Registry (Amazon ECR)
- Amazon Elastic Container Service (Amazon ECS)
- Amazon Elastic Kubernetes Service (Amazon EKS)
- AWS Fargate

Database:

- Amazon DynamoDB
- Amazon RDS
- Amazon Redshift

Developer Tools:

- AWS Cloud Development Kit (AWS CDK)
- AWS CloudShell
- AWS CodeArtifact
- AWS CodeBuild
- AWS CodeCommit
- AWS CodeDeploy
- Amazon CodeGuru
- AWS CodePipeline
- AWS CodeStar
- AWS Command Line Interface (CLI)
- AWS X-Ray

Management and Governance:

- AWS CloudFormation
- AWS CloudTrail
- Amazon CloudWatch
- AWS Config
- AWS OpsWorks
- AWS Organizations
- AWS Systems Manager
- AWS Trusted Advisor

Networking and Content Delivery:

- Amazon API Gateway
- AWS Client VPN
- Amazon CloudFront
- Amazon Route 53
- AWS Site-to-Site VPN
- AWS Transit Gateway
- Amazon VPC
- Elastic Load Balancing

Security, Identity, and Compliance:

- Amazon GuardDuty
- AWS Identity and Access Management (IAM)
- Amazon Inspector
- AWS Key Management Service (AWS KMS)
- AWS Secrets Manager
- AWS Single Sign-On
- AWS WAF

Serverless:

- Amazon EventBridge (Amazon CloudWatch Events)
- AWS Lambda
- AWS Serverless Application Model (AWS SAM)
- Amazon Simple Notification Service (Amazon SNS)
- Amazon Simple Queue Service (Amazon SQS)
- AWS Step Functions

Storage:

- Amazon Elastic Block Store (Amazon EBS)
- Amazon Elastic File System (Amazon EFS)
- Amazon S3
- AWS Storage Gateway

1) A company controls the source code for its product in AWS CodeCommit. The company is creating a CI/CD pipeline for the product using AWS CodePipeline. The pipeline must automatically start on changes to the master branch of the CodeCommit repository. Changes are made to the application every day, so the pipeline needs to be as responsive as possible.

Which actions should the devops engineer take to meet these requirements?

A) Configure the pipeline to periodically check the repository. Start the pipeline when changes are detected.
B) Configure the repository to generate an Amazon CloudWatch Events event upon changes. Configure the pipeline to start in response to the event.
C) Configure the repository to periodically run an AWS Lambda function. The function should check the repository and start the pipeline when changes are detected.
D) Configure the repository to publish an SNS notification upon changes. Subscribe the pipeline to the Amazon SNS topic.

2) A development team wants to set up an AWS CodeCommit repository. Developers should be able push changes to their own branches, but they should not be allowed to push commits or merge pull requests into the master branch. Additionally, whenever a commit or merge occurs into the master branch, the project manager needs to receive a notification.

Which combination of steps will protect the master branch and send the alert with the shortest delay? (Select TWO.)

A) Attach an AWS IAM policy to the developer IAM group that denies the actions of pushing commits, merging pull requests, and adding files to the master branch.
B) Attach a resource policy to the CodeCommit repository that denies members of the IAM developer group the actions of pushing commits, merging pull requests, and adding files to the master branch.
C) Set up a an AWS Lambda function that runs every 15 minutes to check for repository changes and publishes a notification to an Amazon SNS topic.
D) Set up an Amazon CloudWatch Events rule triggered by a `CodeCommit Repository State Change` event for the master branch and add an Amazon SNS topic as a target.
E) Configure AWS CloudTrail to send log events to Amazon CloudWatch Logs. Define a metric filter to identify repository events. Create a CloudWatch alarm with an Amazon SNS topic as a target.

3) A company is using AWS CodeBuild to build its application. Company policy requires that all build artifacts be encrypted at rest. Access to the artifacts must be limited to IAM users with permission to assume the operations role.

How can these requirements be met?

