**AWS Auto Scaling**

Amazon EC2 Auto Scaling

AWS Auto Scaling monitors your applications and automatically adjusts capacity to maintain steady, predictable performance at the lowest possible cost.

AWS Auto Scaling refers to a collection of Auto Scaling capabilities across several AWS services.

The services within the AWS Auto Scaling family include:

* Amazon EC2 (known as Amazon EC2 Auto Scaling).
* [**Amazon ECS**](https://digitalcloud.training/certification-training/aws-solutions-architect-associate/compute/amazon-ecs/).
* [**Amazon DynamoDB**](https://digitalcloud.training/certification-training/aws-solutions-architect-associate/database/amazon-dynamodb/).
* [**Amazon Aurora**](https://digitalcloud.training/certification-training/aws-solutions-architect-associate/database/amazon-rds/).

This page is specifically for Amazon EC2 Auto Scaling – Auto Scaling will also be discussed for the other services on their respective pages.

General EC2 Auto Scaling Concepts

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.

You create collections of EC2 instances, called Auto Scaling groups.

Automatically provides horizontal scaling (scale-out) for your instances.

Triggered by an event of scaling action to either launch or terminate instances.

Availability, cost, and system metrics can all factor into scaling.

Auto Scaling is a region specific service.

Auto Scaling can span multiple AZs within the same AWS region.

Auto Scaling can be configured from the Console, CLI, SDKs and APIs.

There is no additional cost for Auto Scaling, you just pay for the resources (EC2 instances) provisioned.

Auto Scaling works with ELB, CloudWatch and CloudTrail.

You can determine which subnets Auto Scaling will launch new instances into.

Auto Scaling will try to distribute EC2 instances evenly across AZs.

Launch configuration is the template used to create new EC2 instances and includes parameters such as instance family, instance type, AMI, key pair and security groups.

You cannot edit a launch configuration once defined.

A launch configuration:

* Can be created from the AWS console or CLI.
* You can create a new launch configuration, or.
* You can use an existing running EC2 instance to create the launch configuration.
  + The AMI must exist on EC2.
  + EC2 instance tags and any additional block store volumes created after the instance launch will not be taken into account.
* If you want to change your launch configurations you have to create a new one, make the required changes, and use that with your auto scaling groups.

You can use a launch configuration with multiple Auto Scaling Groups (ASG).

An ASG is a logical grouping of EC2 instances managed by an Auto Scaling Policy.

An ASG can be edited once defined.

You can attach one or more classic ELBs to your existing ASG.

You can attach one or more Target Groups to your ASG to include instances behind an ALB.

The ELBs must be in the same region.

Once you do this any EC2 instance existing or added by the ASG will be automatically registered with the ASG defined ELBs.

If adding an instance to an ASG would result in exceeding the maximum capacity of the ASG the request will fail.

You can add a running instance to an ASG if the following conditions are met:

* The instance is in a running state.
* The AMI used to launch the instance still exists.
* The instance is not part of another ASG.
* The instance is in the same AZs for the ASG.

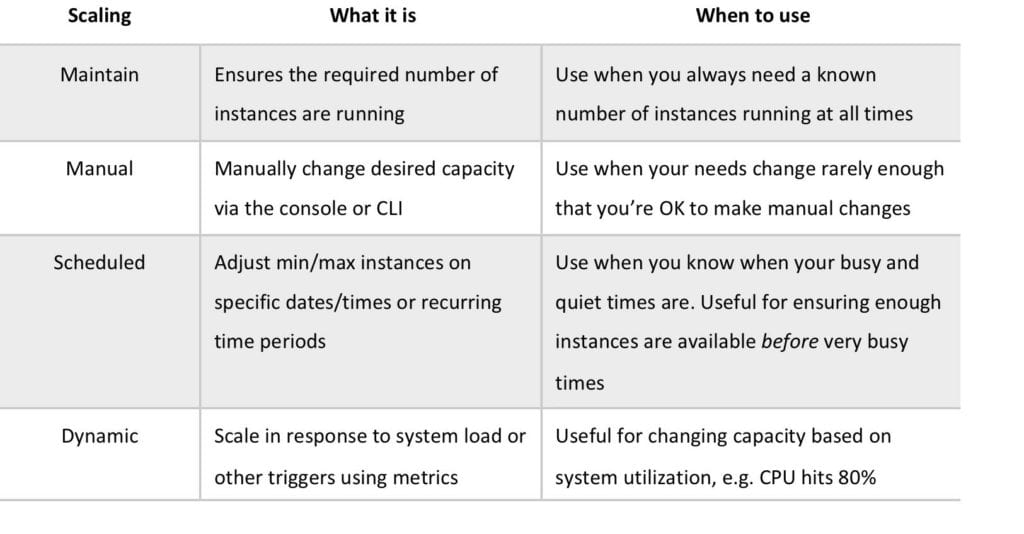
Scaling

The scaling options define the triggers and when instances should be provisioned/de-provisioned.

