**AutoScaling**

1. You have been monitoring a sensitive autoscaling group, and you expect it to scale-in as you enter a period of holiday downtime. The auto scaling group is distributed over three AZs ( AZ - A & -B have two instances each, and AZ -C has three instances). All instances have different CPU and Memory utilization, and all instances have been running for a different number of days. All instances come from different versions of a root AMI, and all instances have different numbers of sessions connected. Which instance will be the 1st to shut down?
   1. The instance in AZ – C that has been running the longest will terminate first.
   2. The instance in AZ - C that has the older launch configuration will terminate first.
   3. The instance that has been running longest will terminate first.
   4. The instance with the fewest current session will terminate first.
   5. The instance in AZ-C that has the least number of sessions will terminate first.
2. You are running a media-rich website with a global audience from us-east-1 for a customer in the publishing industry. The website updates every 20 minutes. The web-tier of the site sits on three EC2 instances inside an Auto Scaling Group. The Auto Scaling group is configured to scale when CPU utilization of the instances is greater than 70%. The Auto Scaling group sits behind an Elastic Load Balancer, and your static content lives in S3 and is distributed globally by CloudFront. Your RDS database is already the largest instance size available. CloudWatch metrics show that your RDS instance usually has around 2GB of memory free, and an average CPU utilization of 75%. Currently, it is taking your users in Japan and Australia approximately 3 - 5 seconds to load your website, and you have been asked to help reduce these load-times. How might you improve your page load times? [Select 3]
   1. Use ElasticCache to cache the most commonly accessed DB queries.
   2. Setup CloudFront with dynamic content support to enable the caching of re-usable content from the media rich website.
   3. Setup a clone of your production environment in the Asia Pacific region and configure latency based routing on Route53
   4. Change your ASG so that it will scale when CPU utilizations is only 50% rather than 70%
   5. Increase the provisioned IOPs on the EBS volume.
3. You work for a toy company that has a busy online store. As you are approaching Christmas, you find that your store is getting more and more traffic. You ensure that the web tier of your store is behind an Auto Scaling group. However, you notice that the web tier is frequently scaling, sometimes multiple times in an hour, only to scale back after peak usage. You need to keep Auto Scaling from scaling up and down so rapidly. Which of the following options would help you to achieve this?
   1. Configure Auto Scaling to terminate your newest instances first, then adjust your CloudWatch Alarms.
   2. Modify the ASG cool-down timers and modify the Amazon CloudWatch alarm period that triggers your ASG scale down policy.
   3. Change your ASG policy so that it only scales at scheduled times.
   4. Configure ASG to terminate your oldest instances first then adjust your cloudwatch alarm.
4. You have set up an Auto Scaling group. The cool down period for the Auto Scaling group is 7 minutes. The first instance is launched after 3 minutes, while the second instance is launched after 4 minutes. How many minutes after the first instance is launched will Auto Scaling accept another scaling actMty request?
   1. 11, b. 7, c 10, d, 14
5. You have been given a scope to deploy some AWS infrastructure for a large organisation. The requirements are that you will have a lot of EC2 instances but may need to add more when the average utilization of your Amazon EC2 fileet is high and conversely remove them when CPU utilization is low. Which AWS services would be best to use to accomplish this?

A. Auto Scaling, Amazon CIoudWatch and AWS Elastic Beanstalk

B. Auto Scaling, Amazon CIoudWatch and Elastic Load Balancing.

C. Amazon CIoudFront, Amazon CIoudWatch and Elastic Load Balancing.

D. AWS Elastic Beanstalk , Amazon CIoudWatch and Elastic Load Balancing

1. In an experiment, if the minimum size for an Auto Scaling group is 1 instance, which of the following statements holds true when you terminate the running instance?

A. Auto Scaling must launch a new instance to replace it.

B. Auto Scaling will raise an alarm and send a notification to the user for action.

C. Auto Scaling must configure the schedule actMty that terminates the instance after 5 days.

D. Auto Scaling will terminate the experiment

1. You decide that you need to create a number of Auto Scaling groups to try and save some money as you have noticed that at certain times most of your EC2 instances are not being used. By default, what is the maximum number of Auto Scaling groups that AWS will allow you to create?

A. 12

B. Unlimited

C. 20

D. 2

1. A user is planning to launch a scalable web application. Which of the below mentioned options will not affect the latency of the application?

A. Region.

B. Provisioned IOPS.

C. Availability Zone.

D. Instance size

1. After moving an E-Commerce website for a client from a dedicated server to AWS you have also set up auto scaling to perform health checks on the instances in your group and replace instances that fail these checks. Your client has come to you with his own health check system that he wants you to use as it has proved to be very useful prior to his site running on AWS. What do you think would be an appropriate response to this given all that you know about auto scaling?