A) Add a post-build command to the CodeBuild build specification that pushes build objects to an Amazon S3 bucket. Set a bucket policy that prevents upload to the bucket unless the request includes the header `x-amz-server-side-encryption`. Add a `Deny` statement for all actions with the `NotPrincipal` section referencing the operations IAM group.
B) Add a post-build command to the CodeBuild build specification that pushes build objects to an Amazon S3 bucket. Configure an S3 event notification to trigger an AWS Lambda function to get the object, encrypt it, then put it back into the S3 bucket with an `encrypted` tag key and a `true` tag value. Add an S3 bucket policy with a `Deny` statement for all actions with the `NotPrincipal` section referencing the operations IAM group, and a `Condition` section referencing the `Encrypted` tag.
C) Add a post-build command to the CodeBuild build specification that pushes build objects to an Amazon S3 bucket that has S3 default encryption enabled. Set an S3 bucket policy containing a `Deny` statement for all actions with the `NotPrincipal` section referencing the operations IAM role.
D) Add a post-build command to the CodeBuild build specification that calls the AWS KMS `Encrypt` API call, passing the artifact to AWS KMS for encryption with a specified customer master key (CMK). Push the encrypted artifact to an Amazon S3 bucket, then set up the IAM operations group as the only key user for that CMK in AWS KMS.

4) A devops engineer wants to implement a blue/green deployment process for an application on AWS and be able to gradually shift the traffic between the environments. The application runs on Amazon EC2 instances behind an Application Load Balancer. The instances run in an EC2 Auto Scaling group. Data is stored in an Amazon RDS Multi-AZ DB instance. External DNS is provided by Amazon Route 53.

Which combination of steps will implement the blue/green process? (Select THREE.)

A) Create a second Auto Scaling group behind the same Application Load Balancer.
B) Create a second Application Load Balancer and Auto Scaling group.
C) Create a second alias record in Route 53 pointing to the new environment and use a failover routing policy between the two records.
D) Create a second alias record in Route 53 pointing to the new environment and use a weighted routing policy between the two records.
E) Configure the new EC2 instances to use the same RDS database instance.
F) Configure the new EC2 instances to use the failover node of the RDS database instance.

5) A devops engineer wrote an AWS Lambda function, defined it in an AWS CloudFormation template snippet (shown below), and stored it in an Amazon S3 bucket.

```
MyLambdaFunctionV1:
  Type: "AWS::Lambda::Function"
  Properties:
    Handler: "index.handler"
    Role: "arn:aws:iam::515290864834:role/AccountScanner"
    Code:
      S3Bucket: "johndoe-com-lambda-source"
      S3Key: "AccountScanner.zip"
    Runtime: "dotnetcore2.1"
    Timeout: 60
```

The CloudFormation stack has been created and the Lambda function is working as expected. The Engineer has obtained a new version of the function code and wants to ensure that this new version will be executed immediately following the stack update.

Which deployment procedures will accomplish this? (Select THREE.)

A) Update the logical name of the Lambda function in the CloudFormation template from `MyLambdaFunctionV1` to `MyLambdaFunctionV2`, then perform a CloudFormation stack update.

B) Enable versioning on the existing S3 bucket. Upload the new code to the existing S3 bucket. Specify the version ID of the S3 object in the `S3ObjectVersion` property of the Lambda function in the CloudFormation template, then perform a CloudFormation stack update.

C) Using AWS SAM, issue a `sam deploy` command to the CloudFormation template to perform a Lambda function version update.

D) Update the S3 bucket property of the Lambda function in the CloudFormation template to point to a different bucket location. Upload the new code to the new S3 bucket location, then perform a CloudFormation stack update.

E) Update the `S3Key` property of the Lambda function in the CloudFormation template to indicate a different location and name of the .zip file. Upload the new code to the S3 bucket, noting the location and name change of the .zip file, then perform a CloudFormation stack update.

F) Using the serverless framework, issue a `serverless deploy function -f MyLambdaFunctionV1` command to perform an update to the existing Lambda function.

