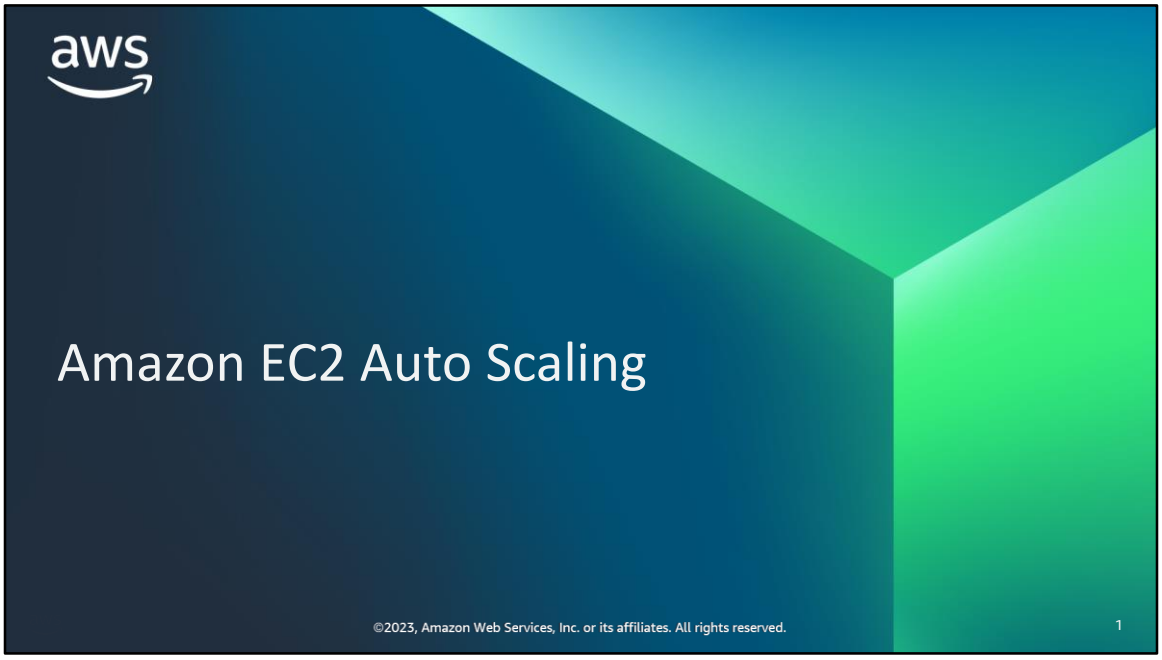


Amazon EC2 Auto Scaling

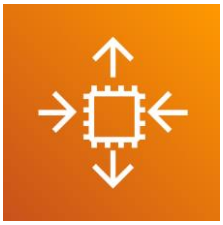
At the core of the lesson

You will learn how to do the following:

- Describe Amazon EC2 Auto Scaling and launch templates.
- Configure and manage launch templates and control scaling.
- Identify best practices for implementing auto scaling.


aws
Amazon EC2 Auto Scaling overview
©2023, Amazon Web Services, Inc. or its affiliates. All rights reserved.
3

What is Amazon EC2 Auto Scaling?



Amazon EC2
Auto Scaling

- Amazon EC2 Auto Scaling is a service that helps ensure application availability by automatically launching or terminating EC2 instances based on scaling options that you define.
- The following are the available scaling options:
  - Manual scaling
  - Scheduled scaling
  - Dynamic scaling
  - Predictive scaling

aws

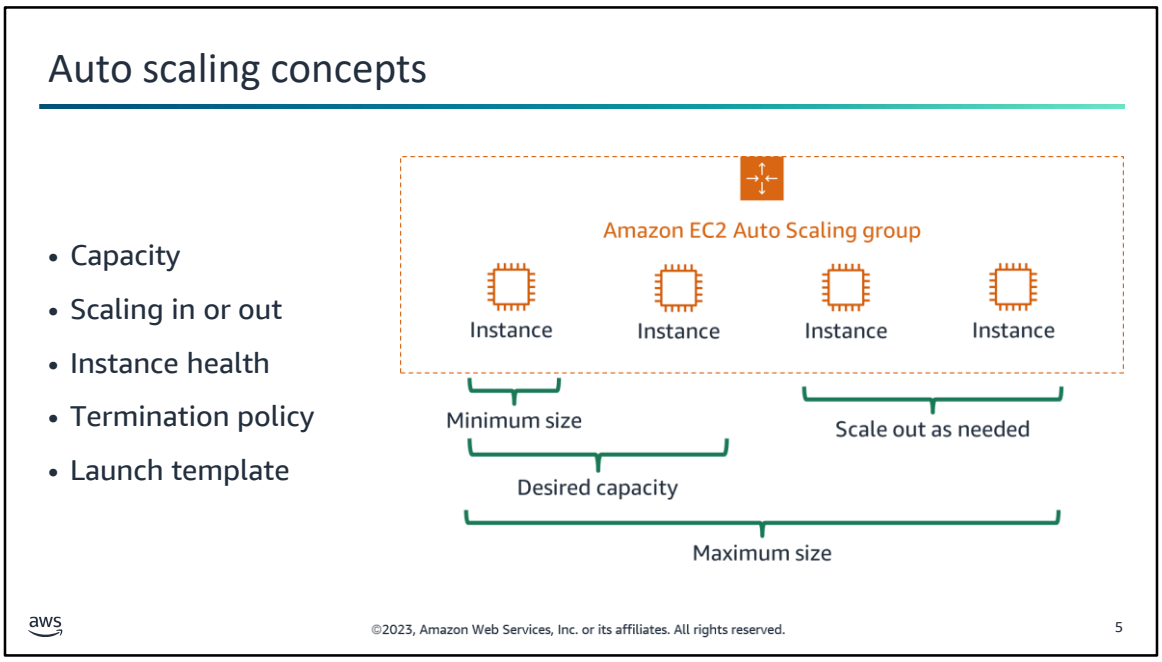
You can use Amazon EC2 Auto Scaling to maintain application availability. This service helps ensure that you have the correct number of Amazon Elastic Compute Cloud (Amazon EC2) instances available to handle the load of the application. You can use Amazon EC2 Auto Scaling to automatically add or remove EC2 instances according to conditions that you define. For example, you can configure Amazon EC2 Auto Scaling to terminate EC2 instances that fail health status checks. Then, you can configure it to launch new EC2 instances to replace the terminated ones. You can create user-defined policies that use Amazon CloudWatch to initiate when to add or remove EC2 instances. These policies would be based on conditions such as the average CPU utilization of your Amazon EC2 fleet.

The most basic scaling option is manual scaling. You specify the change in the maximum, minimum, or desired capacity of your auto scaling group, and Amazon EC2 Auto Scaling implements the change.

If you have predictable load changes, you can set a schedule through Amazon EC2 Auto Scaling to plan your scaling activities. The features of Amazon EC2 Auto Scaling give you the ability to scale out to meet demand and scale in to reduce costs.

