

DESIRE FOR CERTS
WAS NEVER
HOTTER

AWS Certified DevOps
Engineer - Professional

Exam: AWS-DevOps-Engineer

Demo Edition

QUESTION: 1

You have an application which consists of EC2 instances in an Auto Scaling group. Between a particular time frame every day, there is an increase in traffic to your website. Hence users are complaining of a poor response time on the application. You have configured your Auto Scaling group to deploy one new EC2 instance when CPU utilization is greater than 60% for 2 consecutive periods of 5 minutes. What is the least cost-effective way to resolve this problem?

- A. Decrease the consecutive number of collection periods
- B. Increase the minimum number of instances in the Auto Scaling group
- C. Decrease the collection period to ten minutes
- D. Decrease the threshold CPU utilization percentage at which to deploy a new instance

Answer: B

Explanation:

If you increase the minimum number of instances, then they will be running even though the load is not high on the website. Hence you are incurring cost even though there is no need. All of the remaining options are possible options which can be used to increase the number of instances on a high load.

For more information on On-demand scaling, please refer to the below link:

<http://docs.aws.amazon.com/autoscaling/latest/userguide/as-scale-based-on-demand.html>

Note: The tricky part where the question is asking for 'least cost effective way'. You got the design consideration correctly but need to be careful on how the question is phrased.

QUESTION: 2

You have decided that you need to change the instance type of your production instances which are running as part of an AutoScaling group. The entire architecture is deployed using CloudFormation Template. You currently have 4 instances in Production. You cannot have any interruption in service and need to ensure 2 instances are always running during the update? Which of the options below listed can be used for this?

- A. AutoScalingRollingUpdate
- B. AutoScalingScheduledAction
- C. AutoScalingReplacingUpdate
- D. AutoScalingIntegrationUpdate

Answer: A

Explanation:

The AWS::AutoScaling::AutoScalingGroup resource supports an UpdatePolicy attribute. This is used to define how an Auto Scaling group resource is updated when an update to the Cloud Formation stack occurs. A common approach to updating an Auto Scaling group is to perform a rolling update, which is done by specifying the AutoScalingRollingUpdate policy. This retains the same Auto Scaling group and

replaces old instances with new ones, according to the parameters specified. For more information on Autoscaling updates, please refer to the below link:
<https://aws.amazon.com/premiumsupport/knowledge-center/auto-scaling-group-rolling-updates/>

QUESTION: 3

You currently have the following setup in AWS

- 1) An Elastic Load Balancer
- 2) Auto Scaling Group which launches EC2 Instances
- 3) AMIs with your code pre-installed

You want to deploy the updates of your app to only a certain number of users. You want to have a cost-effective solution. You should also be able to revert back quickly. Which of the below solutions is the most feasible one?

- A. Create a second ELB, and a new Auto Scaling Group assigned a new Launch Configuration. Create a new AMI with the updated app. Use Route53 Weighted Round Robin records to adjust the proportion of traffic hitting the two ELBs.
- B. Create new AM Is with the new app. Then use the new EC2 instances in half proportion to the older instances.
- C. Redeploy with AWS Elastic Beanstalk and Elastic Beanstalk versions. Use Route 53 Weighted Round Robin records to adjust the proportion of traffic hitting the two ELBs
- D. Create a full second stack of instances, cut the DNS over to the new stack of instances, and change the DNS back if a rollback is needed.

Answer: A

Explanation:

The Weighted Routing policy of Route53 can be used to direct a proportion of traffic to your application. The best option is to create a second CLB, attach the new Autoscaling Group and then use Route53 to divert the traffic.

Option B is wrong because just having EC2 instances running with the new code will not help. Option C is wrong because Elastic beanstalk is good for development environments, and also there is no mention of having 2 environments where environment url's can be swapped.

Option D is wrong because you still need Route53 to split the traffic.

For more information on Route53 routing policies, please refer to the below link:
<http://docs.aws.amazon.com/Route53/latest/DeveloperGuide/routing-policy.html>

QUESTION: 4

Your application is currently running on Amazon EC2 instances behind a load balancer. Your management has decided to use a Blue/Green deployment strategy. How should you implement this for each deployment?

