Question 1

A company has a serverless application comprised of Amazon CloudFront, Amazon API Gateway, and AWS Lambda functions. The current deployment process of the application code is to create a new version number of the Lambda function and run an AWS CLI script to update. If the new function version has errors, another CLI script reverts by deploying the previous working version of the function. The company would like to decrease the time to deploy new versions of the application logic provided by the Lambda functions, and also reduce the time to detect and revert when errors are identified.

How can this be accomplished?

Options:

A- Create and deploy nested AWS CloudFormation stacks with the parent stack consisting of the AWS CloudFront distribution and API Gateway, and the child stack containing the Lambda function. For changes to Lambda, create an AWS CloudFormation change set and deploy; if errors are triggered, revert the AWS CloudFormation change set to the previous version.

B- Use AWS SAM and built-in AWS CodeDeploy to deploy the new Lambda version, gradually shift traffic to the new version, and use pre-traffic and post-traffic test functions to verify code. Rollback if Amazon CloudWatch alarms are triggered.

C- Refactor the AWS CLI scripts into a single script that deploys the new Lambda version. When deployment is completed, the script tests execute. If errors are detected, revert to the previous Lambda version.

D- Create and deploy an AWS CloudFormation stack that consists of a new API Gateway endpoint that references the new Lambda version. Change the CloudFront origin to the new API Gateway endpoint, monitor errors and if detected, change the AWS CloudFront origin to the previous API Gateway endpoint.

Answer:

B

Explanation:

https://aws.amazon.com/about-aws/whats-new/2017/11/aws-lambda-supports-traffic-shifting-and-phased-deployments-with-aws-codedeploy/

Question 2

Question Type: **MultipleChoice**

A company is planning to store a large number of archived documents and make the documents available to employees through the corporate intranet. Employees will access the system by connecting through a client VPN service that is attached to a VPC. The data must not be accessible to the public.

The documents that the company is storing are copies of data that is held on physical media elsewhere. The number of requests will be low. Availability and speed of retrieval are not concerns of the company.

Which solution will meet these requirements at the LOWEST cost?

Options:

A- Create an Amazon S3 bucket. Configure the S3 bucket to use the S3 One Zone-Infrequent Access (S3 One Zone-IA) storage class as default. Configure the S3 bucket for website hosting. Create an S3 interface endpoint. Configure the S3 bucket to allow access only through that endpoint.

B- Launch an Amazon EC2 instance that runs a web server. Attach an Amazon Elastic File System (Amazon EFS) file system to store the archived data in the EFS One Zone-Infrequent Access (EFS One Zone-IA) storage class Configure the instance security groups to allow access only from private networks.

C- Launch an Amazon EC2 instance that runs a web server Attach an Amazon Elastic Block Store (Amazon EBS) volume to store the archived data. Use the Cold HDD (sc1) volume type. Configure the instance security groups to allow access only from private networks.

D- Create an Amazon S3 bucket. Configure the S3 bucket to use the S3 Glacier Deep Archive storage class as default. Configure the S3 bucket for website hosting. Create an S3 interface endpoint. Configure the S3 bucket to allow access only through that endpoint.

Answer:

D

Question 3

A company is using an on-premises Active Directory service for user authentication. The company wants to use the same authentication service to sign in to the company's AWS accounts, which are using AWS Organizations. AWS Site-to-Site VPN connectivity already exists between the on-premises environment and all the company's AWS accounts.

The company's security policy requires conditional access to the accounts based on user groups and roles. User identities must be managed in a single location.

Which solution will meet these requirements?

Options:

A- Configure AWS Single Sign-On (AWS SSO) to connect to Active Directory by using SAML 2.0. Enable automatic provisioning by using the System for Cross- domain Identity Management (SCIM) v2.0 protocol. Grant access to the AWS accounts by using attribute-based access controls (ABACs).

B- Configure AWS Single Sign-On (AWS SSO) by using AWS SSO as an identity source. Enable automatic provisioning by using the System for Cross-domain Identity Management (SCIM) v2.0 protocol. Grant access to the AWS accounts by using AWS SSO permission sets.

C- In one of the company's AWS accounts, configure AWS Identity and Access Management (IAM) to use a SAML 2.0 identity provider. Provision IAM users that are mapped to the federated users. Grant access that corresponds to appropriate groups in Active Directory. Grant access to the required AWS accounts by using cross-account IAM users.

D- In one of the company's AWS accounts, configure AWS Identity and Access Management (IAM) to use an OpenID Connect (OIDC) identity provider. Provision IAM roles that grant access to the AWS account for the federated users that correspond to appropriate groups in Active Directory. Grant access to the required AWS accounts by using cross-account IAM roles.