You will look at the following scaling options in more detail in the subsequent slides:

- Scheduled
- Dynamic
- Predictive



Key concepts in auto scaling include the following:

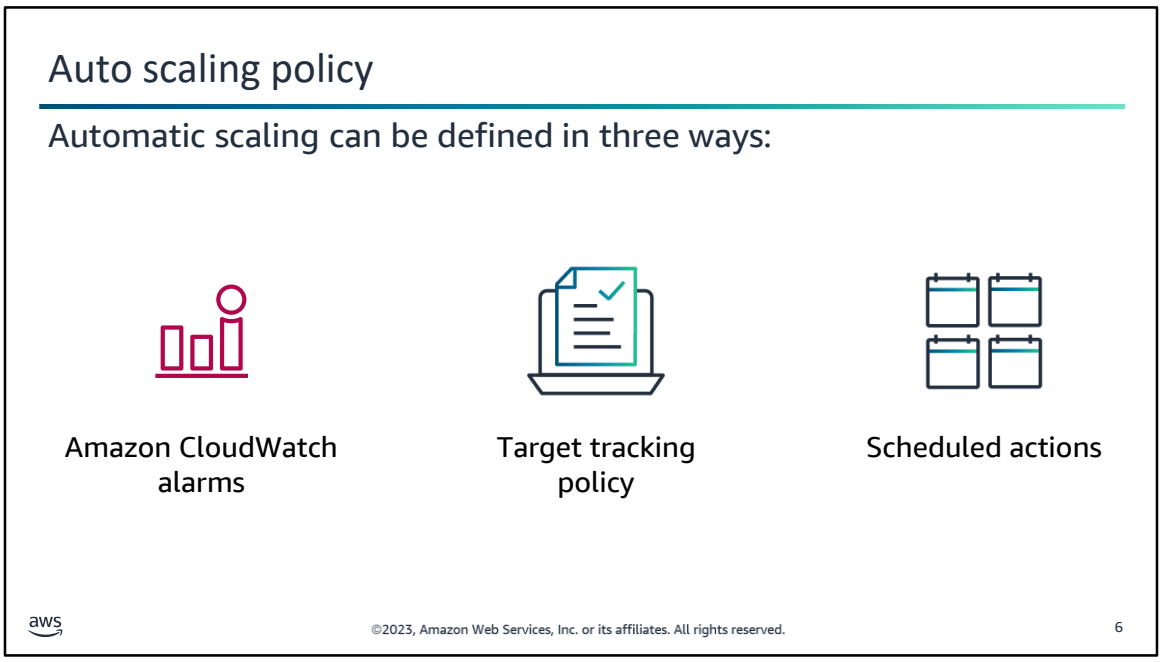
Capacity: Capacity limits represent the minimum and maximum group size that you want for your auto scaling group. The group's desired capacity represents the initial capacity of the auto scaling group at the time of creation. For example, in the diagram, the auto scaling group has a minimum size of one instance, a desired capacity of two instances, and a maximum size of four instances. The scaling policies that you define adjust the number of instances within your minimum and maximum ranges.

Scaling in and out: An increase in CPU utilization outside the desired range could cause the auto scaling group to scale out (adding two instances to the auto scaling group in the example shown). Then when the CPU utilization decreases, the auto scaling group would scale in, potentially returning to the minimum desired capacity by terminating instances.

Instance health: The health status of an auto scaling instance indicates whether it is healthy or unhealthy. This notification can come from sources such as Amazon EC2, Elastic Load Balancing (ELB), or custom health checks. When Amazon EC2 Auto Scaling detects an unhealthy instance, it terminates the instance and launches a new one.

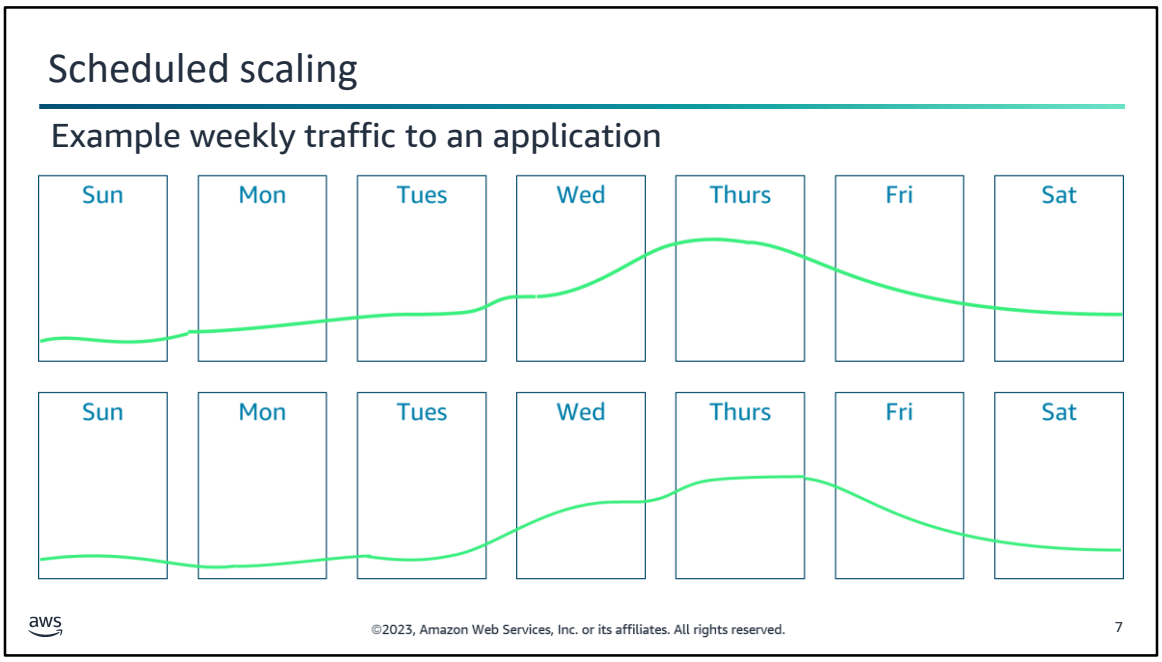
Termination policy: Amazon EC2 Auto Scaling uses termination policies to determine which instances it terminates first during scale-in events.

Launch template: A launch template specifies instance configuration information. It includes the ID of the Amazon Machine Image (AMI), the instance type, a key pair, security groups, and other parameters used to launch EC2 instances. When auto scaling groups scale out, the new instances are launched according to the configuration information specified in the latest version of the launch template.



Remember that automatic scaling can be defined in three ways:

- Define a scaling policy that either scales out or scales in based on an Amazon CloudWatch alarm. You can define a CloudWatch alarm—for example, Average CPU Utilization is greater than 50 percent for 2 minutes—that calls an Amazon EC2 Auto Scaling policy. The policy specifies one of two options. The first option, simple scaling, is to add or remove a fixed number of instances. The second option, step scaling, is to adjust the number of running instances as a percentage of the desired capacity for the Amazon EC2 Auto Scaling group. The response is varied based on the size of the alarm breach.
- Define a target tracking policy where a metric, such as the average CPU utilization, is evaluated against a target For instance, keep the average CPU utilization of your auto scaling group at 50 percent.
- Define a scheduled action. Scheduled actions set a new desired capacity value at a specific time. You can specify a scheduled action to start on a specific date and time. You can also specify a recurring action that is run at specific times throughout a week, month, or year. Scheduled actions are an excellent way to prewarm capacity in response to anticipated traffic spikes.

