Scaling Plans User Guide

# AWS Auto Scaling

# AWS Auto Scaling: Scaling Plans User Guide

# Table of Contents

# What is a scaling plan?

Use a *scaling plan* to configure auto scaling for related or associated scalable resources in a matter of minutes. For example, you can use tags to group resources in categories such as production, testing, or development. Then, you can search for and set up scaling plans for scalable resources that belong to each category. Or, if your cloud infrastructure includes AWS CloudFormation, you can define stack templates to use to create collections of resources. Then, create a scaling plan for the scalable resources that belong to each stack.

## Supported resources

AWS Auto Scaling supports the use of scaling plans for the following services and resources:

- **Amazon Aurora** – Increase or decrease the number of Aurora read replicas that are provisioned for an Aurora DB cluster.
- **Amazon EC2 Auto Scaling** – Launch or terminate EC2 instances by increasing or decreasing the desired capacity of an Auto Scaling group.
- **Amazon Elastic Container Service** – Increase or decrease the desired task count in Amazon ECS.
- **Amazon DynamoDB** – Increase or decrease the provisioned read and write capacity of a DynamoDB table or a global secondary index.
- **Spot Fleet** – Launch or terminate EC2 instances by increasing or decreasing the target capacity of a Spot Fleet.

## Scaling plan features and benefits

Scaling plans provide the following features and benefits:

- **Resource discovery** – AWS Auto Scaling provides automatic resource discovery to help find resources in your application that can be scaled.
- **Dynamic scaling** – Scaling plans use the Amazon EC2 Auto Scaling and Application Auto Scaling services to adjust capacity of scalable resources to handle changes in traffic or workload. Dynamic scaling metrics can be standard utilization or throughput metrics, or custom metrics.
- **Built-in scaling recommendations** – AWS Auto Scaling provides scaling strategies with recommendations that you can use to optimize for performance, costs, or a balance between the two.

- **Predictive scaling** – Scaling plans also support predictive scaling for Auto Scaling groups. This helps to scale your Amazon EC2 capacity faster when there are regularly occurring spikes.

> ⚠ **Important**
>
> If you use scaling plans only for predictive scaling, we strongly recommend that you set predictive scaling policies directly on your Auto Scaling groups instead. This recently introduced option offers more features, such as using metrics aggregations to create new custom metrics or retain historical metric data across blue/green deployments. For more information, see Predictive scaling for Amazon EC2 Auto Scaling in the *Amazon EC2 Auto Scaling User Guide*.
>
> For a guide for migrating from scaling plans to Amazon EC2 Auto Scaling predictive scaling policies, see Migrate your scaling plan.

## How to get started

Use the following resources to help you create and use a scaling plan:

- How scaling plans work

- Best practices for scaling plans

- Getting started with scaling plans

## Work with scaling plans

You can create, access, and manage your scaling plans using any of the following interfaces:

- **AWS Management Console** – Provides a web interface that you can use to access your scaling plans. If you've signed up for an AWS account, you can access your scaling plans by signing into the AWS Management Console, using the search box on the navigation bar to search for **AWS Auto Scaling**, and then choosing **AWS Auto Scaling**.

- **AWS Command Line Interface (AWS CLI)** – Provides commands for a broad set of AWS services, and is supported on Windows, macOS, and Linux. To get started, see AWS Command Line Interface User Guide. For more information, see autoscaling-plans in the *AWS CLI Command Reference*.

- **AWS Tools for Windows PowerShell** – Provides commands for a broad set of AWS products for those who script in the PowerShell environment. To get started, see the AWS Tools for Windows PowerShell User Guide. For more information, see the AWS Tools for PowerShell Cmdlet Reference.

- **AWS SDKs** – Provides language-specific API operations and takes care of many of the connection details, such as calculating signatures, handling request retries, and handling errors. For more information, see AWS SDKs.

- **HTTPS API** – Provides low-level API actions that you call using HTTPS requests. For more information, see the AWS Auto Scaling API Reference.

- **AWS CloudFormation** – Supports creating scaling plans using CloudFormation templates. For more information, see the AWS::AutoScalingPlans::ScalingPlan reference in the *AWS CloudFormation User Guide*.

# Regional availability

The AWS Auto Scaling API is available in several AWS Regions and it provides an endpoint for each of these Regions. For a list of all the Regions and endpoints where the API is currently available, see AWS Auto Scaling endpoints and quotas in the *AWS General Reference*.

# Pricing

All scaling plan features are enabled for your use. The features are provided at no additional charge beyond the service fees for CloudWatch and the other AWS Cloud resources that you use.

> **ⓘ Note**
>
> The predictive scaling feature relies on the CloudWatch GetMetricData operation to collect historical metric data for capacity forecasting, which incurs costs. However, if you enable predictive scaling with an Amazon EC2 Auto Scaling scaling policy instead of a scaling plan, there are no charges for calls to `GetMetricData`.

# How scaling plans work

AWS Auto Scaling lets you use scaling plans to configure a set of instructions for scaling your resources. If you work with AWS CloudFormation or add tags to scalable resources, you can set up scaling plans for different sets of resources, per application. The AWS Auto Scaling console provides recommendations for scaling strategies customized to each resource. After you create your scaling plan, it combines dynamic scaling and predictive scaling methods together to support your scaling strategy.

**What is a scaling strategy?**

The scaling strategy tells AWS Auto Scaling how to optimize the utilization of resources in your scaling plan. You can optimize for availability, for cost, or a balance of both. Alternatively, you can also create your own custom strategy, per the metrics and thresholds you define. You can set separate strategies for each resource or resource type.



**What is dynamic scaling?**

Dynamic scaling creates target tracking scaling policies for the resources in your scaling plan. These scaling policies adjust resource capacity in response to live changes in resource utilization. The intention is to provide enough capacity to maintain utilization at the target value specified by the scaling strategy. This is similar to the way that your thermostat maintains the temperature of your home. You choose the temperature and the thermostat does the rest.
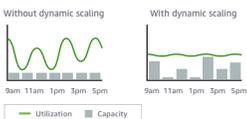


For example, you can configure your scaling plan to keep the number of tasks that your Amazon Elastic Container Service (Amazon ECS) service runs at 75 percent of CPU. When the CPU utilization of your service exceeds 75 percent (meaning that more than 75 percent of the CPU that is reserved for the service is being used), then your scaling policy adds another task to your service to help out with the increased load.

**What is predictive scaling?**

Predictive scaling uses machine learning to analyze each resource's historical workload and regularly forecasts the future load. This is similar to how weather forecasts work. Using the

forecast, predictive scaling generates scheduled scaling actions to make sure that the resource capacity is available before your application needs it. Like dynamic scaling, predictive scaling works to maintain utilization at the target value specified by the scaling strategy.
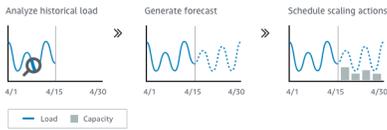


For example, you can enable predictive scaling and configure your scaling strategy to keep the average CPU utilization of your Auto Scaling group at 50 percent. Your forecast calls for traffic spikes to occur every day at 8:00. Your scaling plan creates the future scheduled scaling actions to make sure that your Auto Scaling group is ready to handle that traffic ahead of time. This helps keep the application performance constant, with the aim of always having the capacity required to maintain resource utilization as close to 50 percent as possible at all times.

The following are the key concepts for understanding predictive scaling:

- **Load forecasting**: AWS Auto Scaling analyzes up to 14 days of history for a specified load metric and forecasts the future demand for the next two days. This data is available in one-hour intervals and is updated daily.

- **Scheduled scaling actions**: AWS Auto Scaling schedules the scaling actions that proactively increases and decreases capacity to match the load forecast. At the scheduled time, AWS Auto Scaling updates the minimum capacity with the value specified by the scheduled scaling action. The intention is to maintain resource utilization at the target value specified by the scaling strategy. If your application requires more capacity than is forecast, dynamic scaling is available to add additional capacity.

- **Maximum capacity behavior**: Minimum and maximum capacity limits for auto scaling apply to each resource. However, you can control whether your application can increase capacity beyond the maximum capacity when the forecast capacity is higher than the maximum capacity.

> ⓘ **Note**
>
> You can now use the predictive scaling policies of Auto Scaling groups instead. For more information, see Predictive scaling for Amazon EC2 Auto Scaling in the *Amazon EC2 Auto Scaling User Guide*.

# Best practices for scaling plans

The following best practices can help you make the most of scaling plans:

- When you create a launch template or launch configuration, enable detailed monitoring to get CloudWatch metric data for EC2 instances at a one-minute frequency because that ensures a faster response to load changes. Scaling on metrics with a five-minute frequency can result in a slower response time and scaling on stale metric data. By default, EC2 instances are enabled for basic monitoring, which means metric data for instances is available at five-minute intervals. For an additional charge, you can enable detailed monitoring to get metric data for instances at a one-minute frequency. For more information, see Configure monitoring for Auto Scaling instances in the *Amazon EC2 Auto Scaling User Guide*.

- We also recommend that you enable 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 Monitoring CloudWatch metrics for your Auto Scaling groups and instances in the *Amazon EC2 Auto Scaling User Guide*.

- Check which instance type your Auto Scaling group uses and be wary of using a burstable performance instance type. Amazon 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 CPU credits and baseline performance for burstable performance instances. To configure these instances as `unlimited`, see Using an Auto Scaling group to launch a burstable performance instance as Unlimited in the *Amazon EC2 User Guide*.

# Other considerations

> **ⓘ Note**
>
> There is a more recent version of predictive scaling, released in May, 2021. Some features introduced in this version are not available in scaling plans, and you must use a predictive scaling policy set directly on the Auto Scaling group to access those features. For more information, see Predictive scaling for Amazon EC2 Auto Scaling in the *Amazon EC2 Auto Scaling User Guide*.

Keep the following additional considerations in mind:

- Predictive scaling uses load forecasts to schedule capacity in the future. The quality of the forecasts varies based on how cyclical the load is and the applicability of the trained forecasting model. Predictive scaling can be run in forecast only mode to assess the quality of the forecasts and the scaling actions created by the forecasts. You can set the predictive scaling mode to **Forecast only** when you create the scaling plan and then change it to **Forecast and scale** when you're finished assessing the forecast quality. For more information, see [Predictive scaling settings](#) and [Monitoring and evaluating forecasts](#).
- If you choose to specify different metrics for predictive scaling, you must ensure that the scaling metric and load metric are strongly correlated. The metric value must increase and decrease proportionally to the number of instances in the Auto Scaling group. This ensures that the metric data can be used to proportionally scale out or in the number of instances. For example, the load metric is total request count and the scaling metric is average CPU utilization. If the total request count increases by 50 percent, the average CPU utilization should also increase by 50 percent, provided that capacity remains unchanged.
- Before creating your scaling plan, you should delete any previously scheduled scaling actions that you no longer need by accessing the consoles they were created from. AWS Auto Scaling does not create a predictive scaling action that overlaps an existing scheduled scaling action.
- Your customized settings for minimum and maximum capacity, along with other settings used for dynamic scaling, show up in other consoles. However, we recommend that after you create a scaling plan, you do not modify these settings from other consoles because your scaling plan does not receive the updates from other consoles.
- Your scaling plan can contain resources from multiple services, but each resource can be in only one scaling plan at a time.