Answer:

A

Explanation:

https://aws.amazon.com/blogs/aws/new-attributes-based-access-control-with-aws-single-sign-on/

Question 4

Question Type: **MultipleChoice**

A company that has multiple AWS accounts is using AWS Organizations. The company's AWS accounts host VPCs, Amazon EC2 instances, and containers.

The company's compliance team has deployed a security tool in each VPC where the company has deployments. The security tools run on EC2 instances and send information to the AWS account that is dedicated for the compliance team. The company has tagged all the compliance-related resources with a key of "costCenter" and a value or "compliance".

The company wants to identify the cost of the security tools that are running on the EC2 instances so that the company can charge the compliance team's AWS account. The cost calculation must be as accurate as possible.

What should a solutions architect do to meet these requirements?

Options:

A- In the management account of the organization, activate the costCenter user-defined tag. Configure monthly AWS Cost and Usage Reports to save to an Amazon S3 bucket in the management account. Use the tag breakdown in the report to obtain the total cost for the costCenter tagged resources.

B- In the member accounts of the organization, activate the costCenter user-defined tag. Configure monthly AWS Cost and Usage Reports to save to an Amazon S3 bucket in the management account. Schedule a monthly AWS Lambda function to retrieve the reports and calculate the total cost for the costCenter tagged resources.

C- In the member accounts of the organization activate the costCenter user-defined tag. From the management account, schedule a monthly AWS Cost and Usage Report. Use the tag breakdown in the report to calculate the total cost for the costCenter tagged resources.

D- Create a custom report in the organization view in AWS Trusted Advisor. Configure the report to generate a monthly billing summary for the costCenter tagged resources in the compliance team's AWS account.

Answer:

C

Question 5

A company is in the process of implementing AWS Organizations to constrain its developers to use only Amazon EC2, Amazon S3, and Amazon DynamoDB. The developers account resides in a dedicated organizational unit (OU). The solutions architect has implemented the following SCP on the developers account:

```json
{
        "Version": "2012-10-17",
        "Statement": [
                {
                        "Sid": "AllowEC2",
                        "Effect": "Allow",
                        "Action": "ec2:*",
                        "Resource": "*"
                },
                {
                        "Sid": "AllowDynamoDB",
                        "Effect": "Allow",
                        "Action": "dynamodb:*",
                        "Resource": "*"
                },
                {
                        "Sid": "AllowS3",
                        "Effect": "Allow",
                        "Action": "s3:*",
                        "Resource": "*"
                }
        ]
}
```

When this policy is deployed, IAM users in the developers account are still able to use AWS services that are not listed in the policy.

What should the solutions architect do to eliminate the developers' ability to use services outside the scope of this policy?

Options:

A- Create an explicit deny statement for each AWS service that should be constrained.

B- Remove the FullAWSAccess SCP from the Developer account's OU.

C- Modify the FullAWSAccess SCP to explicitly deny all services.

D- Add an explicit deny statement using a wildcard to the end of the SCP.

Answer:

B

Explanation:

https://docs.aws.amazon.com/organizations/latest/userguide/orgs_manage_policies_scps_strategies.html#orgs_policies_allowlist 'To use SCPs as an allow list, you must replace the AWS managed FullAWSAccess SCP with an SCP that explicitly permits only those services and actions that you want to allow. By removing the default FullAWSAccess SCP, all actions for all services are now implicitly denied. Your custom SCP then overrides the implicit Deny with an explicit Allow for only those actions that you want to permit.'

Question 6

A company is hosting a monolithic REST-based API for a mobile app on five Amazon EC2 instances in public subnets of a VPC. Mobile clients connect to the API by using a domain name that is hosted on Amazon Route 53. The company has created a Route 53 multivalue answer routing policy with the IP addresses of all the EC2 instances. Recently, the app has been overwhelmed by large and sudden increases to traffic. The app has not been able to keep up with the traffic.

A solutions architect needs to implement a solution so that the app can handle the new and varying load.

Which solution will meet these requirements with the LEAST operational overhead?

Options:

A- Separate the API into individual AWS Lambda functions. Configure an Amazon API Gateway REST API with Lambda integration for the backend. Update the Route 53 record to point to the API Gateway API.

B- Containerize the API logic. Create an Amazon Elastic Kubernetes Service (Amazon EKS) cluster. Run the containers in the cluster by using Amazon EC2. Create a Kubernetes ingress. Update the Route 53 record to point to the Kubernetes ingress.

C- Create an Auto Scaling group. Place all the EC2 instances in the Auto Scaling group. Configure the Auto Scaling group to perform scaling actions that are based on CPU utilization. Create an AWS Lambda function that reacts to Auto Scaling group changes and updates the Route 53 record.

D- Create an Application Load Balancer (ALB) in front of the API. Move the EC2 instances to private subnets in the VPC. Add the EC2

instances as targets for the ALB. Update the Route 53 record to point to the ALB.

Answer:

A