There are four scaling options:

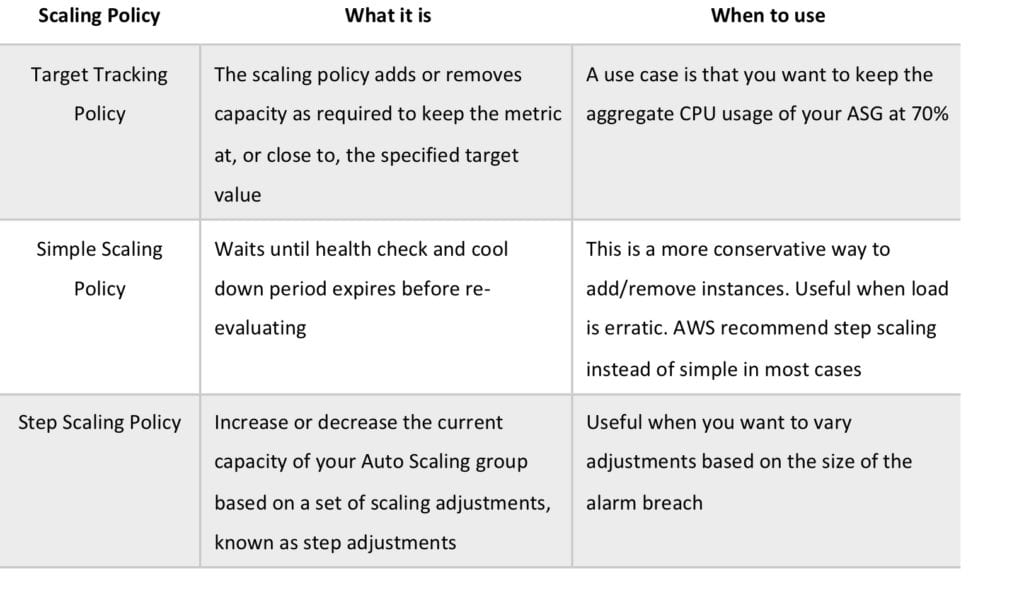
* Maintain – keep a specific or minimum number of instances running.
* Manual – use maximum, minimum, or a specific number of instances.
* Scheduled – increase or decrease the number of instances based on a schedule.
* Dynamic – scale based on real-time system metrics (e.g. CloudWatch metrics).

The following table describes the scaling options available and when to use them:

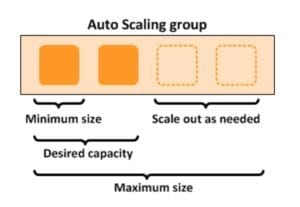


The scaling options are configured through Scaling Policies which determine when, if, and how the ASG scales and shrinks.

The following table describes the scaling policy types available for dynamic scaling policies and when to use them (more detail further down the page):



The diagram below depicts an Auto Scaling group with a Scaling policy set to a minimum size of 1 instance, a desired capacity of 2 instances, and a maximum size of 4 instances:



Scaling based on Amazon SQS

Can also scale based on an Amazon Simple Queue Service (SQS) queue.

This comes up as an exam question for SAA-C02.

Uses a custom metric that’s sent to Amazon CloudWatch that measures the number of messages in the queue per EC2 instance in the Auto Scaling group.

Then use a target tracking policy that configures your Auto Scaling group to scale based on the custom metric and a set target value. CloudWatch alarms invoke the scaling policy.

Use a custom “backlog per instance” metric to track not just the number of messages in the queue but the number available for retrieval.

Can base off the SQS Metric “ApproximateNumberOfMessages”.

ASG Behavior and Configuration

EC2 Auto Scaling – Termination Policy:

* Termination policies control which instances are terminated first when a scale-in event occurs.
* There is a default termination policy and options for configuring your own customized termination policies.
* The default termination policy is designed to help ensure that your instances span Availability Zones evenly for high availability.
* The default policy is kept generic and flexible to cover a range of scenarios.
* Details can be found on the [**AWS Website**](https://docs.aws.amazon.com/autoscaling/ec2/userguide/as-instance-termination.html).

You can define Instance Protection which stops Auto Scaling from scaling in and terminating the instances.

If Auto Scaling fails to launch instances in an AZ it will try other AZs until successful.

The default health check grace period is 300 seconds.

Scale-out is the process in which EC2 instances are launched by the scaling policy.