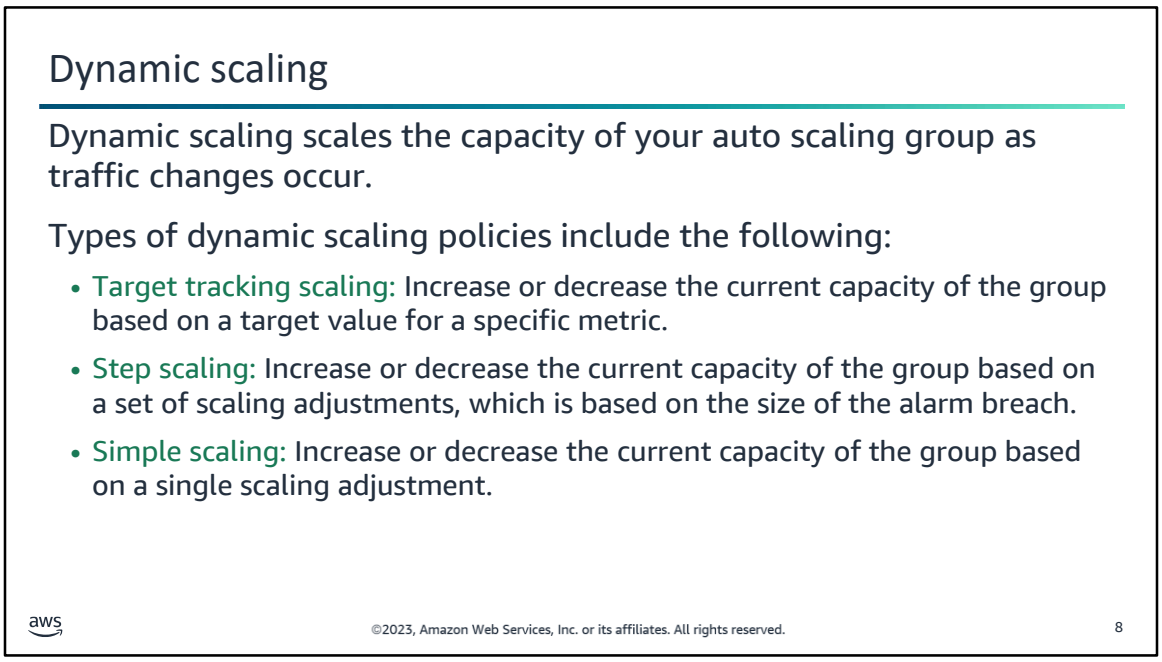


By scaling based on a schedule, you can scale your application in response to predictable load changes. For example, suppose that every week, the traffic to your web application starts to increase on Wednesday, remains partially high on Thursday, and starts to decrease by Friday. In these situations, you can plan your scaling activities based on the predictable traffic patterns of your web application.

The following are the steps to configure an auto scaling group to scale based on a schedule:

1. Create a scheduled action, which tells Amazon EC2 Auto Scaling to perform a scaling action at specified times.
2. Specify the start time when the scaling action should take effect, and the new minimum, maximum, and desired sizes for the scaling action.
3. At the specified time, Amazon EC2 Auto Scaling updates the group to match the capacity values specified in the scaling action.

For more information, see “Scheduled Scaling for Amazon EC2 Auto Scaling” at https://docs.aws.amazon.com/autoscaling/ec2/userguide/ec2-auto-scaling-scheduled-scaling.html.



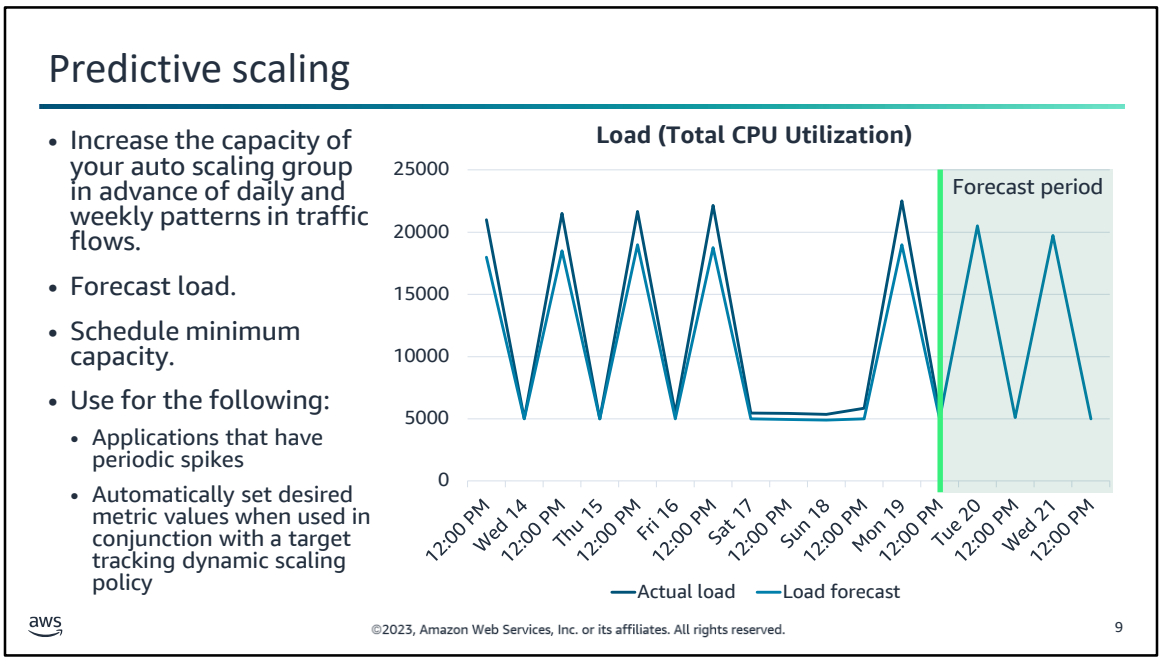
Automatic scaling can be configured to be dynamic. For example, suppose that you have a web application that currently runs on three instances. You do not want the CPU utilization of the auto scaling group to exceed 70 percent for more than 2 minutes. You can configure your auto scaling group to scale automatically to meet this need. The policy type determines how the scaling action is performed.

Amazon EC2 Auto Scaling supports the following types of scaling policies:

- Target tracking scaling: Increase or decrease the current capacity of the group based on a target value for a specific metric.
    - This type of scaling is similar to the way that your thermostat maintains the temperature of your home. You select a temperature, and the thermostat does the rest of the work.
    - In the case of Amazon EC2, you set the metric so that Amazon EC2 Auto Scaling monitors it. Amazon EC2 Auto Scaling does the rest of the work.
- Step scaling: Increase or decrease the current capacity of the group based on a set of scaling adjustments, which are known as step adjustments. These scaling adjustments vary based on the size of the alarm breach.
    - Amazon EC2 Auto Scaling does not support cooldown periods for step scaling policies.
    - Therefore, you cannot specify a cooldown period for these policies, and the default cooldown period for the group does not apply.
- Simple scaling: Increase or decrease the current capacity of the group based on a single scaling adjustment.
    - Simple scaling supports a cooldown period between each scaling activity.

Your scaling might be based on a utilization metric that increases or decreases proportionally to the number of instances in an auto scaling group. If so, the recommended approach is to use target tracking scaling policies. Otherwise, it is recommended that you use step scaling policies.

For more information, see “Dynamic Scaling for Amazon EC2 Auto Scaling” at https://docs.aws.amazon.com/autoscaling/ec2/userguide/as-scale-based-on-demand.html.



AWS also provides predictive scaling. Predictive scaling uses machine learning models to predict your expected traffic (and Amazon EC2 usage), including daily and weekly patterns. These predictions use data that is collected from your actual Amazon EC2 usage and data points that are drawn from your own observations. The model needs historical data from at least 1 day to start making predictions. The model is re-evaluated every 24 hours to create a forecast for the next 48 hours.

Predictive scaling removes the need for manually adjusting auto scaling parameters over time, which makes auto scaling less complex to configure and consume.