# Avoiding the ActiveWithProblems error

An "ActiveWithProblems" error can occur when a scaling plan is created, or resources are added to a scaling plan. The error occurs when the scaling plan is active, but the scaling configuration for one or more resources could not be applied.

Usually, this happens because a resource already has a scaling policy or an Auto Scaling group does not meet the minimum requirements for predictive scaling.

If any of your resources already have scaling policies from various service consoles, AWS Auto Scaling does not overwrite these other scaling policies or create new ones by default. You can

optionally delete the existing scaling policies and replace them with target tracking scaling policies created from the AWS Auto Scaling console. You do this by enabling the **Replace external scaling policies** setting for each resource that has scaling policies to overwrite.

With predictive scaling, we recommend waiting 24 hours after creating a new Auto Scaling group to configure predictive scaling. At minimum, there must be 24 hours of historical data to generate the initial forecast. If the group has less than 24 hours of historical data and predictive scaling is enabled, then the scaling plan can't generate a forecast until the next forecast period, after the group has collected the required amount of data. However, you can also edit and save the scaling plan to restart the forecast process as soon as the 24 hours of data is available.

# Getting started with scaling plans

Before you create a scaling plan for use with your application, review your application thoroughly as it runs in the AWS Cloud. Take note of the following:

- Whether you have existing scaling policies created from other consoles. You can replace the existing scaling policies, or you can keep them (without being allowed to make any changes to their values) when you create your scaling plan.

- The target utilization that makes sense for each scalable resource in your application based on the resource as a whole. For example, the amount of CPU that the EC2 instances in an Auto Scaling group are expected to use compared to their available CPU. Or for a service like DynamoDB that uses a provisioned throughput model, the amount of read and write activity that a table or index is expected to use compared to the available throughput. In other words, the ratio of consumed to provisioned capacity. You can change the target utilization at any time after you create your scaling plan.

- How long it takes to launch and configure a server. Knowing this helps you configure a window for each EC2 instance to warm up after launching to ensure that a new server isn't launched while the previous one is still launching.

- Whether the metric history is sufficiently long to use with predictive scaling (if using newly created Auto Scaling groups). In general, having a full 14 days of historical data translates into more accurate forecasts. The minimum is 24 hours.

The better you understand your application, the more effective you can make your scaling plan.

The following tasks help you become familiar with scaling plans. You will create a scaling plan for a single Auto Scaling group and enable predictive scaling and dynamic scaling.

**Tasks**

- [Step 1: Find your scalable resources](#)
- [Step 2: Specify the scaling strategy](#)
- [Step 3: Configure advanced settings (optional)](#)
- [Step 4: Create your scaling plan](#)
- [Step 5: Clean up](#)
- [Step 6: Next steps](#)

# Step 1: Find your scalable resources

This section includes a hands-on introduction to creating scaling plans in the AWS Auto Scaling console. If this is your first scaling plan, we recommend that you start by creating a sample scaling plan using an Amazon EC2 Auto Scaling group.

## Prerequisites

To practice using a scaling plan, create an Auto Scaling group. Launch at least one Amazon EC2 instance in the Auto Scaling group. For more information, see Getting started with Amazon EC2 Auto Scaling in the *Amazon EC2 Auto Scaling User Guide.*

Use an Auto Scaling group with CloudWatch metrics enabled to have capacity data on the graphs that are available when you complete the **Create Scaling Plan** wizard. For more information, see Enable Auto Scaling group metrics in the *Amazon EC2 Auto Scaling User Guide.*

Generate some load for a few days or more to have CloudWatch metric data available for the predictive scaling feature, if possible.

Verify that you have the permissions required to work with scaling plans. For more information, see Identity and access management for scaling plans.

## Add your Auto Scaling group to your new scaling plan

When you create a scaling plan from the console, it helps you find your scalable resources as a first step. Before you begin, confirm that you meet the following requirements:

- You created an Auto Scaling group and launched at least one EC2 instance, as described in the previous section.

- The Auto Scaling group you created has existed for at least 24 hours.

**To start creating a scaling plan**

1. Open the AWS Auto Scaling console at https://console.aws.amazon.com/awsautoscaling/.

2. On the navigation bar at the top of the screen, choose the same Region that you used when you created your Auto Scaling group.

3. From the welcome page, choose **Get started**.

4.  On the **Find scalable resources** page, do one of the following:

    - Choose **Search by CloudFormation stack**, and then choose the AWS CloudFormation stack to use.

    - Choose **Search by tag**. Then, for each tag, choose a tag key from **Key** and tag values from **Value**. To add tags, choose **Add another row**. To remove tags, choose **Remove**.

    - Choose **Choose EC2 Auto Scaling groups**, and then choose one or more Auto Scaling groups.

    > ⓘ **Note**
    >
    > For an introductory tutorial, choose **Choose EC2 Auto Scaling groups**, and then choose the Auto Scaling group you created.



5.  Choose **Next** to continue with the scaling plan creation process.

## Learn more about discovering your scalable resources

If you have already created a sample scaling plan and would like to create more, see the following scenarios for using a CloudFormation stack or a set of tags in more detail. You can use this section to decide whether to choose the **Search by CloudFormation stack** or **Search by tag** option to discover your scalable resources when using the console to create your scaling plan.

When you choose the **Search by CloudFormation stack** or **Search by tag** option in step 1 of the **Create Scaling Plan** wizard, this makes the scalable resources associated with the stack or set of

tags available to the scaling plan. As you define your scaling plan, you can then choose which of these resources to include or exclude.

## Discovering scalable resources using a CloudFormation stack

When you use CloudFormation, you work with stacks to provision resources. All of the resources in a stack are defined by the stack's template. Your scaling plan adds an orchestration layer on top of the stack that makes it easier to configure scaling for multiple resources. Without a scaling plan, you would need to set up scaling for each scalable resource individually. This means figuring out the order for provisioning resources and scaling policies, and understanding the subtleties of how these dependencies work.

In the AWS Auto Scaling console, you can select an existing stack to scan it for resources that can be configured for automatic scaling. AWS Auto Scaling only finds resources that are defined in the selected stack. It does not traverse through nested stacks.

For your ECS services to be discoverable in a CloudFormation stack, the AWS Auto Scaling console must know which ECS cluster is running the service. This requires that your ECS services be in the same CloudFormation stack as the ECS cluster that is running the service. Otherwise, they must be part of the default cluster. To be identified correctly, the ECS service name must also be unique across each of these ECS clusters.

For more information about CloudFormation, see What is AWS CloudFormation? in the *AWS CloudFormation User Guide*.

## Discovering scalable resources using tags

Tags provide metadata that can be used to discover related scalable resources in the AWS Auto Scaling console, using tag filters.

Use tags to find any of the following resources:

- Aurora DB clusters
- Auto Scaling groups
- DynamoDB tables and global secondary indexes

When you search by more than one tag, each resource must have all of the listed tags to be discovered.

For more information about tagging, read the following documentation.

- Learn how to [tag Aurora clusters](#) in the *Amazon Aurora User Guide*.

- Learn how to [tag Auto Scaling groups](#) in the *Amazon EC2 Auto Scaling User Guide*.

- Learn how to [tag DynamoDB resources](#) in the *Amazon DynamoDB Developer Guide*.

- Learn more about best practices for [tagging AWS resources](#) in the *AWS General Reference*.

# Step 2: Specify the scaling strategy

Use the following procedure to specify scaling strategies for the resources that were found in the previous step.

For each type of resource, AWS Auto Scaling chooses the metric that is most commonly used for determining how much of the resource is in use at any given time. You choose the most appropriate scaling strategy to optimize performance of your application based on this metric. When you enable the dynamic scaling feature and the predictive scaling feature, the scaling strategy is shared between them. For more information, see [How scaling plans work](#).

The following scaling strategies are available:

- **Optimize for availability**—AWS Auto Scaling scales the resource out and in automatically to maintain resource utilization at 40 percent. This option is useful when your application has urgent and sometimes unpredictable scaling needs.

- **Balance availability and cost**—AWS Auto Scaling scales the resource out and in automatically to maintain resource utilization at 50 percent. This option helps you maintain high availability while also reducing costs.

- **Optimize for cost**—AWS Auto Scaling scales the resource out and in automatically to maintain resource utilization at 70 percent. This option is useful for lowering costs if your application can handle having reduced buffer capacity when there are unexpected changes in demand.

For example, the scaling plan configures your Auto Scaling group to add or remove Amazon EC2 instances based on how much of the CPU is used on average for all instances in the group. You choose whether to optimize utilization for availability, cost, or a combination of the two by changing the scaling strategy.

Alternatively, you can configure a custom strategy if an existing strategy doesn't meet your needs. With a custom strategy, you can change the target utilization value, choose a different metric, or both.

> ⚠️ **Important**
>
> For the introductory tutorial, complete only the first step of the following procedure and then choose **Next** to continue.

**To specify a scaling strategy**

1. On the **Specify scaling strategy** page, for **Scaling plan details**, **Name**, enter a name for your scaling plan. The name of your scaling plan must be unique within your set of scaling plans for the Region. It can have a maximum of 128 characters, and it must not contain pipes "|", forward slashes "/", or colons ":".

2. All included resources are listed by resource type. For **Auto Scaling groups**, do the following:



a. Skip this step to use the default scaling strategy and metrics. To use a different scaling strategy or metrics instead, proceed with the following steps:

   i. For the **Scaling strategy**, choose the desired scaling strategy.

   For the introductory tutorial, make sure to choose **Optimize for availability**. This specifies that the average CPU utilization of your Auto Scaling group will be maintained at 40 percent.

    ii.   If you chose **Custom**, expand the **Configuration details** to choose the desired metrics and target value.

- For **Scaling metric**, choose the desired scaling metric.

- For **Target value**, choose the desired target value, such as the target utilization or the target throughput during any one-minute interval.

- For **Load metric** [Auto Scaling groups only], choose the desired load metric to use for predictive scaling.

- Select **Replace external scaling policies** to specify that AWS Auto Scaling can delete scaling policies previously created from outside of the scaling plan (such as from other consoles) and replace them with new target tracking scaling policies created by the scaling plan.

    b.   (Optional) By default, predictive scaling is enabled for Auto Scaling groups. To turn off predictive scaling for the Auto Scaling groups, clear **Enable predictive scaling**.

    c.   (Optional) By default, dynamic scaling is enabled for each resource type. To turn off dynamic scaling for the resource type, clear **Enable dynamic scaling**.

    d.   (Optional) By default, when you specify an application source from which multiple scalable resources are discovered, all resource types are automatically included in your scaling plan. To omit a type of resource from your scaling plan, clear **Include in scaling plan**.