6) A devops engineer has been asked to automate security compliance for a company. The company has developed custom AWS Config rules to detect non-compliant security configurations. When compliance issues are detected, the company wants issues to be automatically remediated and the security team to be notified over the internal security message channel. The message board has a REST interface that publishes the body of HTTPS POST requests over the channel.

Which combination of steps would successfully meet these requirements in the MOST cost-effective way? (Select THREE.)

A) Create an Amazon CloudWatch Events rule that publishes configuration item change notifications to an Amazon SNS topic.
B) Create an Amazon CloudWatch Events rule that publishes compliance change notifications to an Amazon SNS topic.
C) Configure AWS Config to publish configuration item change notifications to an Amazon SNS topic.
D) Create an Amazon API Gateway RESTful API with AWS integration to AWS Config. Subscribe the API to the Amazon SNS topic.
E) Subscribe the message channel HTTPS endpoint to the Amazon SNS topic.
F) Write an AWS Lambda function that addresses the non-compliant security configuration. Subscribe the function to the Amazon SNS topic.

7) A company runs an application on Amazon EC2 instances running the latest version of the Amazon Linux AMI. When applying new security patches, Server administrators manually remove affected instances from service, patch them, and then place them back into service. A new company security policy requires that security patches be applied within 7 days of the patch being released. The security team must verify that all instances are in compliance. Patching should be done during a time that has the least impact on users.

How can administrators automate security policy compliance?

A) Configure an AWS CodeBuild project to download and apply patches to all machines over SSH. Use an Amazon CloudWatch Events scheduled event to run the CodeBuild project during a maintenance window.
B) Use AWS Systems Manager Patch Manager to create a patch baseline. Create a script on the EC2 instances that uses the CLI to pull the latest patches from Patch Manager. Create a cron job to schedule the script to run during a maintenance window.
C) Create a script that applies any available security patches. Create a cron job to schedule the script to run during a maintenance window. Install the script and cron job on the application AMI and redeploy the application.
D) Enlist all application EC2 instances in a patch group. Use AWS Systems Manager Patch Manager to create a patch baseline. Configure a maintenance window to apply the patch baseline.

8) An operator is managing a legacy application on AWS. The application is a monolithic Microsoft Windows program running on a single Amazon EC2 instance. The source code for the application is not available, so the application cannot be modified. The application has a memory leak and malfunctions when memory utilization on the instance goes above 90%. The operator has configured the uniform Amazon CloudWatch agent on the EC2 instance to collect the memory utilization Performance Monitor counter.

Which actions should the operator take to prevent the application from malfunctioning? (Select TWO.)

A) Create an Amazon CloudWatch Events event that publishes to an Amazon SNS topic when memory utilization goes above 80%.
B) Create a metric filter on memory utilization in Amazon CloudWatch Logs. Create a CloudWatch alarm on the memory utilization filter that publishes to an Amazon SNS topic when the memory utilization goes above 80%.
C) Create a CloudWatch alarm on the memory utilization metric that publishes to an Amazon SNS topic when the memory utilization goes above 80%.
D) Subscribe an Amazon Lambda function to the Amazon SNS topic that restarts the application with an AWS Systems Manager Run Command.
E) Subscribe the EC2 instance to the Amazon SNS topic and run a script that restarts the application.

9) A company is migrating more than 100 internal applications to AWS. The applications are independent, but all use similar corporate standard architectures. Key areas of the architectures that vary are:

- **Some applications have both web and application tiers, while others just have a web tier.**
- **If there is a database, it is MySQL, SQL Server or PostgreSQL. (The company plans to manage all databases with Amazon RDS.)**
- **Some applications are built on a LAMP stack, while others are built on a .NET stack.**

The devops team wants to enable each application team to launch the infrastructure to deploy their own application. At the same time, the devops team wants to limit each team's ability to launch infrastructure outside of the corporate standard.

Which approach will allow the teams to launch the infrastructure for their applications with the minimum privileges?