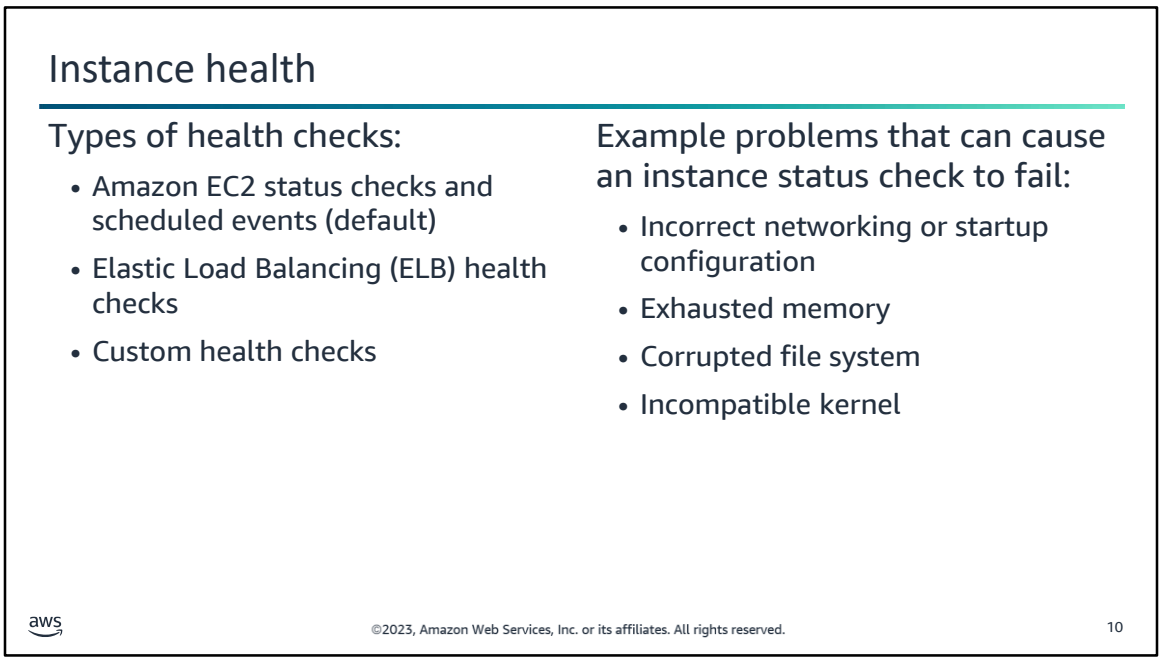
To get started, go to the AWS Management Console. In the upper-right, choose Services, and then choose Management & Governance. From this menu, choose AWS Auto Scaling. Once you are on this page, you can create a scaling plan for Amazon EC2 resources that includes predictive scaling. After you turn on predictive scaling, you can visualize the forecasted traffic and the generated scaling actions.

You can even use predictive scaling with dynamic scaling. Predictive scaling works by forecasting load and scheduling minimum capacity; dynamic scaling uses target tracking to adjust a designated CloudWatch metric to a specific target. The two models work well together because of the scheduled minimum capacity that the predictive scaling already has set.

Predictive scaling is a good choice for websites and applications that experience periodic traffic spikes. It is not designed to help in situations where spikes in load are not cyclic or predictable.

For more information, see the following resources:

- Predictive Scaling Settings at https://docs.aws.amazon.com/autoscaling/plans/userguide/gs-specify-custom-settings.html#gs-customize-predictive-scaling
- Monitoring and Evaluating Forecasts at https://docs.aws.amazon.com/autoscaling/plans/userguide/gs-create-scaling-plan.html#gs-monitoring-forecasts



Amazon EC2 Auto Scaling periodically checks the health status of all instances within the auto scaling group to make sure that they're running and in good condition. If Amazon EC2 Auto Scaling detects that an instance is no longer in the running state, it is treated as an immediate failure, marks the instance as unhealthy, and replaces it.

Amazon EC2 Auto Scaling can determine the health status of an instance by using one or more of the following health checks:

- Amazon EC2 status checks and scheduled events
    - Checks that the instance is running
    - Checks for underlying hardware or software issues that might impair the instance
- ELB health checks
    - Checks whether the load balancer reports the instance as healthy, which confirms whether the instance is available to handle requests
- Custom health checks
    - Checks for any other problems that might indicate instance health issues according to your custom health checks

By default, Amazon EC2 Auto Scaling uses EC2 instance status checks. If an auto scaling group is behind a load balancer, either the load balancer's instance checks or the EC2 instance checks are used for health monitoring depending on which is turned on.

For more information on health checks, see “Health Checks for Auto Scaling Instances” at https://docs.aws.amazon.com/autoscaling/ec2/userguide/ec2-auto-scaling-health-checks.html.

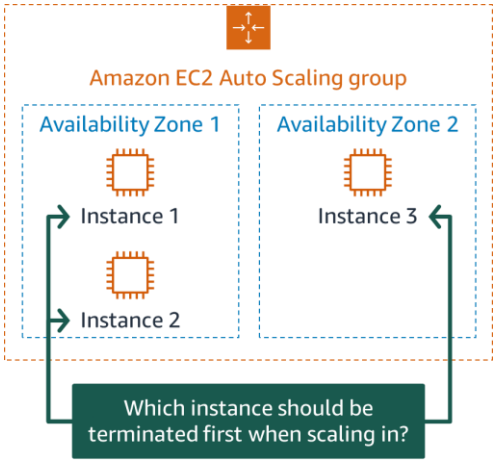
For more information about instance status checks, see “Status Checks for Your Instances” at https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/monitoring-system-instance-status-check.html.

Termination policy

This type of policy determines which instance is terminated when scaling in.

The following are examples of a termination policy:

- Default (Availability Zone with the largest number of instances)
- Oldest instance
- Newest instance
- Oldest launch template
- Closest to next instance billable hour



A termination policy specifies the criteria that Amazon EC2 Auto Scaling uses to choose an instance for termination when scaling in. There are various predefined termination policies, including a default policy.

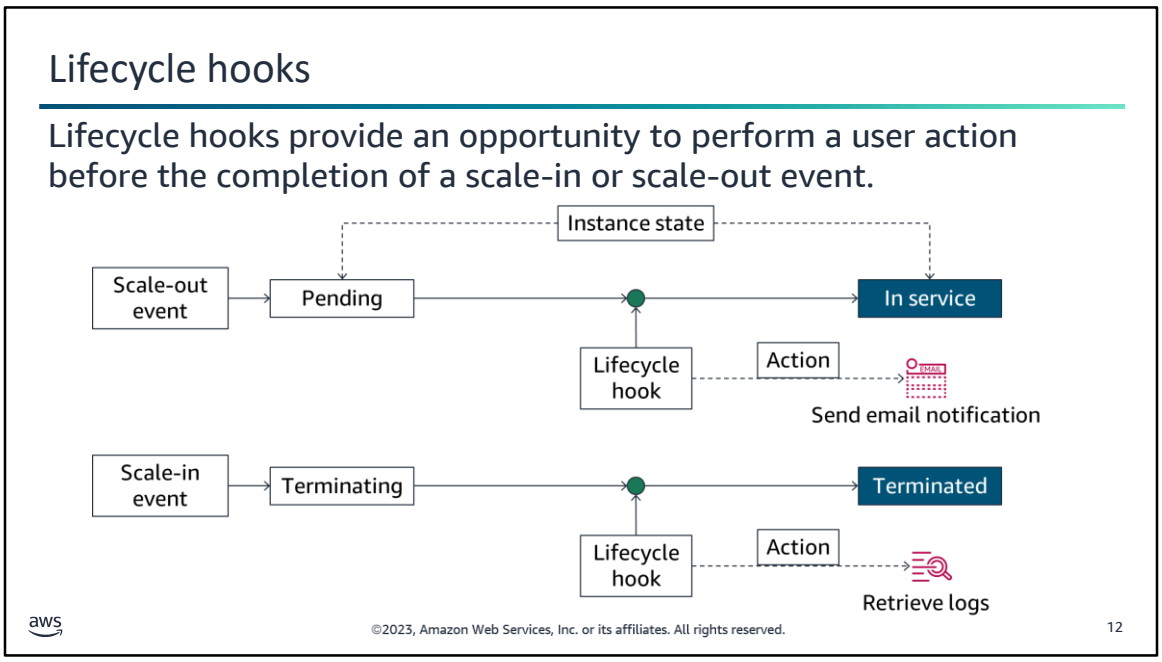
The default termination policy is designed to help ensure that your instances span Availability Zones evenly for high availability. When Amazon EC2 Auto Scaling terminates instances, it first determines which Availability Zones have the most instances, and it finds at least one instance that is not protected from scale in. For example, the diagram of the auto scaling group contains two Availability Zones. Availability Zone 1 has two instances, and Availability Zone 2 has one instance. With the default termination policy, an eligible instance from Availability Zone 1 would be terminated. The default termination policy is typically sufficient for most situations. However, you have the option to select a different one to better suit your needs.