3.   (Optional) To specify a scaling strategy for another resource type, repeat the preceding steps.

4.   When you are finished, choose **Next** to continue with the scaling plan creation process.

# Step 3: Configure advanced settings (optional)

Now that you have specified the scaling strategy to use for each resource type, you can choose to customize any of the default settings on a per resource basis using the **Configure advanced settings** step. For each resource type, there are multiple groups of settings that you can customize. In most cases, however, the default settings should be more efficient, with the possible exception of the values for minimum capacity and maximum capacity, which should be carefully adjusted.

Skip this procedure if you would like to keep the default settings. You can change these settings anytime by editing the scaling plan.

> ⚠️ **Important**
>
> For the introductory tutorial, let's make a few changes to update the maximum capacity of your Auto Scaling group and enable predictive scaling in forecast only mode. Although you do not need to customize all of the settings for the tutorial, let's also briefly examine the settings in each section.

# General settings

Use this procedure to view and customize the settings you specified in the previous step, on a per resource basis. You can also customize the minimum capacity and maximum capacity for each resource.

**To view and customize the general settings**

1.  On the **Configure advanced settings** page, choose the arrow to the left of any of the section headings to expand the section. For the tutorial, expand the **Auto Scaling groups** section.
2.  From the table that's displayed, choose the Auto Scaling group that you are using in this tutorial.
3.  Leave the **Include in scaling plan** option selected. If this option is not selected, the resource is omitted from the scaling plan. If you do not include at least one resource, the scaling plan cannot be created.
4.  To expand the view and see the details of the **General Settings** section, choose the arrow to the left of the section heading.
5.  You can make choices for any of the following items. For this tutorial, locate the **Maximum capacity** setting and enter a value of 3 in place of the current value.

    -   **Scaling strategy**—Allows you to optimize for availability, cost, or a balance of both, or to specify a custom strategy.
    -   **Enable dynamic scaling**—If this setting is cleared, the selected resource cannot scale using a target tracking scaling configuration.
    -   **Enable predictive scaling**—[Auto Scaling groups only] If this setting is cleared, the selected group cannot scale using predictive scaling.

- **Scaling metric**—Specifies the scaling metric to use. If you choose **Custom**, you can specify a custom metric to use instead of the predefined metrics that are available in the console. For more information, see the next topic in this section.

- **Target value**—Specifies the target utilization value to use.

- **Load metric**—[Auto Scaling groups only] Specifies the load metric to use. If you choose **Custom**, you can specify a custom metric to use instead of the predefined metrics that are available in the console. For more information, see the next topic in this section.

- **Minimum capacity**—Specifies the minimum capacity for the resource. AWS Auto Scaling ensures that your resource never goes below this size.

- **Maximum capacity**—Specifies the maximum capacity for the resource. AWS Auto Scaling ensures that your resource never goes above this size.

> ⓘ **Note**
>
> When you use predictive scaling, you can optionally choose a different maximum capacity behavior to use based on the forecast capacity. This setting is in the **Predictive scaling settings** section.

## Custom metrics

AWS Auto Scaling provides the most commonly used metrics for automatic scaling. However, depending on your needs, you might prefer to get data from different metrics instead of the metrics in the console. Amazon CloudWatch has many different metrics to choose from. CloudWatch also lets you publish your own metrics.

You use JSON to specify a CloudWatch custom metric. Before you follow these instructions, we recommend that you become familiar with the [Amazon CloudWatch User Guide](#).

To specify a custom metric, you construct a JSON-formatted payload using a set of required parameters from a template. You add the values for each parameter from CloudWatch. We provide the template as part of the custom options for **Scaling metric** and **Load metric** in the advanced settings of your scaling plan.

JSON represents data in two ways:

- An *object*, which is an unordered collection of name-value pairs. An object is defined within left ({) and right (}) braces. Each name-value pair begins with the name, followed by a colon, followed by the value. Name-value pairs are comma-separated.

- An *array*, which is an ordered collection of values. An array is defined within left ([) and right (]) brackets. Items in the array are comma-separated.

Here is an example of the JSON template with sample values for each parameter:

```json
{
  "MetricName": "MyBackendCPU",
  "Namespace": "MyNamespace",
  "Dimensions": [
    {
      "Name": "MyOptionalMetricDimensionName",
      "Value": "MyOptionalMetricDimensionValue"
    }
  ],
  "Statistic": "Sum"
}
```

For more information, see Customized scaling metric specification and Customized load metric specification in the *AWS Auto Scaling API Reference*.

# Dynamic scaling settings

Use this procedure to view and customize the settings for the target tracking scaling policy that AWS Auto Scaling creates.

**To view and customize the settings for dynamic scaling**

1. To expand the view and see the details of the **Dynamic scaling settings** section, choose the arrow to the left of the section heading.

2. You can make choices for the following items. However, the default settings are fine for this tutorial.

   - **Replace external scaling policies**—If this setting is cleared, it keeps existing scaling policies created from outside of this scaling plan, and does not create new ones.

   - **Disable scale-in**—If this setting is cleared, automatic scale-in to decrease the current capacity of the resource is allowed when the specified metric is below the target value.

- **Cooldown**—Creates scale-out and scale-in cooldown periods. The cooldown period is the amount of time the scaling policy waits for a previous scaling activity to take effect. For more information, see [Cooldown period](#) in the *Application Auto Scaling User Guide*. (This setting is not shown if the resource is an Auto Scaling group.)

- **Instance warmup**—[Auto Scaling groups only] Controls the amount of time that elapses before a newly launched instance begins contributing to the CloudWatch metrics. For more information, see [Instance warmup](#) in the *Amazon EC2 Auto Scaling User Guide*.

# Predictive scaling settings

If your resource is an Auto Scaling group, use this procedure to view and customize the settings AWS Auto Scaling uses for predictive scaling.

**To view and customize the settings for predictive scaling**

1. To expand the view and see the details of the **Predictive scaling settings** section, choose the arrow to the left of the section heading.

2. You can make choices for the following items. For this tutorial, change the **Predictive scaling mode** to **Forecast only**.

   - **Predictive scaling mode**—Specifies the scaling mode. The default is **Forecast and scale**. If you change it to **Forecast only**, the scaling plan forecasts future capacity but doesn't apply the scaling actions.

   - **Pre-launch instances**—Adjusts the scaling actions to run earlier when scaling out. For example, the forecast says to add capacity at 10:00 AM, and the buffer time is 5 minutes (300 seconds). The runtime of the corresponding scaling action is then 9:55 AM. This is helpful for Auto Scaling groups, where it can take a few minutes from the time an instance launches until it comes in service. The actual time can vary as it depends on several factors, such as the size of the instance and whether there are startup scripts to complete. The default is 300 seconds.

   - **Max capacity behavior**—Controls whether the selected resource can scale up above the maximum capacity when the forecast capacity is close to or exceeds the currently specified maximum capacity. The default is **Enforce the maximum capacity setting**.

     - **Enforce the maximum capacity setting**—AWS Auto Scaling cannot scale resource capacity higher than the maximum capacity. The maximum capacity is enforced as a hard limit.

- **Set the maximum capacity to equal forecast capacity**—AWS Auto Scaling can scale resource capacity higher than the maximum capacity to equal but not exceed forecast capacity.

- **Increase maximum capacity above forecast capacity**—AWS Auto Scaling can scale resource capacity higher than the maximum capacity by a specified buffer value. The intention is to give the target tracking scaling policy extra capacity if unexpected traffic occurs.

- **Max capacity behavior buffer**—If you chose **Increase maximum capacity above forecast capacity**, choose the size of the capacity buffer to use when the forecast capacity is close to or exceeds the maximum capacity. The value is specified as a percentage relative to the forecast capacity. For example, with a 10 percent buffer, if the forecast capacity is 50, and the maximum capacity is 40, then the effective maximum capacity is 55.

3. When you are finished customizing settings, choose **Next**.

> ⓘ **Note**
>
> To revert any of your changes, select the resources and choose **Revert to original**. This resets the selected resources to their last known state within the scaling plan.

# Step 4: Create your scaling plan

On the **Review and create** page, review the details of your scaling plan and choose **Create scaling plan**. You are directed to a page that shows the status of your scaling plan. The scaling plan can take a moment to finish being created while your resources are updated.

With predictive scaling, AWS Auto Scaling analyzes the history of the specified load metric from the past 14 days (minimum of 24 hours of data is required) to generate a forecast for two days ahead. It then schedules scaling actions to adjust the resource capacity to match the forecast for each hour in the forecast period.

After the creation of the scaling plan is complete, view the scaling plan details by choosing its name from the **Scaling plans** screen.

## (Optional) View scaling information for a resource

Use this procedure to view the scaling information created for a resource.

Data is presented in the following ways:

- Graphs showing recent metric history data from CloudWatch.

- Predictive scaling graphs showing load forecasts and capacity forecasts based on data from AWS Auto Scaling.

- A table that lists all the predictive scaling actions scheduled for the resource.

**To view scaling information for a resource**

1. Open the AWS Auto Scaling console at https://console.aws.amazon.com/awsautoscaling/.

2. On the **Scaling plans** page, choose the scaling plan.

3. On the **Scaling plan details** page, choose the resource to view.

## Monitoring and evaluating forecasts

When your scaling plan is up and running, you can monitor the load forecast, the capacity forecast, and scaling actions to examine the performance of predictive scaling. All of this data is available in the AWS Auto Scaling console for all Auto Scaling groups that are enabled for predictive scaling. Keep in mind that your scaling plan requires at least 24 hours of historical load data to make the initial forecast.