A. It is not possible to implement your own health check syste

B. You need to use AWSs health check system.

C. It is not possible to implement your own health check system due to compatibility issues.

D. It is possible to implement your own health check system and then send the instance's health information directly from your system to Cloud Watch.

E. It is possible to implement your own health check system and then send the instance's health information directly from your system to Cloud Watch but only in the US East (Virginia) region.

1. You run an automobile reselling company that has a popular online store on AWS. The application sits behind an Auto Scaling group and requires new instances of the Auto Scaling group to identify their public and private IP addresses. How can you achieve this?
   1. By using ipconfig for windows and ifconfig for linux.
   2. By using CloudWatch Metric.
   3. Using a Curl or Get command to get the latest user-data from <http://169.254.169.254/latest/user-data>
   4. Using a Curl or Get command to get the latest meta-data from http://169.254.169.254/latest/meta-data

FAQ

**Q. What are the benefits of AWS Auto Scaling?**

AWS Auto Scaling is a fast, easy way to optimize the performance and costs of your applications.

Setup scaling quickly: AWS Auto Scaling provides a unified scaling experience for all of the scalable resources powering your application. You can see the average utilization for all of your scalable resources and quickly define target utilization levels for each group of like resources from a single, intuitive interface.

Make smart scaling decisions: AWS Auto Scaling lets you automate how groups of different resources respond to changes in demand. Easy-to-understand scaling strategies let you choose to optimize availability, costs, or a balance of both. AWS Auto Scaling automatically creates all of the scaling policies and sets targets for you based on your preference.

Automatically maintain performance: AWS Auto Scaling continually monitors resources underlying your application to make sure that they are operating at your desired performance levels. When demand spikes, AWS Auto Scaling automatically increases the capacity of constrained resources so you maintain a high quality of service.

Anticipate costs and avoid overspending: AWS Auto Scaling can help you optimize your utilization and cost efficiencies when consuming AWS services so you only pay for the resources you actually need. When demand drops, AWS Auto Scaling will automatically remove any excess resource capacity so you avoid overspending.

**Q. When should I use AWS Auto Scaling?**

You should use AWS Auto Scaling if you have an application that uses one or more scalable resources and experiences variable load. A good example would be an e-commerce web application that receives variable traffic through the day. It follows a standard three tier architecture with Elastic Load Balancing for distributing incoming traffic, Amazon EC2 for the compute layer, and DynamoDB for the data layer. In this case, AWS Auto Scaling will scale one or more EC2 Auto Scaling groups and DynamoDB tables that are powering the application in response to the demand curve.

**Q. What are the different ways that I can scale AWS resources?**

AWS customers have multiple options for scaling resources. Amazon EC2 Auto Scaling helps you ensure that you have the correct number of Amazon EC2 instances available to handle the load for your application. EC2 Auto Scaling can also detect when an instance is unhealthy, terminate it, and launch an instance to replace it. When you use EC2 Auto Scaling, your applications gain better fault tolerance, availability, and cost management.

To scale a resource other than EC2, you can use the Application Auto Scaling API, which allows you to define scaling policies to automatically scale your AWS resources or schedule one-time or recurring scaling actions. Application Auto Scaling can scale Amazon ECS services, Amazon EC2 Spot fleets, Amazon EMR clusters, Amazon AppStream 2.0 fleets, provisioned read and write capacity for Amazon DynamoDB tables and global secondary indexes, Amazon Aurora Replicas, and Amazon SageMaker endpoint variants.

To configure automatic scaling for multiple resources across multiple services, use AWS Auto Scaling to create a scaling plan for the resources underlying your application. AWS Auto Scaling is also used to create predictive scaling for EC2 resources.

**Q. When should I use AWS Auto Scaling vs. Amazon EC2 Auto Scaling?**

You should use AWS Auto Scaling to manage scaling for multiple resources across multiple services. AWS Auto Scaling lets you define dynamic scaling policies for multiple EC2 Auto Scaling groups or other resources using predefined scaling strategies. Using AWS Auto Scaling to configure scaling policies for all of the scalable resources in your application is faster than managing scaling policies for each resource via its individual service console. It’s also easier, as AWS Auto Scaling includes predefined scaling strategies that simplify the setup of scaling policies. You should also use AWS Auto Scaling if you want to create predictive scaling for EC2 resources.

You should use EC2 Auto Scaling if you only need to scale Amazon EC2 Auto Scaling groups, or if you are only interested in maintaining the health of your EC2 fleet. You should also use EC2 Auto Scaling if you need to create or configure Amazon EC2 Auto Scaling groups, or if you need to set up scheduled or step scaling policies (as AWS Auto Scaling supports only target tracking scaling policies).