Amazon EC2 Auto Scaling provides other predefined termination policies, including the following:

- **OldestInstance:** This policy terminates the oldest instance in the group. This option is useful when you are upgrading the instances in the auto scaling group to a new EC2 instance type. You can gradually replace instances of the old type with instances of the new type.
- **NewestInstance:** This policy terminates the newest instance in the group. This policy is useful when you're testing a new launch template but do not want to keep it in production.
- **OldestLaunchTemplate:** This policy terminates instances that have the oldest launch template. This choice is good when you are updating a group and phasing out the instances from a previous template configuration.
- **ClosestToNextInstanceHour:** This policy terminates instances that are closest to the next billing hour. Using this policy is a good way to maximize the use of your instances and manage your Amazon EC2 usage costs.

For more information, see the following resources:

- Controlling which auto scaling Instances Terminate During Scale In at https://docs.aws.amazon.com/autoscaling/ec2/userguide/as-instance-termination.html
- Work with Amazon EC2 Auto Scaling Termination Policies at https://docs.aws.amazon.com/autoscaling/ec2/userguide/ec2-auto-scaling-termination-policies.html

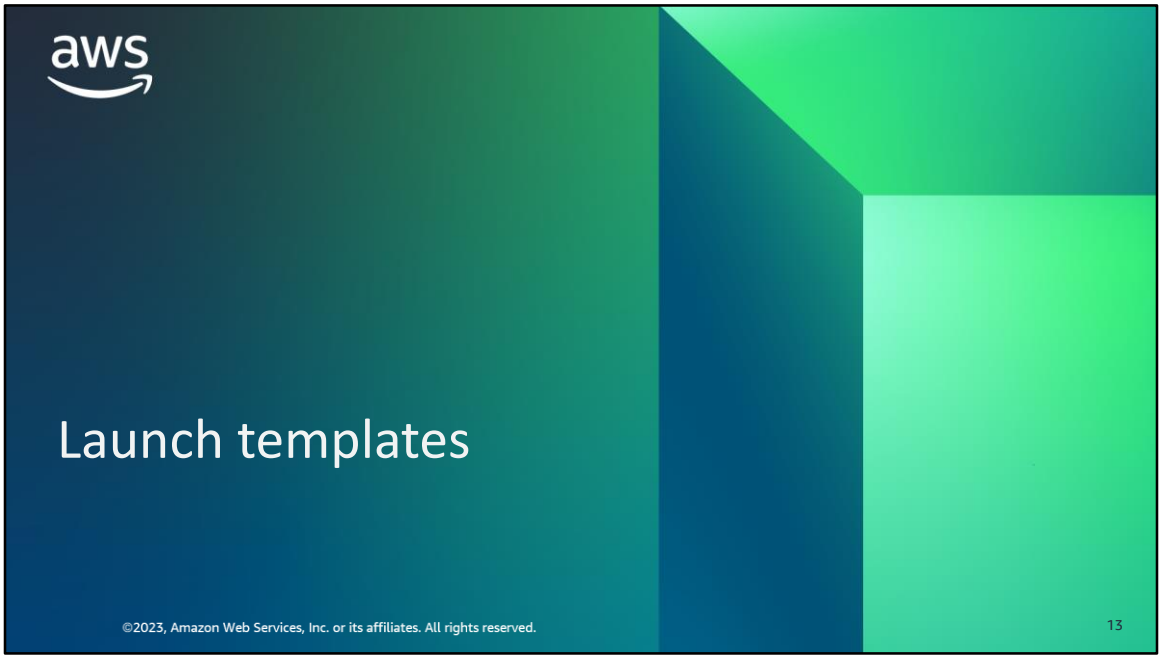


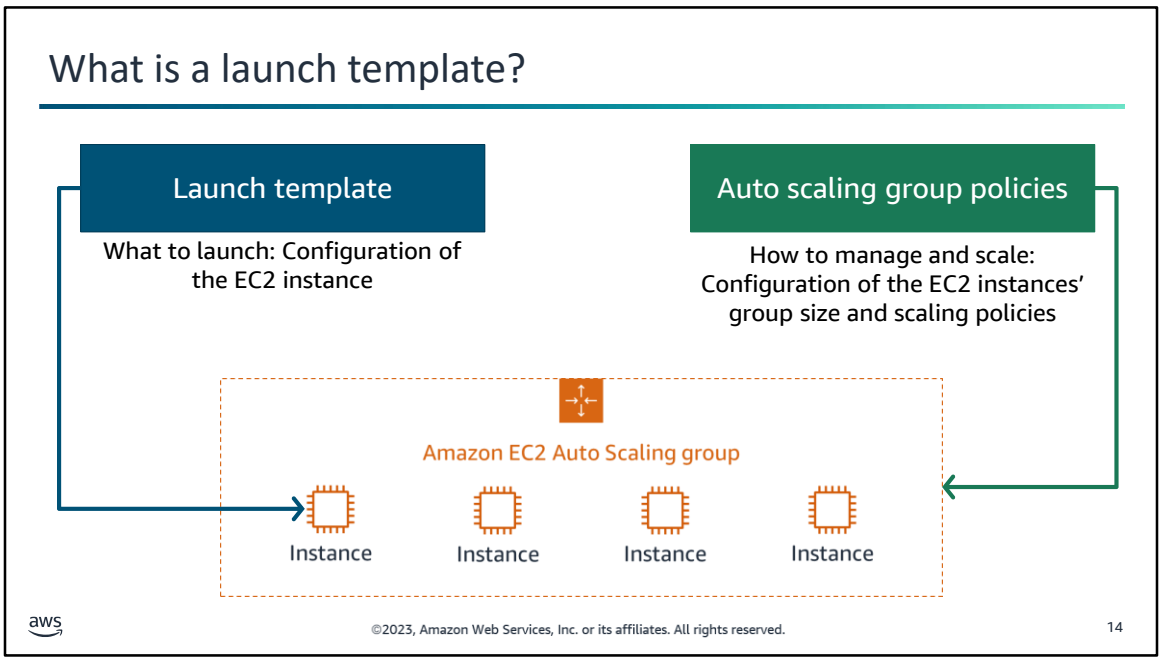
In some cases, you might want to intervene before an Amazon EC2 Auto Scaling action adds to or subtracts from your Amazon EC2 Auto Scaling group. Amazon EC2 Auto Scaling group lifecycle hooks give you this flexibility.

The diagram illustrates the following:

- During a scale-out event, Amazon EC2 Auto Scaling begins to launch a new instance and puts it in a *Pending* state. At this point, you can use a lifecycle hook to perform an action on the instance. For example, you can use the hook to send an email notification to a systems administrator who could then install security software updates on the instance. When the administrator is finished, Amazon EC2 Auto Scaling completes the launching process and puts the instance in an *In service* state.
- Similarly, during a scale-in event, Amazon EC2 Auto Scaling begins terminating an instance and puts it in a *Terminating* state. At this point, you can use a lifecycle hook to perform an action on the instance before it is terminated. For example, you could have the hook run a script to retrieve and back up the instance logs for audit purposes. When the script is finished, Amazon EC2 Auto Scaling completes the termination process and puts the instance in a *Terminated* state.