In the following example, the left side of each graph shows a historical pattern. The right side shows the forecast that was generated by the scaling plan for the forecast period. Both actual and forecast values (in blue and orange) are plotted.
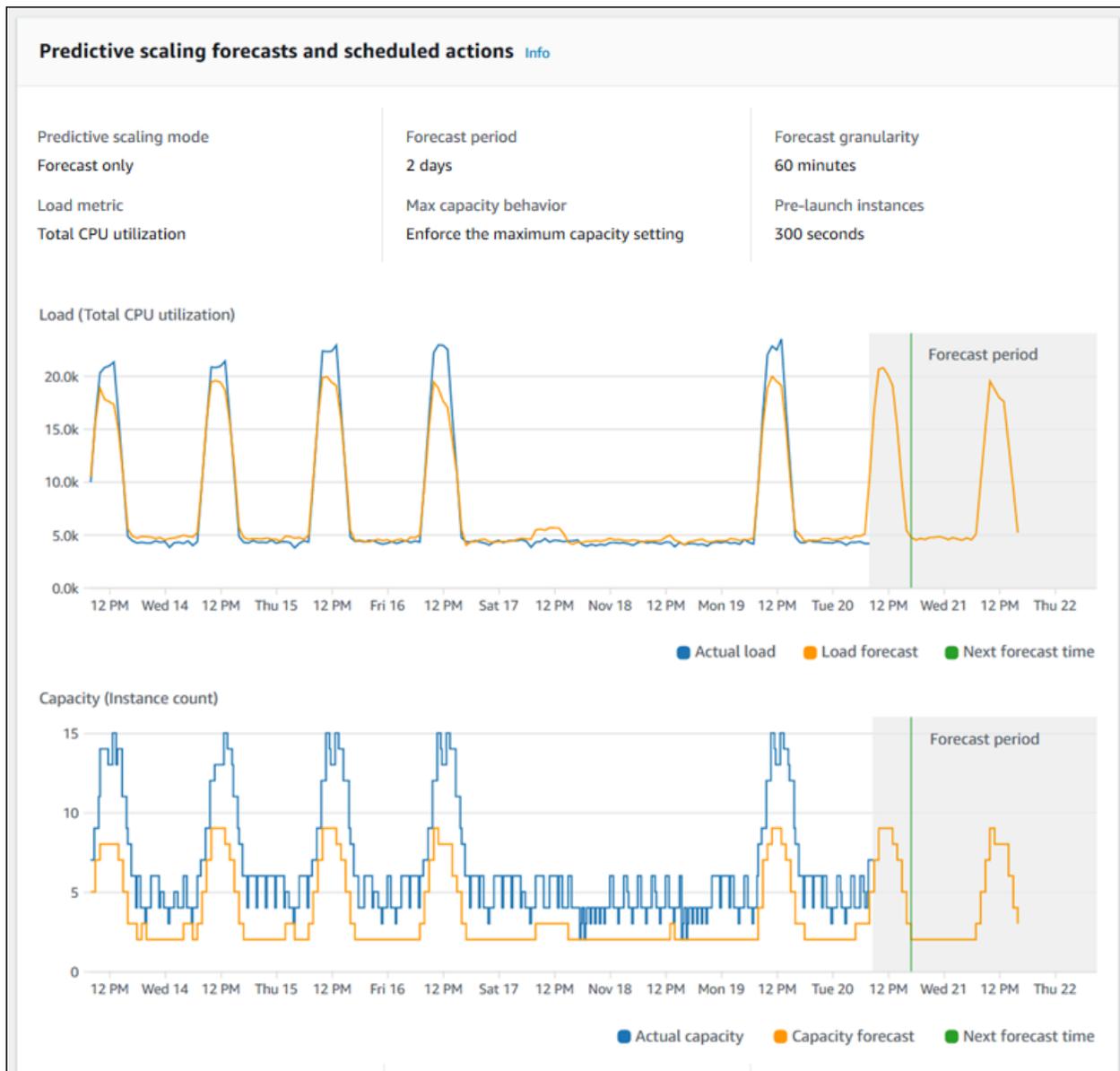
AWS Auto Scaling learns from your data automatically. First, it makes a load forecast. Then, a capacity forecast calculation determines the minimum number of instances that are required to support the application. Based on the capacity forecast, AWS Auto Scaling schedules scaling actions that scale the Auto Scaling group in advance of predicted load changes. If dynamic scaling is enabled (recommended), the Auto Scaling group can scale out additional capacity (or remove capacity) based on the current utilization of the group of instances.

When evaluating how well predictive scaling performs, monitor how closely the actual and forecast values match *over time*. When you create a scaling plan, AWS Auto Scaling provides graphs based on the most recent actual data. It also provides an initial forecast for the next 48 hours. However, when the scaling plan is created, there is very little forecast data to compare the actual data to.

Wait until the scaling plan has obtained forecast values for a few periods before comparing the historical forecast values against the actual values. After a few days of daily forecasts, you'll have a larger sample of forecast values to compare with actual values.

For patterns that occur on a daily basis, the time interval between creating your scaling plan and evaluating the forecast effectiveness can be as short as a few days. However, this length of time is insufficient to evaluate the forecast based on a recent pattern change. For example, let's say you are looking at the forecast for an Auto Scaling group that started a new marketing campaign in the past week. The campaign significantly increases your web traffic for the same two days each week. In situations like this, we recommend waiting for the group to collect a full week or two of new data before evaluating the effectiveness of the forecast. The same recommendation applies for a brand new Auto Scaling group that has only started to collect metric data.

If the actual and forecast values don't match after monitoring them over an appropriate length of time, you should also consider your choice of load metric. To be effective, the load metric must represent a reliable and accurate measure of the total load on all instances in the Auto Scaling group. The load metric is core to predictive scaling. If you choose a non-optimal load metric, it can prevent predictive scaling from making accurate load and capacity forecasts and scheduling the correct capacity adjustments for your Auto Scaling group.

# Step 5: Clean up

After you have completed the getting started tutorial, you can choose to keep your scaling plan. However, if you are not actively using your scaling plan, you should consider deleting it so that your account does not incur unnecessary charges.

Deleting a scaling plan deletes the target tracking scaling policies, their associated CloudWatch alarms, and the predictive scaling actions that AWS Auto Scaling created on your behalf.

Deleting a scaling plan does not delete your AWS CloudFormation stack, Auto Scaling group, or other scalable resources.

**To delete a scaling plan**

1.  Open the AWS Auto Scaling console at https://console.aws.amazon.com/awsautoscaling/.
2.  On the **Scaling plans** page, select the scaling plan that you created for this tutorial and choose **Delete**.
3.  When prompted for confirmation, choose **Delete**.

After you delete your scaling plan, your resources do not revert to their original capacity. For example, if your Auto Scaling group is scaled to 10 instances when you delete the scaling plan, your group is still scaled to 10 instances after the scaling plan is deleted. You can update the capacity of specific resources by accessing the console for each individual service.

## Delete your Auto Scaling group

To prevent your account from accruing Amazon EC2 charges, you should also delete the Auto Scaling group that you created for this tutorial.

For step-by-step instructions, see Delete your Auto Scaling group in the *Amazon EC2 Auto Scaling User Guide*.

# Step 6: Next steps

Now that you have familiarized yourself with scaling plans and some of its features, you may want to try creating your own scaling plan template using AWS CloudFormation.

An AWS CloudFormation template is a JSON or YAML-formatted text file that describes the Amazon Web Services infrastructure needed to run an application or service along with any interconnections among infrastructure components. With AWS CloudFormation, you deploy and manage an associated collection of resources as a *stack*. AWS CloudFormation is available at no additional charge, and you pay only for the AWS resources needed to run your applications. Resources can consist of any AWS resource you define within the template. For more information, see AWS CloudFormation concepts in the *AWS CloudFormation User Guide*.

In the *AWS CloudFormation User Guide*, we provide a simple template to get you started. The sample template is available as an example in the AWS::AutoScalingPlans::ScalingPlan section of the AWS CloudFormation template reference documentation. The sample template creates a scaling plan for a single Auto Scaling group and enables predictive scaling and dynamic scaling.

For more information, see Getting started with AWS CloudFormation in the *AWS CloudFormation User Guide*.

# Migrate your scaling plan

You can migrate from a scaling plan to Amazon EC2 Auto Scaling and Application Auto Scaling scaling policies.

**Migration process**

- [Step 1: Review your existing setup](#)
- [Step 2: Create predictive scaling policies](#)
- [Step 3: Review the forecasts that the predictive scaling policies generate](#)
- [Step 4: Prepare to delete the scaling plan](#)
- [Step 5: Delete the scaling plan](#)
- [Step 6: Reactivate dynamic scaling](#)
- [Step 7: Reactivate predictive scaling](#)
- [Amazon EC2 Auto Scaling reference for migrating target tracking scaling policies](#)
- [Application Auto Scaling reference for migrating target tracking scaling policies](#)
- [Additional information](#)

> ⚠️ **Important**
>
> To migrate a scaling plan, you must complete multiple steps in exact order. While you migrate your scaling plan, *don't update it*, as that breaks the order of operations and could cause undesirable behavior.

# Step 1: Review your existing setup

To determine which scaling settings you must move over, use the [describe-scaling-plans](#) command.

```
aws autoscaling-plans describe-scaling-plans \
  --scaling-plan-names my-scaling-plan
```

Make a note of items that you want to preserve from the existing scaling plan, which can include the following:

- **MinCapacity** – The minimum capacity of the scalable resource.

- **MaxCapacity** – The maximum capacity of the scalable resource.

- **PredefinedLoadMetricType** – A load metric for predictive scaling.

- **PredefinedScalingMetricType** – A scaling metric for target tracking (dynamic) scaling and predictive scaling.

- **TargetValue** – The target value for the scaling metric.

## Differences between scaling plans and scaling policies

There are a few important differences between scaling plans and scaling policies:

- A scaling policy can enable only one type of scaling: either target tracking scaling or predictive scaling. To use both scaling methods, you must create separate policies.

- Likewise, you must define the scaling metric for predictive scaling and the scaling metric for target tracking scaling separately within their respective policies.

# Step 2: Create predictive scaling policies

If you don't use predictive scaling, then skip ahead to [Step 4: Prepare to delete the scaling plan](#).

To provide time to evaluate the forecast, we recommend that you create predictive scaling policies before other scaling policies.

For any Auto Scaling groups with an existing load metric specification, do the following to turn it into an Amazon EC2 Auto Scaling-based predictive scaling policy.

**To create predictive scaling policies**

1. In a JSON file, define a `MetricSpecifications` structure as shown in the following example:

   ```
   {
     "MetricSpecifications":[
       {
         ...
       }
     ]
   }
   ```

2.  In the `MetricSpecifications` structure, for each load metric in your scaling plan, create
    a `PredefinedLoadMetricSpecification` or `CustomizedLoadMetricSpecification`
    using the equivalent settings from the scaling plan.

    The following are examples of the structure of the load metric section.

    With predefined metrics

```
{
    "MetricSpecifications":[
      {
        "PredefinedLoadMetricSpecification":{
          "PredefinedMetricType":"ASGTotalCPUUtilization"
        },
        ...
      }
    ]
}
```

    For more information, see PredictiveScalingPredefinedLoadMetric in the *Amazon EC2 Auto
    Scaling API Reference.*

    With custom metrics

```
{
    "MetricSpecifications":[
      {
        "CustomizedLoadMetricSpecification":{
          "MetricDataQueries":[
            {
              "Id":"load_metric",
              "MetricStat":{
                "Metric":{
                  "MetricName":"MyLoadMetric",
                  "Namespace":"MyNameSpace",
                  "Dimensions":[
                    {
                      "Name":"MyOptionalMetricDimensionName",
                      "Value":"MyOptionalMetricDimensionValue"
                    }
                  ]
                },
                "Stat":"Sum"
```

```
                    }
                }
            ]
        },
        ...
    }
]
}
```

For more information, see [PredictiveScalingCustomizedLoadMetric](#) in the *Amazon EC2 Auto Scaling API Reference*.

3. Add the scaling metric specification to the `MetricSpecifications` and define a target value.

   The following are examples of the structure of the scaling metric and target value sections.