A) Create two AWS Service Catalog products: one that creates a two-tier architecture and one that creates a three-tier architecture. Pass in the technology stack and the database technology as parameters. Grant the application teams the rights needed to launch the products.
B) Create two AWS CloudFormation templates: one that creates a two-tier architecture and one that creates a three-tier architecture. Pass in the technology stack and the database technology as parameters. Grant the application teams the rights needed to create the CloudFormation stacks.
C) Create an AWS CloudFormation template that launches an AWS Elastic Beanstalk web server environment application. Pass in the number of tiers, the technology stack, and the database technology as parameters. Grant the application teams the rights needed to create the CloudFormation stacks.
D) Create an AWS Service Catalog product that launches an AWS Elastic Beanstalk web server environment application. Pass in the number of tiers, the technology stack, and the database technology as parameters. Grant the application teams the rights needed to launch the product.

10) A company is designing a cross-region disaster recovery solution for an Amazon RDS PostgreSQL Multi-AZ DB instance. The disaster recovery solution requires an RPO of 4 hours and an RTO of 2 hours.

Which solution meets the requirements in the MOST cost-effective manner?

A) Create an AWS Lambda function that creates an RDS snapshot and copies it to another region. Create an Amazon CloudWatch Events scheduled event to trigger the Lambda function every 4 hours. Create an RDS notification event to publish an Amazon SNS message for database availability events. Subscribe a Lambda function to the SNS topic that will restore the snapshot to a new instance in the disaster recovery region, and update the connection string for the application.

B) Create an AWS Lambda function that generates a SQL dump file and saves it in an Amazon S3 bucket in another region. Create an Amazon CloudWatch Events scheduled event to trigger the Lambda function every 4 hours. Create an RDS notification event to publish an Amazon SNS message for database availability events. Subscribe a Lambda function to the SNS topic that will launch a new database instance, execute the SQL dump file, and update the connection string for the application.

C) Create an AWS Lambda function that copies the latest automated snapshot to another region. Create an Amazon CloudWatch Events scheduled event to trigger the Lambda function every 4 hours. Create an RDS notification event to publish an Amazon SNS message for database availability events. Subscribe a Lambda function to the SNS topic that will restore the snapshot to a new instance in the disaster recovery region, and update the connection string for the application.

D) Configure a read replica for the database instance in a different region. Create an RDS notification event to publish an Amazon SNS message for database availability events. Create an AWS Lambda function that will promote the read replica and update the connection string for the application. Subscribe the Lambda function to the SNS topic.

Answers

1) B – This is the most responsive of the given options as it is deterministic; the change will directly fire the event and the event will directly trigger the pipeline. While the periodic checks described in A will work, they are non-deterministic as they will not launch the pipeline until the next periodic check occurs. B is also the recommended solution. C is not a feature that CodeCommit supports. D is not a valid method to start the pipeline.

2) A, D – CodeCommit uses IAM policies to grant and deny access privileges for a repository. CloudWatch Events provides a near-real-time stream of CodeCommit events including repository state changes. CloudWatch Events rules can trigger on events matching a pattern and send a notification to an SNS topic. B is incorrect because CodeCommit supports IAM policies only, and not resource policies. C is incorrect because it could take up to 15 minutes for the Lambda function to detect the event. E is incorrect because CloudTrail logs can take up to 15 minutes to record an event.

3) C – S3 default encryption ensures that the artifacts are encrypted at rest. The `Deny` statement with `NotPrincipal` set to the operations role will deny access to the bucket except for requests using the role. It is implied in the stem that the operations role would have a permissions policy that allows access to the bucket. A and B are incorrect because the bucket policy is referencing the IAM group and not the role. A is also incorrect because AWS recommends using default encryption over a bucket policy to enforce encryption. B also allows artifacts to be stored at rest briefly without encryption. D is incorrect because AWS KMS can encrypt data only up to 4 KB in size.