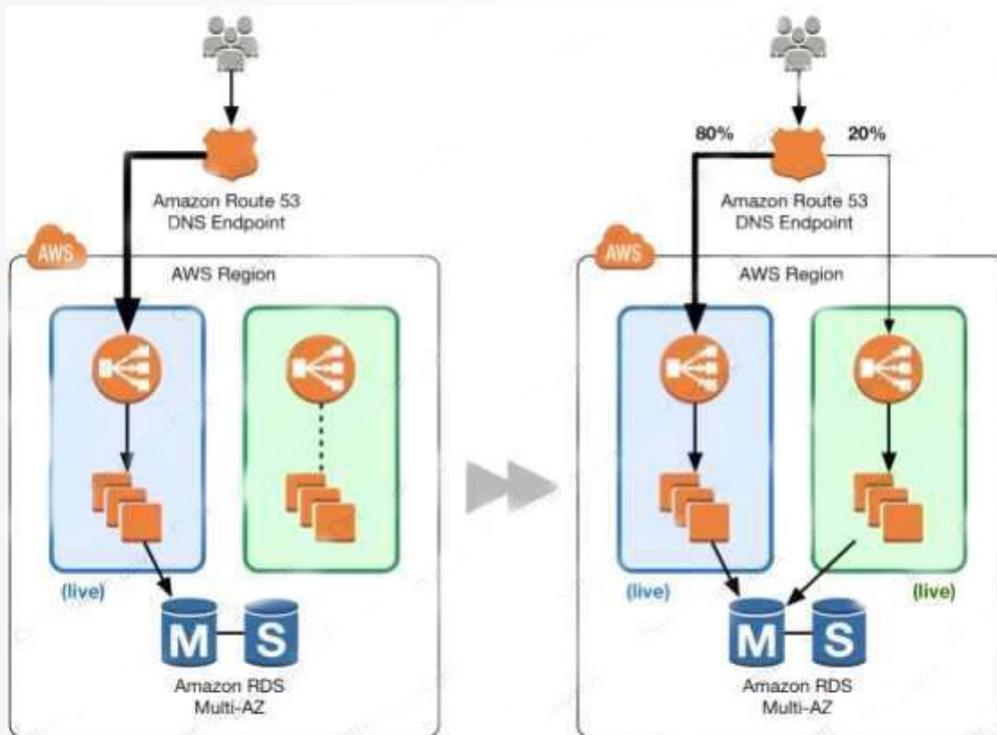
- A. Set up Amazon Route 53 health checks to fail over from any Amazon EC2 instance that is currently being deployed to.

- B. Using AWS CloudFormation, create a test stack for validating the code, and then deploy the code to each production Amazon EC2 instance.
- C. Create a new load balancer with new Amazon EC2 instances, carry out the deployment, and then switch DNS over to the new load balancer using Amazon Route 53 after testing.
- D. Launch more Amazon EC2 instances to ensure high availability, de-register each Amazon EC2 instance from the load balancer, upgrade it, and test it, and then register it again with the load balancer.

Answer: C

Explanation:

The below diagram shows how this can be done



- 1) First create a new ELB which will be used to point to the new production changes.
 - 2) Use the Weighted Route policy for Route53 to distribute the traffic to the 2 ELB's based on a 80- 20% traffic scenario. This is the normal case, the % can be changed based on the requirement.
 - 3) Finally when all changes have been tested, Route53 can be set to 100% for the new ELB. Option A is incorrect because this is a failover scenario and cannot be used for Blue green deployments. In Blue Green deployments, you need to have 2 environments running side by side.
- Option B is incorrect, because you need to have a production stack with the changes which will run side by side.

Option D is incorrect because this is not a blue green deployment scenario. You cannot control which users will go the new EC2 instances.

For more information on blue green deployments, please refer to the below document link: from https://dOawsstatic.com/whitepapers/AWS_Blue_Green_Deployments.pdf

QUESTION: 5

You have an application running a specific process that is critical to the application's functionality, and have added the health check process to your Auto Scaling Group. The instances are showing healthy but the application itself is not working as it should. What could be the issue with the health check, since it is still showing the instances as healthy.

- A. You do not have the time range in the health check properly configured
- B. It is not possible for a health check to monitor a process that involves the application
- C. The health check is not configured properly
- D. The health check is not checking the application process

Answer: D

Explanation:

If you have custom health checks, you can send the information from your health checks to Auto Scaling so that Auto Scaling can use this information. For example, if you determine that an instance is not functioning as expected, you can set the health status of the instance to Unhealthy. The next time that Auto Scaling performs a health check on the instance, it will determine that the instance is unhealthy and then launch a replacement instance

For more information on Autoscaling health checks, please refer to the below document link: from [http://docs.aws.a mazon.com/autoscaling/latest/userguide/healthcheck.html](http://docs.aws.amazon.com/autoscaling/latest/userguide/healthcheck.html)