Scale-in is the process in which EC2 instances are terminated by the scaling policy.

It is recommended to create a scale-in event for each scale-out event created.

An imbalance may occur due to:

* Manually removing AZs/subnets from the configuration.
* Manually terminating EC2 instances.
* EC2 capacity issues.
* Spot price is reached.

Health checks:

* By default uses EC2 status checks.
* Can also use ELB health checks and custom health checks.
* ELB health checks are in addition to the EC2 status checks.
* If any health check returns an unhealthy status the instance will be terminated.
* With ELB an instance is marked as unhealthy if ELB reports it as OutOfService.
* A healthy instance enters the InService state.
* If an instance is marked as unhealthy it will be scheduled for replacement.
* If connection draining is enabled, Auto Scaling waits for in-flight requests to complete or timeout before terminating instances.
* The health check grace period allows a period of time for a new instance to warm up before performing a health check (300 seconds by default).

If using an ELB it is best to enable ELB health checks as otherwise EC2 status checks may show an instance as being healthy that the ELB has determined is unhealthy. In this case the instance will be removed from service by the ELB but will not be terminated by Auto Scaling.

Elastic IPs and EBS volumes are detached from terminated instances and will need to be manually reattached.

Using custom health checks a CLI command can be issued to set the instance’s status to unhealthy, e.g.:

***aws autoscaling set–instance-health –instance-id i-123abc45d –health-status Unhealthy***

Once in a terminating state an EC2 instance cannot be put back into service again.

However there is a short time period in which a CLI command can be run to change an instance to healthy.

Unlike AZ rebalancing, termination of unhealthy instances happens first, then Auto Scaling attempts to launch new instances to replace terminated instances.

You can manually remove (detach) instances from an ASG using the AWS Console or CLI.

When detaching an instance you can optionally decrement the ASG’s desired capacity (so it doesn’t launch another instance).

An instance can be attached to one ASG at a time.

You can suspend and then resume one or more of the scaling processes for your Auto Scaling group.

Suspending scaling processes can be useful when you want to investigate a configuration problem or other issue with your web application and then make changes to your application, without invoking the scaling processes.

You can manually move an instance from an ASG and put it in the standby state.

Instances in standby state are still managed by Auto Scaling, are charged as normal, and do not count towards available EC2 instance for workload/application use.

Auto scaling does not perform health checks on instances in the standby state.

Standby state can be used for performing updates/changes/troubleshooting etc. without health checks being performed or replacement instances being launched.

When you delete an ASG the instances will be terminated.

You can choose to use Spot instances in launch configurations and specify a bid price.

Auto Scaling treats spot instances the same as on-demand instances.

You cannot mix Spot instances with on-demand.

If you want to change the bid price you need to create a new launch configuration.

Auto Scaling can be configured to send an SNS email when:

* An instance is launched.
* An instance is terminated.
* An instance fails to launch.
* An instance fails to terminate.

Merging ASGs.

* You can merge multiple single AZ Auto Scaling Groups into a single multi-AZ ASG.
* Merging can only be performed by using the CLI.
* Process is to rezone one of the groups to cover/span the other AZs for the other ASGs.
* Then delete the other ASGs.
* Can be performed on ASGs with or without ELBs attached to them.
* The resulting ASG must be one of the pre-existing ASGs.

Cooldown Period:

* The cooldown period is a configurable setting for your Auto Scaling group that helps to ensure that it doesn’t launch or terminate additional instances before the previous scaling activity takes effect.
* The default cooldown period is applied when you create your Auto Scaling group.
* The default value is 300 seconds.
* You can configure the default cooldown period when you create the Auto Scaling group, using the AWS Management Console, the create-auto-scaling-group command (AWS CLI), or the CreateAutoScalingGroup API operation.
* Automatically applies to dynamic scaling and optionally to manual scaling but not supported for scheduled scaling.
* Can override the default cooldown via scaling-specific cooldown.

The warm-up period is the period of time in which a newly created EC2 instance launched by ASG using step scaling is not considered toward the ASG metrics.

Monitoring

Basic monitoring sends EC2 metrics to CloudWatch about ASG instances every 5 minutes.

Detailed can be enabled and sends metrics every 1 minute (chargeable).

When the launch configuration is created from the console basic monitoring of EC2 instances is enabled by default.

When the launch configuration is created from the CLI detailed monitoring of EC2 instances is enabled by default.

When you enable Auto Scaling group metrics, Auto Scaling sends sampled data to CloudWatch every minute.

Configure ASG and EC2 monitoring options so they use the same time period, e.g. detailed monitoring (EC2) and 60 seconds (ASG), or basic monitoring (EC2) and 300 seconds (ASG).

Limits