For more information, see “How Lifecycle Hooks Work” at https://docs.aws.amazon.com/autoscaling/ec2/userguide/lifecycle-hooks-overview.html.





A launch template specifies instance configuration information (including the ID of the AMI, the instance type, a key pair, security groups, and other parameters) and gives you the option to have multiple versions. The Amazon EC2 Auto Scaling group then maintains the right number of EC2 instances defined by the launch template depending on your needs. Together, both the launch template and the auto scaling group policies determine what to launch within the auto scaling group and how to manage them.

You can create an auto scaling group using a launch configuration, but AWS strongly recommends that you use a launch template rather than launch configurations.

Creating a launch template

When configuring a launch template, the parameters you can specify include the following:

- Amazon Machine Image (AMI)
- Instance type
- Instance key pair
- Security group
- Storage
- AWS Identity and Access Management (IAM) roles
- User data
- Tagging

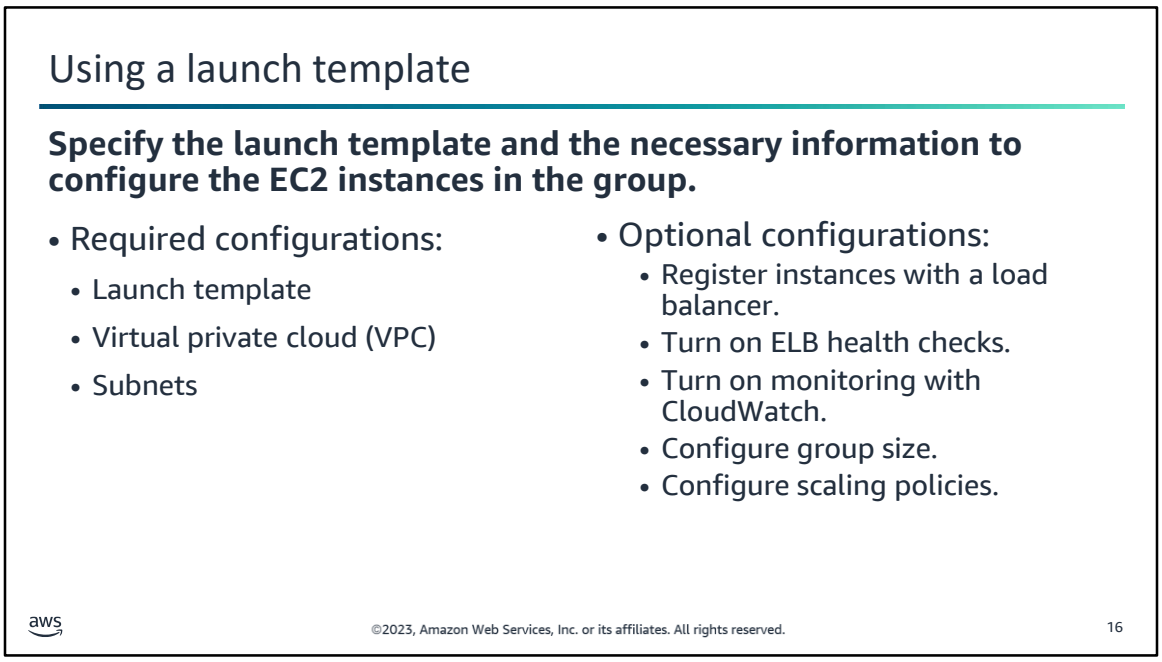
The following steps describe how to configure your launch template:

- Specify the AMI from which to launch the instances.
- Choose an instance type that is compatible with the AMI that you specify.
- Specify the key pair to use when connecting to instances (for example, using SSH).
- Add one or more security groups to allow network access to the instances.
- Specify whether to attach additional volumes to each instance.
- Add custom tags (key-value pairs) to the instances and volumes.

When you create a launch template, all parameters are optional and can be used to launch an instance. However, if you are creating a launch template for an auto scaling group, the ID of the AMI and an instance type are required. If it does not specify an AMI, you cannot add the AMI when you create your auto scaling group.

Keep in mind that you can also create different versions of your template. For example, you can create a launch template that defines a base configuration without an AMI or user data script. After you create your launch template, you can create a new version and add the AMI and user data that has the latest version of your application for testing. This results in two versions of the launch template.

For further documentation, see “Create a Launch Template for an Auto Scaling Group” at https://docs.aws.amazon.com/autoscaling/ec2/userguide/create-launch-template.html.



Once you have created the launch template, you can use the launch template to configure an auto scaling group.

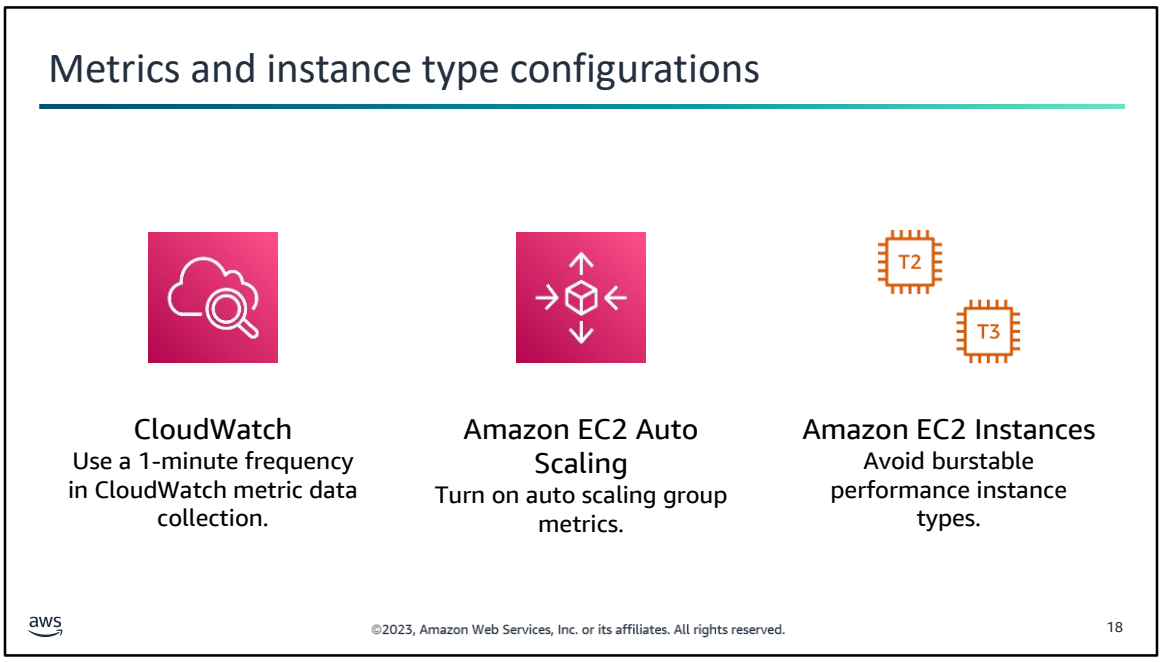
To create the auto scaling group based on the launch template, you must do the following:
- Choose the launch template, and specify version preferences.
- Choose instance launch options. For example, you must create the auto scaling group in the same virtual private cloud (VPC) as the security group specified in the launch template.
- Choose one or more subnets in the specified VPC. To support high availability, use subnets in multiple Availability Zones.