   With predefined metrics

```
{
    "MetricSpecifications":[
        {
            "PredefinedLoadMetricSpecification":{
                "PredefinedMetricType":"ASGTotalCPUUtilization"
            },
            "PredefinedScalingMetricSpecification":{
                "PredefinedMetricType":"ASGCPUUtilization"
            },
            "TargetValue":50
        }
    ],
    ...
}
```

For more information, see [PredictiveScalingPredefinedScalingMetric](#) in the *Amazon EC2 Auto Scaling API Reference*.

With custom metrics

```
{
    "MetricSpecifications":[
        {
            "CustomizedLoadMetricSpecification":{
```

```
      "MetricDataQueries":[
        {
          "Id":"load_metric",
          "MetricStat":{
            "Metric":{
              "MetricName":"MyLoadMetric",
              "Namespace":"MyNameSpace",
              "Dimensions":[
                {
                  "Name":"MyOptionalMetricDimensionName",
                  "Value":"MyOptionalMetricDimensionValue"
                }
              ]
            },
            "Stat":"Sum"
          }
        }
      ]
    },
    "CustomizedScalingMetricSpecification":{
      "MetricDataQueries":[
        {
          "Id":"scaling_metric",
          "MetricStat":{
            "Metric":{
              "MetricName":"MyUtilizationMetric",
              "Namespace":"MyNameSpace",
              "Dimensions":[
                {
                  "Name":"MyOptionalMetricDimensionName",
                  "Value":"MyOptionalMetricDimensionValue"
                }
              ]
            },
            "Stat":"Average"
          }
        }
      ]
    },
    "TargetValue":50
  }
],
...
```

```
}
```

For more information, see PredictiveScalingCustomizedScalingMetric in the *Amazon EC2 Auto Scaling API Reference*.

4.  To forecast only, add the property Mode with a value of ForecastOnly. After you finish migrating predictive scaling and making sure that the forecast is accurate and reliable, you can change the mode to allow scaling. For more information, see Step 7: Reactivate predictive scaling.

```
{
   "MetricSpecifications":[
   ...
   ],
   "Mode":"ForecastOnly",
   ...
}
```

For more information, see PredictiveScalingConfiguration in the *Amazon EC2 Auto Scaling API Reference*.

5.  If the **ScheduledActionBufferTime** property is present in your scaling plan, then copy its value to the SchedulingBufferTime property in your predictive scaling policy.

```
{
   "MetricSpecifications":[
   ...
   ],
   "Mode":"ForecastOnly",
   "SchedulingBufferTime":300,
   ...
}
```

For more information, see PredictiveScalingConfiguration in the *Amazon EC2 Auto Scaling API Reference*.

6.  If the **PredictiveScalingMaxCapacityBehavior** and **PredictiveScalingMaxCapacityBuffer** properties are present in your scaling plan, then you can configure the MaxCapacityBreachBehavior and MaxCapacityBuffer properties in your predictive scaling policy. These properties define what should happen if the forecast capacity approaches or exceeds the maximum capacity specified for the Auto Scaling group.

> ⚠️ **Warning**
>
> If you set the `MaxCapacityBreachBehavior` property to `IncreaseMaxCapacity`, then more instances could launch than intended unless you monitor and manage the increased maximum capacity. The increased maximum capacity becomes the new normal maximum capacity for the Auto Scaling group until you manually update it. The maximum capacity doesn't automatically decrease back to the original maximum.

```
{
  "MetricSpecifications":[
  ...
  ],
  "Mode":"ForecastOnly",
  "SchedulingBufferTime":300,
  "MaxCapacityBreachBehavior": "IncreaseMaxCapacity",
  "MaxCapacityBuffer": 10
}
```

For more information, see [PredictiveScalingConfiguration](#) in the *Amazon EC2 Auto Scaling API Reference*.

7. Save the JSON file with a unique name. Make a note of the file name. You need it in the next step and again at the end of the migration procedure when you reactivate your predictive scaling policies. For more information, see [Step 7: Reactivate predictive scaling](#).

8. After you save your JSON file, run the [put-scaling-policy](#) command. In the following example, replace each *user input placeholder* with your own information.

```
aws autoscaling put-scaling-policy --policy-name my-predictive-scaling-policy \
   --auto-scaling-group-name my-asg --policy-type PredictiveScaling \
   --predictive-scaling-configuration file://my-predictive-scaling-config.json
```

If successful, this command returns the policy's Amazon Resource Name (ARN).

```
{
  "PolicyARN": "arn:aws:autoscaling:region:account-id:scalingPolicy:2f4f5048-
d8a8-4d14-b13a-d1905620f345:autoScalingGroupName/my-asg:policyName/my-predictive-
scaling-policy",
```

```
        "Alarms": []
    }
```

9.   Repeat these steps for each load metric specification that you're migrating to an Amazon EC2 Auto Scaling-based predictive scaling policy.

# Step 3: Review the forecasts that the predictive scaling policies generate

If you don't use predictive scaling, then skip the following procedure.

A forecast is available shortly after you create a predictive scaling policy. After Amazon EC2 Auto Scaling generates the forecast, you can review the forecast for the policy through the Amazon EC2 Auto Scaling console and adjust as necessary.

**To review the forecast for a predictive scaling policy**

1.   Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.

2.   In the navigation pane, choose **Auto Scaling Groups**, and then choose the name of your Auto Scaling group from the list.

3.   On the **Automatic scaling** tab, in **Predictive scaling policies**, choose your policy.

4.   In the **Monitoring** section, you can view your policy's past and future forecasts for load and capacity against actual values.

     For more information, see Review predictive scaling monitoring graphs in the *Amazon EC2 Auto Scaling User Guide*.

5.   Repeat these steps for each predictive scaling policy that you created.

# Step 4: Prepare to delete the scaling plan

For any resources with an existing target tracking scaling configuration, do the following to collect any additional information that you need from the scaling plan before deleting it.

To describe the scaling policy information from the scaling plan, use the describe-scaling-plan-resources command. In the following example command, replace *my-scaling-plan* with your own information.

```
aws autoscaling-plans describe-scaling-plan-resources \
   --scaling-plan-name my-scaling-plan \
   --scaling-plan-version 1
```

Review the output and confirm you want to migrate the scaling policies described. Use this information to create new Amazon EC2 Auto Scaling and Application Auto Scaling-based target tracking scaling polices in Step 6: Reactivate dynamic scaling.

# Step 5: Delete the scaling plan

Before creating new target tracking scaling policies, you must delete the scaling plan to delete the scaling policies that it created.

To delete your scaling plan, use the delete-scaling-plan command. In the following example command, replace *my-scaling-plan* with your own information.

```
aws autoscaling-plans delete-scaling-plan \
   --scaling-plan-name my-scaling-plan \
   --scaling-plan-version 1
```

After you delete the scaling plan, dynamic scaling is deactivated. So if there are sudden surges in traffic or workload, the capacity available for each scalable resource won't increase on its own. As a precaution, you might want to manually increase the capacity of your scalable resources in the short term.

**To increase the capacity of an Auto Scaling group**

1.  Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2.  In the navigation pane, choose **Auto Scaling Groups**, and then choose the name of your Auto Scaling group from the list.
3.  On the **Details** tab, choose **Group details**, **Edit**.
4.  For **Desired capacity**, increase the desired capacity.
5.  When you're finished, choose **Update**.

**To add an Aurora replica to a DB cluster**

1.  Open the Amazon RDS console at https://console.aws.amazon.com/rds/.

2.  In the navigation pane, choose **Databases**, and then select your DB cluster.

3.  Make sure that both the cluster and the primary instance are in the **Available** state.

4.  Choose **Actions**, **Add reader**.

5.  On the **Add reader** page, specify options for your new Aurora replica.

6.  Choose **Add reader**.


**To increase the provisioned read and write capacity of a DynamoDB table or global secondary index**

1.  Open the DynamoDB console at https://console.aws.amazon.com/dynamodb/.

2.  In the navigation pane, choose **Tables**, and then choose the name of your table from the list.

3.  On the **Additional settings** tab, choose **Read/write capacity**, **Edit**.

4.  On the **Edit read/write capacity** page, for **Read capacity**, **Provisioned capacity units**, increase the provisioned read capacity of the table.

5.  (Optional) If you want your global secondary indexes to use the same read capacity settings as the base table, then select the **Use the same read capacity settings for all global secondary indexes** check box.

6.  For **Write capacity**, **Provisioned capacity units**, increase the provisioned write capacity of the table.

7.  (Optional) If you want your global secondary indexes to use the same write capacity settings as the base table, then select the **Use the same write capacity settings for all global secondary indexes** check box.

8.  If you *didn't* select the check boxes in steps 5 or 7, then scroll down the page to update the read and write capacity of any global secondary indexes.

9.  Choose **Save changes** to continue.


**To increase the running task count for your Amazon ECS service**

1.  Open the console at https://console.aws.amazon.com/ecs/v2.

2.  In the navigation pane, choose **Clusters**, and then choose the name of your cluster from the list.

3.  In the **Services** section, select the check box next to the service, and then choose **Update**.

4.  For **Desired tasks**, enter the number of tasks that you want to run for the service.

5.   Choose **Update**.

**To increase the capacity of a Spot Fleet**

1.   Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.

2.   In the navigation pane, choose **Spot Requests**, and then select your Spot Fleet request.

3.   Choose **Actions**, **Modify target capacity**.

4.   In **Modify target capacity**, enter the new target capacity and On-Demand Instance portion.

5.   Choose **Submit**.

# Step 6: Reactivate dynamic scaling

Reactivate dynamic scaling by creating target tracking scaling policies.

When you create a target tracking scaling policy for an Auto Scaling group, you add it directly to the group. When you create a target tracking scaling policy for other scalable resources, you first register the resource as a scalable target and then you add a target tracking scaling policy to the scalable target.

**Topics**

- Create target tracking scaling policies for Auto Scaling groups
- Create target tracking scaling policies for other scalable resources

# Create target tracking scaling policies for Auto Scaling groups

**To create target tracking scaling policies for Auto Scaling groups**

1.   In a JSON file, create a `PredefinedMetricSpecification` or `CustomizedMetricSpecification` using the equivalent settings from the scaling plan.

The following are examples of a target tracking configuration. In these examples, replace each *user input placeholder* with your own information.

With predefined metrics

```
{
```

```
    "TargetValue": 50.0,
    "PredefinedMetricSpecification":
      {
        "PredefinedMetricType": "ASGAverageCPUUtilization"
      }
 }
```

For more information, see PredefinedMetricSpecification in the *Amazon EC2 Auto Scaling API Reference.*

With custom metrics

```
{
  "TargetValue": 100.0,
  "CustomizedMetricSpecification": {
    "MetricName": "MyBacklogPerInstance",
    "Namespace": "MyNamespace",
    "Dimensions": [{
      "Name": "MyOptionalMetricDimensionName",
      "Value": "MyOptionalMetricDimensionValue"
    }],
    "Statistic": "Average",
    "Unit": "None"
  }
}
```

For more information, see CustomizedMetricSpecification in the *Amazon EC2 Auto Scaling API Reference.*

2. To create your scaling policy, use the put-scaling-policy command, along with the JSON file that you created in the previous step. In the following example, replace each *user input placeholder* with your own information.

```
aws autoscaling put-scaling-policy --policy-name my-target-tracking-scaling-policy \
  --auto-scaling-group-name my-asg --policy-type TargetTrackingScaling \
  --target-tracking-configuration file://config.json
```

3. Repeat this process for each scaling plan-based scaling policy that you're migrating to an Amazon EC2 Auto Scaling-based target tracking scaling policy.