EC2 Auto Scaling groups must be created and configured outside of AWS Auto Scaling, such as through the EC2 console, Auto Scaling API or via CloudFormation. AWS Auto Scaling can help you configure dynamic scaling policies for your existing EC2 Auto Scaling groups

**Q. When should I use AWS Auto Scaling vs. Auto Scaling for individual services?**

You should use AWS Auto Scaling to manage scaling for multiple resources across multiple services. AWS Auto Scaling enables unified scaling for multiple resources, and has predefined guidance that helps make it easier and faster to configure scaling. If you prefer, you can instead choose to use the individual service consoles, Auto Scaling API, or Application Auto Scaling API to scale individual AWS services. You should also use the individual consoles or API if you want to setup step scaling policies or scheduled scaling, as AWS Auto Scaling creates target tracking scaling policies only.

**Q. What is Predictive Scaling?**

Predictive Scaling is a feature of AWS Auto Scaling that looks at historic traffic patterns and forecasts them into the future to schedule changes in the number of EC2 instances at the appropriate times going forward. Predictive Scaling uses machine learning models to forecast daily and weekly patterns.

Auto Scaling enhanced with Predictive Scaling delivers faster, simpler, and more accurate capacity provisioning resulting in lower cost and more responsive applications. By predicting traffic changes, Predictive Scaling provisions EC2 instances in advance of changing traffic, making Auto Scaling faster and more accurate.

**Q. Which services can I use Predictive Scaling with?**

At this time, Predictive Scaling only generates schedules for EC2 instances.

**Q. How can I use Predictive Scaling with target tracking?**

Predictive Scaling works with in conjunction with target tracking to make your EC2 capacity changes more responsive to your incoming application traffic. While Predictive Scaling sets up the minimum capacity for your application based on forecasted traffic, target tracking changes the actual capacity based on the actual traffic at the moment. Target tracking works to track the desired capacity utilization levels over varying traffic conditions and addresses unpredicted traffic spikes and other fluctuations. Predictive Scaling and target tracking are configured together by a user to generate a scaling plan.

**Q. What is a scaling plan?**

A scaling plan is a collection of scaling instructions for multiple AWS resources. You configure a scaling plan by first selecting all the EC2 resources underlying your application in AWS Auto Scaling. Then you pick the resource utilization metric that you would like to track, such as CPU utilization, and set the value to track, for example 50%. Finally, you select the CloudWatch metric that represents your input traffic flow – you might have to set this up if you haven’t already.

The resource utilization metric and the incoming traffic metric are the key parameters for the scaling plan. The incoming traffic metric is used by Predictive Scaling to generate traffic forecasts. Based on these forecasts, Predictive Scaling then schedules future scaling actions to configure minimum capacity. Dynamic Scaling uses the resource utilization metric and its target value to dynamically change the EC2 capacity for your application over time as traffic varies.

**Q. Can I configure a scaling plan without Predictive Scaling?**

Yes, you can configure a scaling plan with only Dynamic Scaling and opt-out of Predictive Scaling. Conversely, you can also enable just Predictive Scaling without configuring Dynamic Scaling.

**Q. How much historical data does Predictive Scaling need to generate the scaling plan?**

Predictive Scaling needs up to two weeks of historical data but can generate a predictive scaling schedule with as little as a day's worth of data.

**Q. How much into the future does Predictive Scaling forecast the traffic?**

Every 24 hours, Predictive Scaling forecasts traffic 48 hours into the future and schedules capacity changes for those 48 hours.

**Q. Can I configure Predictive Scaling to provision instances before an actual spike in traffic?**

Yes, you can optionally configure buffer time to provision instances at some time before a predicted traffic change. This is useful for applications whose EC2 instances need some “warm-up” time before they are ready to serve application traffic

**Q. How much does Predictive Scaling cost?**

As with other Auto Scaling features, Predictive Scaling is free to use. You pay for the resources being utilized for running your applications.

**Q. What can I scale with AWS Auto Scaling?**

You can use AWS Auto Scaling to setup scaling for the following resources in your application through a single, unified interface:

Amazon EC2 Auto Scaling groups

Amazon Elastic Container Service (ECS) services (currently ECS services cannot be discovered using resource tags)

Amazon EC2 Spot Fleets

Amazon DynamoDB throughput capacity

Aurora replicas for Amazon Aurora

**Q. How does AWS Auto Scaling make scaling recommendations?**

AWS Auto Scaling bases its scaling recommendations on the most popular scaling metrics and thresholds used for Auto Scaling. It also recommends safe guardrails for scaling by providing recommendations for the minimum and maximum sizes of the resources. This way you can get started quickly and can then fine tune your scaling strategy over time.