You can also configure (optional) advanced settings, such as the following:
- Register the EC2 instances with a load balancer. This setting makes it possible for traffic to be distributed across multiple EC2 instances depending on the workload and their health.
- Turn on ELB health checks. These checks help ensure that the distributed traffic is sent to only healthy instances.
- Turn on monitoring with CloudWatch group metrics collection. Monitoring increases your visibility of metrics that can be indicators of a potential issue, such as the number of terminating instances or number of pending instances.
- Configure group size policies. This setting determines the minimum and maximum capacity of the group. As demand grows, the group will scale out to use more instances. If demand decreases, the group will scale in and terminate instances that are not needed anymore.
- Configure scaling policies. This setting includes selecting target tracking scaling policies to automatically scale the size of the auto scaling group. It also determines the health status of the instances provided by the EC2 instance, the load balancer, or custom health checks. If an instance is unhealthy, it will be replaced.

To update the configuration of the EC2 instances after the group is created, you can create a new version of the launch template. After you change the launch template for an auto scaling group, any new instances are launched using the new configuration options, but existing instances are not affected.

For more information, see “Create an Auto Scaling Group Using a Launch Template” at https://docs.aws.amazon.com/autoscaling/ec2/userguide/create-asg-launch-template.html.

Best practices



Use a 1-minute frequency in CloudWatch metric data collection. When you create a launch template or launch configuration, turn on detailed monitoring to get CloudWatch metric data for EC2 instances at a 1-minute frequency to help ensure a faster response to load changes. Scaling on metrics with a 5-minute frequency can result in a slower response time and scaling on stale metric data.

For more information, see "Configure Monitoring for Auto Scaling Instances" in the *Amazon EC2 Auto Scaling User Guide* at https://docs.aws.amazon.com/autoscaling/ec2/userguide/enable-as-instance-metrics.html.

Turn on auto scaling group metrics. Otherwise, actual capacity data is not shown in the capacity forecast graphs that are available on completion of the Create Scaling Plan wizard.

For more information, see "Monitor CloudWatch Metrics for Your Auto Scaling Groups and Instances" at https://docs.aws.amazon.com/autoscaling/ec2/userguide/ec2-auto-scaling-cloudwatch-monitoring.html.

Avoid burstable performance instance types. Be wary of using a burstable performance instance type. EC2 instances with burstable performance, such as T3 and T2 instances, are designed to provide a baseline level of CPU performance with the ability to burst to a higher level when required by your workload. Depending on the target utilization specified by the scaling plan, you could run the risk of exceeding the baseline and then running out of CPU credits, which limits performance.

For more information, see the following resources:

- Key Concepts and Definitions for Burstable Performance Instances at https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/burstable-credits-baseline-concepts.html.
- Best Practices for Scaling Plans at https://docs.aws.amazon.com/autoscaling/plans/userguide/best-practices-for-scaling-plans.html.

Create a steady-state group

Steady-state group:

- You set an Amazon EC2 Auto Scaling group with the same min, max, and desired values.
- An instance is recreated automatically if it becomes unhealthy or if an Availability Zone fails.
- There is still potential downtime while an instance recycles.

Use case:

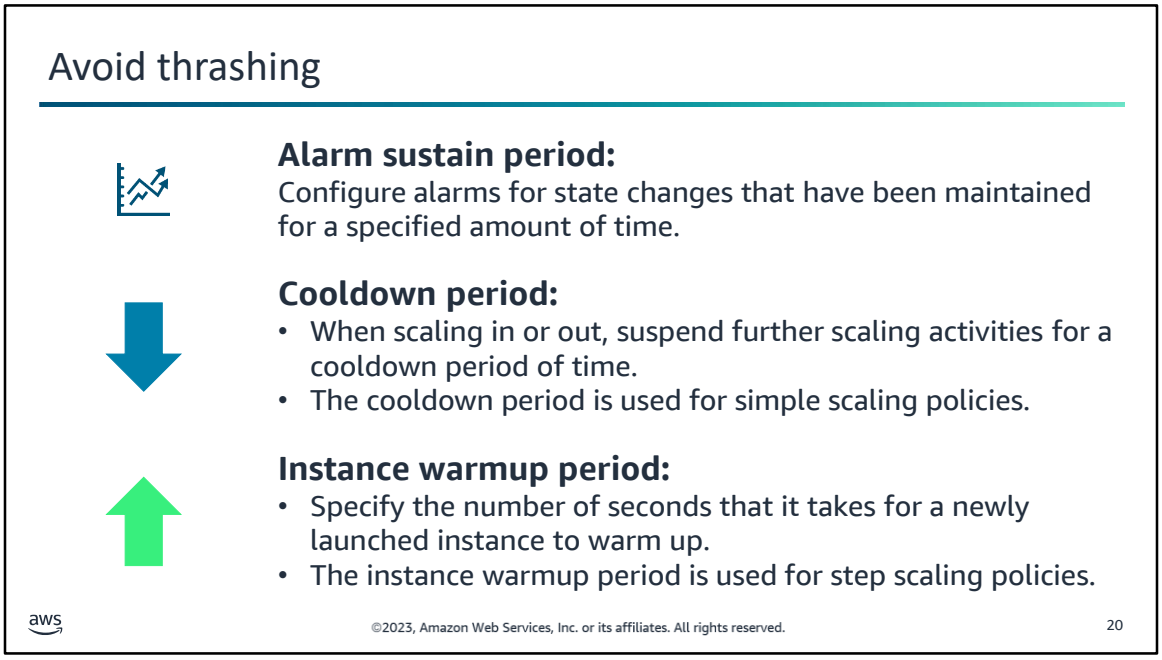
Maintain a steady-state NAT server in each Availability Zone.

With Amazon EC2 Auto Scaling health checks, you can create a steady-state group to help ensure that a single instance is always running. For example, suppose you have a NAT server that you do not want to be a single point of failure in a standard public-private subnet architecture. A steady-state group is useful for this instance.

The following are the steps to create a steady-state group for an instance:

1. Create a launch template that creates the instance.
2. Create an Amazon EC2 Auto Scaling group with a minimum, maximum, and desired size of 1. One of these instances might be is marked as unhealthy because an instance check fails. Alternatively, an external script could mark the instance as unhealthy with a call to the AWS CLI aws autoscaling set-instance-health command.
3. The Amazon EC2 Auto Scaling group will terminate the existing instance and create a new instance from the group's launch template.

Notice that in cases such as deploying a NAT instance, the NAT is still a single point of failure. You can still experience significant downtime while a failed NAT instance is recycling.



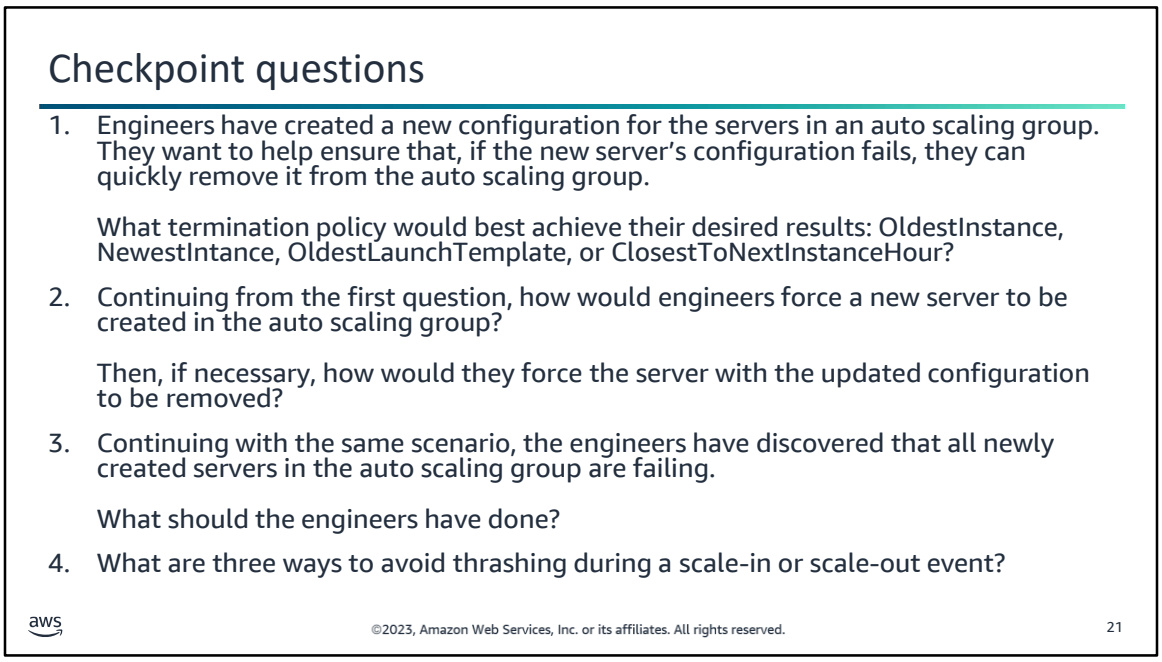
Thrashing is the condition in which there is excessive use of a computer's virtual memory, and the computer is no longer able to service the resource needs of applications that run on it. When you configure automatic scaling, make sure that you avoid thrashing. Thrashing could occur if instances are removed and added—or added and removed—in succession too quickly.