# Create target tracking scaling policies for other scalable resources

Next, create target tracking scaling policies for other scalable resources by performing the following configuration tasks.

- Register a scalable target for auto scaling with the Application Auto Scaling service.

- Add a target tracking scaling policy on the scalable target.

**To create target tracking scaling policies for other scalable resources**

1.  Use the register-scalable-target command to register the resource as a scalable target and define the scaling limits for the scaling policy.

    In the following example, replace each *user input placeholder* with your own information. For the command options, provide the following information:

    - `--service-namespace` – A namespace for the target service (for example, **ecs**). To obtain service namespaces, see the RegisterScalableTarget reference.

    - `--scalable-dimension` – A scalable dimension associated with the target resource (for example, `ecs:service:DesiredCount`). To obtain scalable dimensions, see the RegisterScalableTarget reference.

    - `--resource-id` – A resource ID for the target resource (for example, **service/*my-cluster*/*my-service***). For information about the syntax and examples of specific resource IDs, see the RegisterScalableTarget reference.

    ```
    aws application-autoscaling register-scalable-target --service-namespace namespace \
      --scalable-dimension dimension \
      --resource-id identifier \
      --min-capacity 1 --max-capacity 10
    ```

    If successful, this command returns the ARN of the scalable target.

    ```
    {
        "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
    }
    ```

2.  In a JSON file, create a `PredefinedMetricSpecification` or
    `CustomizedMetricSpecification` using the equivalent settings from the scaling plan.

    The following are examples of a target tracking configuration.

    With predefined metrics

    ```
    {
      "TargetValue": 70.0,
      "PredefinedMetricSpecification":
        {
          "PredefinedMetricType": "ECSServiceAverageCPUUtilization"
        }
    }
    ```

    For more information, see PredefinedMetricSpecification in the *Application Auto Scaling API
    Reference*.

    With custom metrics

    ```
    {
      "TargetValue": 70.0,
      "CustomizedMetricSpecification": {
        "MetricName": "MyUtilizationMetric",
        "Namespace": "MyNamespace",
        "Dimensions": [{
          "Name": "MyOptionalMetricDimensionName",
          "Value": "MyOptionalMetricDimensionValue"
        }],
        "Statistic": "Average",
        "Unit": "Percent"
      }
    }
    ```

    For more information, see CustomizedMetricSpecification in the *Application Auto Scaling
    API Reference*.

3.  To create your scaling policy, use the put-scaling-policy command, along with the JSON file
    that you created in the previous step.

    ```
    aws application-autoscaling put-scaling-policy --service-namespace namespace \
      --scalable-dimension dimension \
    ```

```
    --resource-id identifier \
    --policy-name my-target-tracking-scaling-policy --policy-
 type TargetTrackingScaling \
    --target-tracking-scaling-policy-configuration file://config.json
```

4.   Repeat this process for each scaling plan-based scaling policy that you're migrating to an
     Application Auto Scaling-based target tracking scaling policy.

# Step 7: Reactivate predictive scaling

If you don't use predictive scaling, then skip this step.

Reactivate predictive scaling by switching predictive scaling to forecast and scale.

To make this change, update the JSON files that you created in Step 2: Create predictive scaling
policies and change the value of the Mode option to ForecastAndScale as in the following
example:

```
"Mode":"ForecastAndScale"
```

Then, update each predictive scaling policy with the put-scaling-policy command. In this example,
replace each user input placeholder with your own information.

```
aws autoscaling put-scaling-policy --policy-name my-predictive-scaling-policy \
   --auto-scaling-group-name my-asg --policy-type PredictiveScaling \
   --predictive-scaling-configuration file://my-predictive-scaling-config.json
```

Alternatively, you can make this change from the Amazon EC2 Auto Scaling console by turning on
the **Scale based on forecast** setting. For more information, see Predictive scaling for Amazon EC2
Auto Scaling in the *Amazon EC2 Auto Scaling User Guide*.

# Amazon EC2 Auto Scaling reference for migrating target tracking scaling policies

For reference purposes, the following table lists all the target tracking configuration properties
in the scaling plan with their corresponding property in the Amazon EC2 Auto Scaling
PutScalingPolicy API operation.

| Scaling plan source property | Amazon EC2 Auto Scaling target property |
|---|---|
| PolicyName | PolicyName |
| PolicyType | PolicyType |
| TargetTrackingConfiguration.CustomizedScalingMetricSpecification.Dimensions.Name | TargetTrackingConfiguration.CustomizedMetricSpecification.Dimensions.Name |
| TargetTrackingConfiguration.CustomizedScalingMetricSpecification.Dimensions.Value | TargetTrackingConfiguration.CustomizedMetricSpecification.Dimensions.Value |
| TargetTrackingConfiguration.CustomizedScalingMetricSpecification.MetricName | TargetTrackingConfiguration.CustomizedMetricSpecification.MetricName |
| TargetTrackingConfiguration.CustomizedScalingMetricSpecification.Namespace | TargetTrackingConfiguration.CustomizedMetricSpecification.Namespace |
| TargetTrackingConfiguration.CustomizedScalingMetricSpecification.Statistic | TargetTrackingConfiguration.CustomizedMetricSpecification.Statistic |
| TargetTrackingConfiguration.CustomizedScalingMetricSpecification.Unit | TargetTrackingConfiguration.CustomizedMetricSpecification.Unit |
| TargetTrackingConfiguration.DisableScaleIn | TargetTrackingConfiguration.DisableScaleIn |
| TargetTrackingConfiguration.EstimatedInstanceWarmup | TargetTrackingConfiguration.EstimatedInstanceWarmup [1] |
| TargetTrackingConfiguration.PredefinedScalingMetricSpe | TargetTrackingConfiguration.PredefinedMetricSpecification.PredefinedMetricType |

| Scaling plan source property | Amazon EC2 Auto Scaling target property |
|---|---|
| `cification.PredefinedScalingMetricType` | |
| `TargetTrackingConfiguration.PredefinedScalingMetricSpecification.ResourceLabel` | `TargetTrackingConfiguration.PredefinedMetricSpecification.ResourceLabel` |
| `TargetTrackingConfiguration.ScaleInCooldown` | `Not available` |
| `TargetTrackingConfiguration.ScaleOutCooldown` | `Not available` |
| `TargetTrackingConfiguration.TargetValue` | `TargetTrackingConfiguration.TargetValue` |

[1] Instance warmup is a feature for Auto Scaling groups that helps to ensure that newly launched instances are ready to receive traffic before contributing their usage data to the scaling metric. While instances are still warming up, Amazon EC2 Auto Scaling slows down the process of adding or removing instances to the group. Instead of specifying a warmup time for a scaling policy, we recommend that you use the default instance warmup setting of your Auto Scaling group to ensure that all instance launches use the same instance warmup time. For more information, see Set the default instance warmup for an Auto Scaling group in the *Amazon EC2 Auto Scaling User Guide*.

# Application Auto Scaling reference for migrating target tracking scaling policies

For reference purposes, the following table lists all the target tracking configuration properties in the scaling plan with their corresponding property in the Application Auto Scaling `PutScalingPolicy` API operation.

| Scaling plan source property | Application Auto Scaling target property |
|---|---|
| `PolicyName` | `PolicyName` |

| Scaling plan source property | Application Auto Scaling target property |
|---|---|
| PolicyType | PolicyType |
| TargetTrackingConfiguration .CustomizedScalingMetricSpe cification.Dimensions.Name | TargetTrackingScalingPolicy Configuration.CustomizedMet ricSpecification.Dimensions .Name |
| TargetTrackingConfiguration .CustomizedScalingMetricSpe cification.Dimensions.Value | TargetTrackingScalingPolicy Configuration.CustomizedMet ricSpecification.Dimensions .Value |
| TargetTrackingConfiguration .CustomizedScalingMetricSpe cification.MetricName | TargetTrackingScalingPolicy Configuration.CustomizedMet ricSpecification.MetricName |
| TargetTrackingConfiguration .CustomizedScalingMetricSpe cification.Namespace | TargetTrackingScalingPolicy Configuration.CustomizedMet ricSpecification.Namespace |
| TargetTrackingConfiguration .CustomizedScalingMetricSpe cification.Statistic | TargetTrackingScalingPolicy Configuration.CustomizedMet ricSpecification.Statistic |
| TargetTrackingConfiguration .CustomizedScalingMetricSpe cification.Unit | TargetTrackingScalingPolicy Configuration.CustomizedMet ricSpecification.Unit |
| TargetTrackingConfiguration .DisableScaleIn | TargetTrackingScalingPolicy Configuration.DisableScaleIn |
| TargetTrackingConfiguration .EstimatedInstanceWarmup | Not available |

| Scaling plan source property | Application Auto Scaling target property |
|---|---|
| `TargetTrackingConfiguration .PredefinedScalingMetricSpe cification.PredefinedScalin gMetricType` | `TargetTrackingScalingPolicy Configuration.PredefinedMet ricSpecification.Predefined MetricType` |
| `TargetTrackingConfiguration .PredefinedScalingMetricSpe cification.ResourceLabel` | `TargetTrackingScalingPolicy Configuration.PredefinedMet ricSpecification.ResourceLabel` |
| `TargetTrackingConfiguration .ScaleInCooldown` [1] | `TargetTrackingScalingPolicy Configuration.ScaleInCooldown` |
| `TargetTrackingConfiguration .ScaleOutCooldown` [1] | `TargetTrackingScalingPolicy Configuration.ScaleOutCooldown` |
| `TargetTrackingConfiguration .TargetValue` | `TargetTrackingScalingPolicy Configuration.TargetValue` |

[1] Application Auto Scaling uses cooldown periods to slow down scaling when your scalable resource is scaling out (increasing capacity) and scaling in (reducing capacity). For more information, see Define cooldown periods in the *Application Auto Scaling User Guide*.

# Additional information

To learn how to create new predictive scaling policies from the console, see the following topic:

- **Amazon EC2 Auto Scaling** – Create a predictive scaling policy in the *Amazon EC2 Auto Scaling User Guide*.

To learn how to create new target tracking scaling policies using the console, see the following topics:

- **Amazon Aurora** – Using Amazon Aurora Auto Scaling with Aurora Replicas in the *Amazon RDS User Guide*.

- **DynamoDB** – [Using the AWS Management Console with DynamoDB auto scaling](#) in the *Amazon DynamoDB Developer Guide*.