4) B, D, E – In a blue/green deployment, two separate environments are stood up with the blue environment containing EC2 instances in an Auto Scaling group running the current production version of the application, and the green environment containing another set of EC2 instances in an Auto Scaling group running the new version of the application. Each Auto Scaling group would be behind their own Application Load Balancer (ALB), so you could configure two alias records as endpoints in Route 53 and use a weighted routing policy to gradually shift traffic from the ALB for the blue environment to the green. Unless there are required schema changes for the new release, it is best to point both environments to the same database so the data remains consistent during the cutover. A is incorrect because two ALBs as endpoints are needed to gradually shift the traffic using Route 53. C is incorrect because a failover routing policy sends all traffic to a single endpoint unless there is a failure detected by a health check, so it cannot be used to gradually shift the traffic. F is incorrect because the secondary instance in a multi-AZ RDS is a hot standby and not available for reads or writes.

5) B, D, E – The key to this item is that there must be some indication to CloudFormation in the template that the source file on S3 has changed, as CloudFormation stores neither the timestamp of the source file, nor any sort of checksum. The keys all make a change to the template, either with versions (B), code location (D) or object name (E). C and F would not work unless there was also a significant rewrite of the template to make it a SAM template (SAM is an extension of standard CloudFormation templates) or a serverless.yml file. A would upload the new code, but as an entirely new function with a new ARN and new function name, requiring additional edits to the rest of the template, and also breaking any resources outside the template that are dependent upon the function.

6) B, E, F – There are two parts to the solution: telling the world about the non-compliant configuration, then configuring the SNS fan-out to accomplish all the resulting requirements. B is the way to send the correct notification about non-compliance. A and C will send notifications for any configuration change regardless of compliance status; this would force recipients to do extra work deciding whether each message was important. Accomplishing the resulting requirements is achieved by using multiple SNS endpoints. E uses SNS HTTPS endpoints, delivering the message in the body of a POST request. F uses an SNS Lambda endpoint, triggering an AWS Lambda function from the SNS message. D does not achieve any desired result, as it just sends the message back to the AWS Config service.

7) D – Patch Manager will automatically apply security patches during a maintenance window according to a list of approved patches that you define in a patch baseline. The security team can view the patch compliance of the instances in the Systems Manager console or pull a summary using the CLI. A is incorrect because CodeBuild builds your source code into artifacts. It does not deploy patches to instances. B is incorrect because AWS Systems Manager Agent does not need to be scheduled to pull the patches. You only need to associate the patching configuration with a Systems Manager maintenance window. C is incorrect because it does not include a way for the security team to verify patch compliance, and it includes a single point of failure in the cron job.

8) C, D – There are two parts to this question. First is how does the unified CloudWatch agent publish system-level metrics? These are published as CloudWatch metrics that can be used directly for alarms like any other metric, so C is correct. The agent publishes log files from the EC2 instance to CloudWatch Logs, so B is incorrect. CloudWatch Events is a different feature of CloudWatch that fires events upon system events or on a schedule, so A is incorrect. The second part of the question is how to take action in response to an SNS message. An EC2 instance cannot subscribe to an SNS message, so E is incorrect. A Lambda function can subscribe to an SNS message, so D is correct.

9) A – AWS Service Catalog allows Administrators to publish products and grant IAM users privileges to launch the products *without* granting those users the ability to launch the underlying services. To launch a CloudFormation stack, the user needs privileges to launch all the underlying infrastructure in the stack. While there is a feature that allows privileges to be granted to CloudFormation directly with an IAM service role, the responses here clearly state the privileges are granted directly to the application teams. Elastic Beanstalk web server environments allow for a single web tier, not a web tier and application tier.

10) A – This solution meets the RPO requirements by taking manual snapshots of the standby instance and copying them to a different region. RDS supports notification events that can be published to an SNS topic. The Lambda function will restore the snapshot to a new database instance, so the DNS name will need to be updated in the connection string for the application. B will work, but the pg_dump process will use significant I/O on the primary instance, whereas RDS snapshots are taken against the secondary instance. Also, SQL dump files are very large for a large database, so creating a new instance and then executing the SQL commands in the dump file could likely exceed the RTO of 2 hours. C is incorrect because automated snapshots are created once per day only, so they do not meet the RPO requirement. D will work, but cross-region read replication is too costly for the requirements since the RPO is only 4 hours.