One way to avoid thrashing is to initiate alarm actions for sustained state changes only. Configure launches for state changes that have been maintained for a reasonable period of time. For example, CPU utilization is at 90 percent for 10 minutes.

The Amazon EC2 Auto Scaling cooldown period is another configurable setting that helps to ensure that Amazon EC2 Auto Scaling doesn't launch or terminate additional instances before the previous scaling activity takes effect. For example, suspend scaling for 5 minutes.

When you manually scale your Amazon EC2 Auto Scaling group, the default is not to wait for the cooldown period. However, you can override the default and honor the cooldown period. Note that if an instance becomes unhealthy, Amazon EC2 Auto Scaling does not wait for the cooldown period to complete before replacing the unhealthy instance. Amazon EC2 Auto Scaling supports both default cooldown periods and scaling-specific cooldown periods. It supports cooldown periods when using simple scaling policies but not when using target tracking policies, step scaling policies, or scheduled scaling.

Also, with step scaling policies, you can specify the number of seconds that it takes for a newly launched instance to warm up. For example, warmup takes 5 minutes. Until its specified warmup time has expired, an instance is not counted toward the aggregated metrics of the auto scaling group. When scaling out, AWS also does not consider instances that are warming up as part of the current capacity of the group. Therefore, multiple alarm breaches that fall in the range of the same step adjustment result in a single scaling activity. This step helps ensure that Amazon EC2 Auto Scaling doesn't add more instances than you need.



The answers to the questions are as follows:

1. Engineers have created a new configuration for the servers in an auto scaling group. They want to help ensure that, if the new server's configuration fails, they can quickly remove it from the auto scaling group.

   What termination policy would best achieve their desired results: OldestInstance, NewestIntance, OldestLaunchTemplate, or ClosestToNextInstanceHour?

   The engineers should use the NewestInstance policy. Any newly launched EC2 servers would have the latest configuration. Because those servers would also be the newest instances, they would be the first removed during a scale-in event.

2. Continuing from the first question, how would engineers force a new server to be created in the auto scaling group?

   Then, if necessary, how would they force the server with the updated configuration to be removed?

   The engineers could force new servers to be created by adjusting the minimum number of instances configuration attribute on the auto scaling group. When the value is increased, the auto scaling group launches new instances. Forcing the removal of EC2 instances requires adjusting the minimum value to a lower value. Because the engineers configured the termination policy to NewestInstance, the last EC2 instance that was launched would be shut down and removed from the group.

3. Continuing with the same scenario, the engineers have discovered that all newly created servers in the auto scaling group are failing.

   What should the engineers have done?

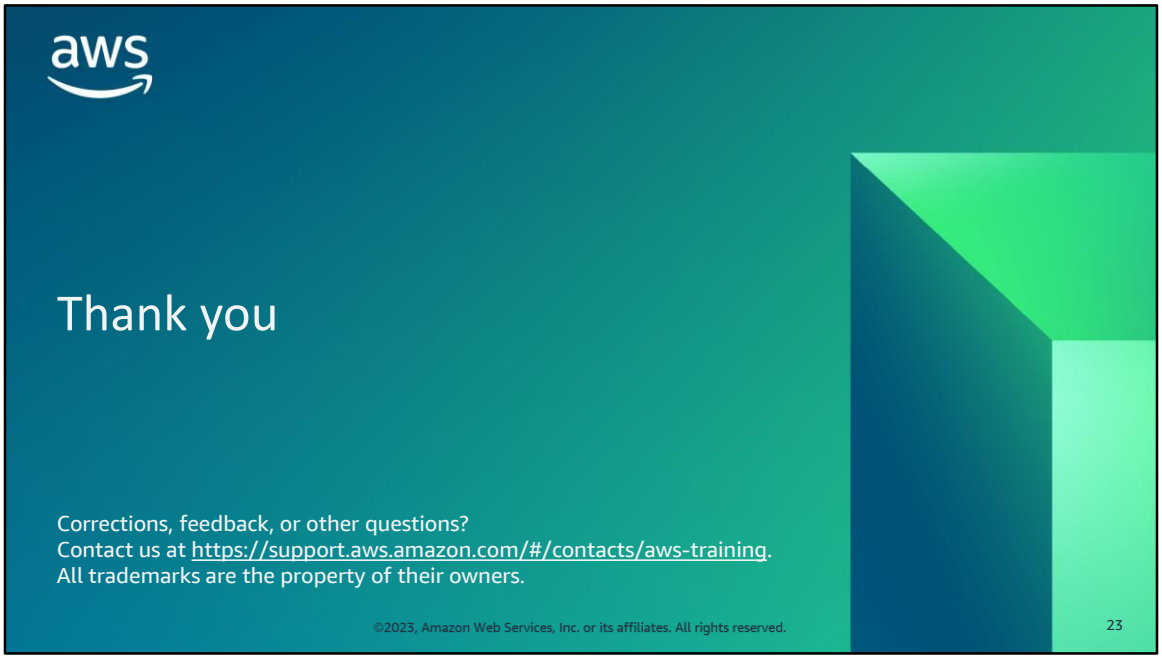
   The engineers should have updated the launch template with the previous value or values. In this case, the configuration is still using the new AMI that failed. The engineers needed to update the launch template with the older AMI.

4. What are three ways to avoid thrashing during a scale-in or scale-out event?

   The following are three ways to avoid thrashing:
   - Configure an alarm sustain period. This option requires the alarm state to exist for a specified period of time before an event occurs. In case of a transient spike of only a few minutes, the rule would not allow scaling to occur.
   - A cooldown period helps ensure that another rule does not take effect for some period of time. This option helps ensure that time is allotted for newly instantiated servers to take on the additional load before an additional scaling event occurs. This is also applicable during a scale-in event.
   - A warmup period gives a new launched EC2 instance time to completely start. This option gives the instance time to start and take on a workload before demand is re-evaluated for additional scale-in or scale-out events.

Key ideas

- Amazon EC2 Auto Scaling helps maintain application availability by automatically adding or removing EC2 instances according to defined conditions.
- Amazon EC2 Auto Scaling consists of three parts:
  - Launch template
  - Auto scaling group
  - Scaling policies
- Scaling can be based on instance health, Amazon CloudWatch alarms, and time schedule or past usage (prediction).
- Thrashing is a condition where scaling occurs and instances are created or removed too fast. To avoid thrashing, use alarm sustain, cooldown, and warmup periods.



Thank you

Corrections, feedback, or other questions?
Contact us at https://support.aws.amazon.com/#/contacts/aws-training.
All trademarks are the property of their owners.