- **Amazon EC2 Auto Scaling** – [Create a target tracking scaling policy](#) in the *Amazon EC2 Auto Scaling User Guide*.

- **Amazon ECS** – [Updating a service using the console](#) in the *Amazon Elastic Container Service Developer Guide*.

- **Spot Fleet** – [Scale Spot Fleet using a target tracking policy](#) in the *Amazon EC2 User Guide*.

# Scaling plan security

Cloud security at AWS is the highest priority. As an AWS customer, you benefit from a data center and network architecture that is built to meet the requirements of the most security-sensitive organizations.

Security is a shared responsibility between AWS and you. The shared responsibility model describes this as security *of* the cloud and security *in* the cloud:

- **Security of the cloud** – AWS is responsible for protecting the infrastructure that runs AWS services in the AWS Cloud. AWS also provides you with services that you can use securely. Third-party auditors regularly test and verify the effectiveness of our security as part of the AWS compliance programs. To learn about the compliance programs that apply to AWS Auto Scaling, see AWS services in scope by compliance program.

- **Security in the cloud** – Your responsibility is determined by the AWS service that you use. You are also responsible for other factors including the sensitivity of your data, your company's requirements, and applicable laws and regulations.

This documentation helps you understand how to apply the shared responsibility model when using scaling plans, and also helps you understand how to manage access to scaling plans.

**Topics**

- Access scaling plans using interface VPC endpoints
- Data protection for scaling plans
- Identity and access management for scaling plans
- Compliance validation for scaling plans
- Infrastructure security for scaling plans

# Access scaling plans using interface VPC endpoints

You can use AWS PrivateLink to create a private connection between your VPC and AWS Auto Scaling. You can access AWS Auto Scaling as if it were in your VPC, without the use of an internet gateway, NAT device, VPN connection, or AWS Direct Connect connection. Instances in your VPC don't need public IP addresses to access AWS Auto Scaling.

You establish this private connection by creating an *interface endpoint*, powered by AWS PrivateLink. We create an endpoint network interface in each subnet that you enable for the interface endpoint. These are requester-managed network interfaces that serve as the entry point for traffic destined for AWS Auto Scaling.

For more information, see [Access AWS services through AWS PrivateLink](#) in the *AWS PrivateLink Guide*.

**Topics**

- [Create an interface VPC endpoint for scaling plans](#)
- [Create a VPC endpoint policy for scaling plans](#)
- [Endpoint migration](#)

## Create an interface VPC endpoint for scaling plans

Create an endpoint for AWS Auto Scaling scaling plans using the following service name:

```
com.amazonaws.region.autoscaling-plans
```

For more information, see [Access an AWS service using an interface VPC endpoint](#) in the *AWS PrivateLink Guide*.

You do not need to change any other settings. AWS Auto Scaling API calls other AWS services using either service endpoints or private interface VPC endpoints, whichever are in use.

## Create a VPC endpoint policy for scaling plans

You can attach a policy to your VPC endpoint to control access to the AWS Auto Scaling API. The policy specifies:

- The principal that can perform actions.
- The actions that can be performed.
- The resource on which the actions can be performed.

The following example shows a VPC endpoint policy that denies everyone permission to delete a scaling plan through the endpoint. The example policy also grants everyone permission to perform all other actions.

```
{
    "Statement": [
        {
            "Action": "*",
            "Effect": "Allow",
            "Resource": "*",
            "Principal": "*"
        },
        {
            "Action": "autoscaling-plans:DeleteScalingPlan",
            "Effect": "Deny",
            "Resource": "*",
            "Principal": "*"
        }
    ]
}
```

For more information, see VPC endpoint policies in the *AWS PrivateLink Guide.*

# Endpoint migration

On November 22, 2019, we introduced `autoscaling-plans.`*`region`*`.amazonaws.com` as the new default DNS hostname and endpoint for calls to the AWS Auto Scaling API. The new endpoint is compatible with the latest release of the AWS CLI and SDKs. If you have not done so already, install the latest AWS CLI and SDKs to use the new endpoint. To update the AWS CLI, see Installing the AWS CLI using pip in the *AWS Command Line Interface User Guide.* For information about the AWS SDKs, see Tools for Amazon Web Services.

> ⚠️ **Important**
>
> For backward compatibility, the existing `autoscaling.`*`region`*`.amazonaws.com` endpoint will continue to be supported for calls to the AWS Auto Scaling API. To set up the `autoscaling.`*`region`*`.amazonaws.com` endpoint as a private interface VPC endpoint, see Amazon EC2 Auto Scaling and interface VPC endpoints in the *Amazon EC2 Auto Scaling User Guide.*

**Endpoint to Call When Using the CLI or the AWS Auto Scaling API**

For the current release of AWS Auto Scaling, your calls to the AWS Auto Scaling API automatically go to the `autoscaling-plans.`*`region`*`.amazonaws.com` endpoint instead of `autoscaling.`*`region`*`.amazonaws.com`.

You can call the new endpoint in the CLI by using the following parameter with each command to specify the endpoint: `--endpoint-url https://autoscaling-plans.`*`region`*`.amazonaws.com`.

Although it is not recommended, you can also call the old endpoint in the CLI by using the following parameter with each command to specify the endpoint: `--endpoint-url https://autoscaling.`*`region`*`.amazonaws.com`.

For the various SDKs used to call the APIs, see the documentation for the SDK of interest to learn how to direct the requests to a specific endpoint. For more information, see Tools for Amazon Web Services.

# Data protection for scaling plans

The AWS shared responsibility model applies to data protection in AWS Auto Scaling. As described in this model, AWS is responsible for protecting the global infrastructure that runs all of the AWS Cloud. You are responsible for maintaining control over your content that is hosted on this infrastructure. You are also responsible for the security configuration and management tasks for the AWS services that you use. For more information about data privacy, see the Data Privacy FAQ. For information about data protection in Europe, see the AWS Shared Responsibility Model and GDPR blog post on the *AWS Security Blog*.

For data protection purposes, we recommend that you protect AWS account credentials and set up individual users with AWS IAM Identity Center or AWS Identity and Access Management (IAM). That way, each user is given only the permissions necessary to fulfill their job duties. We also recommend that you secure your data in the following ways:

- Use multi-factor authentication (MFA) with each account.
- Use SSL/TLS to communicate with AWS resources. We require TLS 1.2 and recommend TLS 1.3.
- Set up API and user activity logging with AWS CloudTrail.
- Use AWS encryption solutions, along with all default security controls within AWS services.
- Use advanced managed security services such as Amazon Macie, which assists in discovering and securing sensitive data that is stored in Amazon S3.

- If you require FIPS 140-3 validated cryptographic modules when accessing AWS through a command line interface or an API, use a FIPS endpoint. For more information about the available FIPS endpoints, see Federal Information Processing Standard (FIPS) 140-3.

We strongly recommend that you never put confidential or sensitive information, such as your customers' email addresses, into tags or free-form text fields such as a **Name** field. This includes when you work with AWS Auto Scaling or other AWS services using the console, API, AWS CLI, or AWS SDKs. Any data that you enter into tags or free-form text fields used for names may be used for billing or diagnostic logs. If you provide a URL to an external server, we strongly recommend that you do not include credentials information in the URL to validate your request to that server.

# Identity and access management for scaling plans

AWS Identity and Access Management (IAM) is an AWS service that helps an administrator securely control access to AWS resources. IAM administrators control who can be *authenticated* (signed in) and *authorized* (have permissions) to use AWS Auto Scaling resources. IAM is an AWS service that you can use with no additional charge.

For complete IAM documentation, see the IAM User Guide.

## Access control

You can have valid credentials to authenticate your requests, but unless you have permissions you cannot create or access scaling plans. For example, you must have permissions to create scaling plans, configure predictive scaling, and so on.

The following sections provide details on how an IAM administrator can use IAM to help secure your scaling plans, by controlling who can work with scaling plans.

**Topics**

- How scaling plans work with IAM
- Predictive scaling service-linked role
- Identity-based policy examples for scaling plans

# How scaling plans work with IAM

Before you use IAM to manage who can create, access, and manage AWS Auto Scaling scaling plans, you should understand what IAM features are available to use with scaling plans.

**Topics**

- [Identity-based policies](#)
- [Resource-based policies](#)
- [Access Control Lists (ACLs)](#)
- [Authorization based on tags](#)
- [IAM roles](#)

## Identity-based policies

With IAM identity-based policies, you can specify allowed or denied actions and resources, and the conditions under which actions are allowed or denied. Scaling plans support specific actions, resources, and condition keys. To learn about all of the elements that you use in a JSON policy, see [IAM JSON policy elements reference](#) in the *IAM User Guide*.

**Actions**

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The `Action` element of a JSON policy describes the actions that you can use to allow or deny access in a policy. Policy actions usually have the same name as the associated AWS API operation. There are some exceptions, such as *permission-only actions* that don't have a matching API operation. There are also some operations that require multiple actions in a policy. These additional actions are called *dependent actions*.

Include actions in a policy to grant permissions to perform the associated operation.

Scaling plan actions in IAM policy statements use the following prefix before the action: `autoscaling-plans:`. Policy statements must include either an `Action` or `NotAction` element. Scaling plans have their own sets of actions that describe tasks that you can perform with this service.

To specify multiple actions in a single statement, separate them with commas as shown in the following example.

```
"Action": [
      "autoscaling-plans:DescribeScalingPlans",
      "autoscaling-plans:DescribeScalingPlanResources"
```

You can specify multiple actions using wildcards (*). For example, to specify all actions that begin with the word `Describe`, include the following action.

```
"Action": "autoscaling-plans:Describe*"
```

To see a complete list of scaling plan actions that can be used in policy statements, see Actions, resources, and condition keys for AWS Auto Scaling in the *Service Authorization Reference*.

**Resources**

The `Resource` element specifies the object or objects to which the action applies.

Scaling plans have no service-defined resources that can be used as the `Resource` element of an IAM policy statement. Therefore, there are no Amazon Resource Names (ARNs) for you to use in an IAM policy. To control access to scaling plan actions, always use an * (asterisk) as the resource when writing an IAM policy.

**Condition keys**

The `Condition` element (or `Condition` *block*) lets you specify conditions in which a statement is in effect. For example, you might want a policy to be applied only after a specific date. To express conditions, use predefined condition keys.

Scaling plans do not provide any service-specific condition keys, but they do support using some global condition keys. To see all AWS global condition keys, see AWS global condition context keys in the *IAM User Guide*.

The `Condition` element is optional.

**Examples**

To view examples of identity-based policies for scaling plans, see Identity-based policy examples for scaling plans.

## Resource-based policies

Other Amazon Web Services, such as Amazon Simple Storage Service, support resource-based permissions policies. For example, you can attach a permissions policy to an S3 bucket to manage access permissions to that bucket.

Scaling plans do not support resource-based policies.

## Access Control Lists (ACLs)

Scaling plans do not support Access Control Lists (ACLs).

## Authorization based on tags

Scaling plans cannot be tagged. They also have no service-defined resources that can be tagged. Therefore, they do not support controlling access based on tags on a resource.