**Q. How do I select an application stack within AWS Auto Scaling?**

You can either select an AWS CloudFormation stack or select resources based on common resource tag(s). Please note that currently, ECS services cannot be discovered using tags

**Q. How does AWS Auto Scaling discover what resources can scale?**

AWS Auto Scaling will scan your selected AWS CloudFormation stack or resources with the specified tags to identify the supported AWS resource types that can be scaled. Please note that currently, ECS services cannot be discovered using tags

**Q. Which regions is AWS Auto Scaling available in?**

AWS Auto Scaling is available in Asia Pacific (Mumbai), Asia Pacific (Tokyo), Asia Pacific (Seoul), Asia Pacific (Sydney), Canada (Central), US West (Northern California), Europe (London), Europe (Frankfurt), EU (Paris), EU (Milan), US East (Virginia), US East (Ohio), US West (Oregon), EU (Ireland), and Asia Pacific (Singapore).

**Q. How much does AWS Auto Scaling cost?**

Similar to Auto Scaling on individual AWS resources, AWS Auto Scaling is free to use. AWS Auto Scaling is enabled by Amazon CloudWatch, so service fees apply for CloudWatch and your application resources (such as Amazon EC2 instances, Elastic Load Balancing load balancers, etc.).

|  |  |  |
| --- | --- | --- |
| No | Answer | Explanation |
| 1 | B | AutoScaling scales-in according to a hierarchy of decisions. |
| 2 | A, B, C | Additional clones of your production environment, ElastiCache, and CloudFront can all help improve your site performance. Changing your autoscaling policies will not help improve performance times as it is much more likely that the performance issue is with the database back end rather than the front end. The Provisioned IOPS would also not help, as the bottleneck is with the memory, not the storage |
| 3 | B |  |
| 4 | A | If an Auto Scaling group is launching more than one instance, the cool down period for each instance starts after that instance is launched. The group remains locked until the last instance that was launched has completed its cool down period. In this case the cool down period for the first instance starts after 3 minutes and finishes at the 10th minute (3+7 cool down), while for the second instance it starts at the 4th minute and finishes at the 11th minute (4+7 cool down). Thus, the Auto Scaling group will receive another request only after 11 minutes. |
| 5 | B | Auto Scaling enables you to follow the demand curve for your applications closely, reducing the need to manually provision Amazon EC2 capacity in advance. For example, you can set a condition to add new  Amazon EC2 instances in increments to the Auto Scaling group when the average utilization of your Amazon EC2 fileet is high; and similarly, you can set a condition to remove instances in the same increments when CPU utilization is low. If you have predictable load changes, you can set a schedule through Auto Scaling to plan your scaling actMties. You can use Amazon CIoudWatch to send alarms to trigger scaling actMties and Elastic Load Balancing to help distribute traffic to your instances within Auto Scaling groups. Auto Scaling enables you to run your Amazon EC2 fileet at optimal utilization |
| 6 | A | If the minimum size for an Auto Scaling group is 1 instance, when you terminate the running instance, Auto Scaling must launch a new instance to replace it. |
| 7 | C | Auto Scaling is an AWS service that allows you to increase or decrease the number of EC2 instances within your appIication's architecture. With Auto Scaling, you create collections of EC2 instances, called Auto Scaling groups. You can create these groups from scratch, or from existing EC2 instances that are already in production. |
| 8 | C | In AWS, the instance size decides the I/O characteristics. The provisioned IOPS ensures higher throughput, and lower latency. The region does affect the latency; latency will always be less when the instance is near to the end user. Within a region the user uses any AZ and this does not affect the latency. The AZ is mainly for fault toleration or HA. |
| 9 | C | Auto Scaling periodically performs health checks on the instances in your group and replaces instances that fail these checks. By default, these health checks use the results of EC2 instance status checks to determine the health of an instance. If you use a load balancer with your Auto Scaling group, you  can optionally choose to include the results of Elastic Load Balancing health checks.  Auto Scaling marks an instance unhealthy if the calls to the Amazon EC2 action DescribeInstanceStatus returns any other state other than running, the system status shows impaired, or the calls to Elastic Load Balancing action DescribeInstanceHeaIth returns OutOfService in the instance state field. After an instance is marked unhealthy because of an Amazon EC2 or Elastic Load Balancing health check, it is scheduled for replacement.  You can customize the health check conducted by your Auto Scaling group by specifying additional checks or by having your own health check system and then sending the instance's health information directly from your system to Auto Scaling |
| 10 | D |  |