Scaling plans may contain taggable resources, such as Auto Scaling groups, that support controlling access based on tags. For more information, see the documentation for that AWS service.

## IAM roles

An IAM role is an entity within your AWS account that has specific permissions.

### Using temporary credentials

You can use temporary credentials to sign in with federation, to assume an IAM role, or to assume a cross-account role. You obtain temporary security credentials by calling AWS STS API operations such as AssumeRole or GetFederationToken.

Scaling plans support using temporary credentials.

### Service-linked roles for scaling plans

AWS Auto Scaling uses service-linked roles for the permissions that it requires to call other AWS services on your behalf. Service-linked roles make setting up scaling plans easier because you don't have to manually add the necessary permissions. For more information, see Using service-linked roles in the *IAM User Guide*.

AWS Auto Scaling uses a few types of service-linked roles to call other AWS services on your behalf when you work with a scaling plan:

- *Predictive scaling service-linked role* — Allows AWS Auto Scaling to access historical metric data from CloudWatch. Also allows creation of scheduled actions for Auto Scaling groups based on a load forecast and capacity prediction. For more information, see Predictive scaling service-linked role.

- *Amazon EC2 Auto Scaling service-linked role* — Allows AWS Auto Scaling to access and manage target tracking scaling policies for Auto Scaling groups. For more information, see Service-linked roles for Amazon EC2 Auto Scaling in the *Amazon EC2 Auto Scaling User Guide*.

- *Application Auto Scaling service-linked role* — Allows AWS Auto Scaling to access and manage target tracking scaling policies for other scalable resources. There is one service-linked role for each service. For more information, see Service-linked roles for Application Auto Scaling in the *Application Auto Scaling User Guide*.

You can use the following procedure to determine if your account already has a service-linked role.

**To determine whether a service-linked role already exists**

1. Open the IAM console at https://console.aws.amazon.com/iam/.

2. In the navigation pane, choose **Roles**.

3. Search the list for `AWSServiceRole` to find the service-linked roles that exist in your account. Look for the name of the service-linked role that you want to check.

**Service roles**

AWS Auto Scaling has no service roles for scaling plans.

# Predictive scaling service-linked role

AWS Auto Scaling uses service-linked roles for the permissions that it requires to call other AWS on your behalf when you work with a scaling plan. For more information, see Service-linked roles for scaling plans.

The following sections describe how to create and manage the service-linked role for predictive scaling. Start by configuring permissions to allow an IAM entity (such as a user, group, or role) to create, edit, or delete a service-linked role.

# Permissions granted by the service-linked role

AWS Auto Scaling uses the service-linked role named
**AWSServiceRoleForAutoScalingPlans_EC2AutoScaling** to call the following actions on your
behalf when you enable predictive scaling:

- `cloudwatch:GetMetricData`
- `autoscaling:DescribeAutoScalingGroups`
- `autoscaling:DescribeScheduledActions`
- `autoscaling:BatchPutScheduledUpdateGroupAction`
- `autoscaling:BatchDeleteScheduledAction`

**AWSServiceRoleForAutoScalingPlans_EC2AutoScaling** trusts the `autoscaling-plans.amazonaws.com` service to assume the role.

## Create the service-linked role (automatic)

You don't need to manually create the **AWSServiceRoleForAutoScalingPlans_EC2AutoScaling**
role. AWS creates this role for you when you create a scaling plan in your account and enable
predictive scaling.

For AWS to create a service-linked role on your behalf, you must have the required permissions. For
more information, see Service-linked role permissions in the *IAM User Guide*.

## Create the service-linked role (manual)

To create the service-linked role manually, you can use the IAM console, IAM CLI, or IAM API. For
more information, see Creating a service-linked role in the *IAM User Guide*.

**To create a service-linked role (AWS CLI)**

Use the following create-service-linked-role CLI command to create the service-linked role.

```
aws iam create-service-linked-role --aws-service-name autoscaling-plans.amazonaws.com
```

## Edit the service-linked role

You can edit the description of **AWSServiceRoleForAutoScalingPlans_EC2AutoScaling** using IAM.
For more information, see Editing a service-linked role in the *IAM User Guide*.

## Delete the service-linked role

If you no longer need to use scaling plans, we recommend that you delete
**AWSServiceRoleForAutoScalingPlans_EC2AutoScaling**.

You can delete a service-linked role only after you delete all scaling plans in your AWS account that
have predictive scaling enabled. This ensures that you can't inadvertently remove permissions to
access your scaling plans.

You can use the IAM console, IAM CLI, or IAM API to delete the service-linked role. For more
information, see Deleting a service-linked role in the *IAM User Guide*.

After you delete the **AWSServiceRoleForAutoScalingPlans_EC2AutoScaling** service-linked role,
AWS Auto Scaling creates the role again if you create a scaling plan with predictive scaling enabled.

### Supported Regions

AWS Auto Scaling supports using service-linked roles in all of the AWS Regions where scaling plans
available. For information about the Regional availability of scaling plans, see AWS Auto Scaling
endpoints and quotas in the *AWS General Reference*.

# Identity-based policy examples for scaling plans

By default, a brand new IAM user has no permissions to do anything. An IAM administrator must
create and assign IAM policies that give an IAM identity (such as a user or role) permission to work
with scaling plans.

To learn how to create an IAM policy using these example JSON policy documents, see Creating
policies on the JSON tab in the *IAM User Guide*.

**Topics**

- Policy best practices
- Allow users to create scaling plans
- Allow users to enable predictive scaling
- Additional required permissions
- Permissions required to create a service-linked role

# Policy best practices

Identity-based policies determine whether someone can create, access, or delete AWS Auto Scaling resources in your account. These actions can incur costs for your AWS account. When you create or edit identity-based policies, follow these guidelines and recommendations:

- **Get started with AWS managed policies and move toward least-privilege permissions** – To get started granting permissions to your users and workloads, use the *AWS managed policies* that grant permissions for many common use cases. They are available in your AWS account. We recommend that you reduce permissions further by defining AWS customer managed policies that are specific to your use cases. For more information, see AWS managed policies or AWS managed policies for job functions in the *IAM User Guide*.

- **Apply least-privilege permissions** – When you set permissions with IAM policies, grant only the permissions required to perform a task. You do this by defining the actions that can be taken on specific resources under specific conditions, also known as *least-privilege permissions*. For more information about using IAM to apply permissions, see Policies and permissions in IAM in the *IAM User Guide*.

- **Use conditions in IAM policies to further restrict access** – You can add a condition to your policies to limit access to actions and resources. For example, you can write a policy condition to specify that all requests must be sent using SSL. You can also use conditions to grant access to service actions if they are used through a specific AWS service, such as AWS CloudFormation. For more information, see IAM JSON policy elements: Condition in the *IAM User Guide*.

- **Use IAM Access Analyzer to validate your IAM policies to ensure secure and functional permissions** – IAM Access Analyzer validates new and existing policies so that the policies adhere to the IAM policy language (JSON) and IAM best practices. IAM Access Analyzer provides more than 100 policy checks and actionable recommendations to help you author secure and functional policies. For more information, see IAM Access Analyzer policy validation in the *IAM User Guide*.

- **Require multi-factor authentication (MFA)** – If you have a scenario that requires IAM users or a root user in your AWS account, turn on MFA for additional security. To require MFA when API operations are called, add MFA conditions to your policies. For more information, see Configuring MFA-protected API access in the *IAM User Guide*.

For more information about best practices in IAM, see Security best practices in IAM in the *IAM User Guide*.

## Allow users to create scaling plans

The following shows an example of an identity-based policy that grants permissions to create scaling plans.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "autoscaling-plans:*",
                "cloudwatch:PutMetricAlarm",
                "cloudwatch:DeleteAlarms",
                "cloudwatch:DescribeAlarms",
                "cloudformation:ListStackResources"
            ],
            "Resource": "*"
        }
    ]
}
```

To work with a scaling plan, end users must have additional permissions that allow them to work with specific resources in their account. These permissions are listed in Additional required permissions.

Each console user also needs permissions that allow them to discover the scalable resources in their account and to view graphs of CloudWatch metric data from the AWS Auto Scaling console. The additional set of permissions required to work with the AWS Auto Scaling console is listed below:

- `cloudformation:ListStacks`: To list stacks.
- `tag:GetTagKeys`: To find scalable resources that contain certain tag keys.
- `tag:GetTagValues`: To find resources that contain certain tag values.
- `autoscaling:DescribeTags`: To find Auto Scaling groups that contain certain tags.
- `cloudwatch:GetMetricData`: To view data in metric graphs.

## Allow users to enable predictive scaling

The following shows an example of an identity-based policy that grants permissions to enable predictive scaling. These permissions extend the features of scaling plans that are set up to scale Auto Scaling groups.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "cloudwatch:GetMetricData",
              "autoscaling:DescribeAutoScalingGroups",
              "autoscaling:DescribeScheduledActions",
              "autoscaling:BatchPutScheduledUpdateGroupAction",
              "autoscaling:BatchDeleteScheduledAction"
            ],
            "Resource": "*"
        }
    ]
}
```

## Additional required permissions

To successfully configure scaling plans, end users must be granted permissions for each target service for which they will configure scaling. To grant the minimum permissions required to work with target services, read the information in this section and specify the relevant actions in the `Action` element of an IAM policy statement.

### Auto Scaling groups

To add Auto Scaling groups to a scaling plan, users must have the following permissions from Amazon EC2 Auto Scaling:

- `autoscaling:UpdateAutoScalingGroup`
- `autoscaling:DescribeAutoScalingGroups`
- `autoscaling:PutScalingPolicy`
- `autoscaling:DescribePolicies`
- `autoscaling:DeletePolicy`

**ECS services**

To add ECS services to a scaling plan, users must have the following permissions from Amazon ECS and Application Auto Scaling:

- `ecs:DescribeServices`
- `ecs:UpdateService`
- `application-autoscaling:RegisterScalableTarget`
- `application-autoscaling:DescribeScalableTargets`
- `application-autoscaling:DeregisterScalableTarget`
- `application-autoscaling:PutScalingPolicy`
- `application-autoscaling:DescribeScalingPolicies`
- `application-autoscaling:DeleteScalingPolicy`

**Spot Fleet**

To add Spot Fleets to a scaling plan, users must have the following permissions from Amazon EC2 and Application Auto Scaling:

- `ec2:DescribeSpotFleetRequests`
- `ec2:ModifySpotFleetRequest`
- `application-autoscaling:RegisterScalableTarget`
- `application-autoscaling:DescribeScalableTargets`
- `application-autoscaling:DeregisterScalableTarget`
- `application-autoscaling:PutScalingPolicy`
- `application-autoscaling:DescribeScalingPolicies`
- `application-autoscaling:DeleteScalingPolicy`

**DynamoDB tables or global indexes**

To add DynamoDB tables or global indexes to a scaling plan, users must have the following permissions from DynamoDB and Application Auto Scaling:

- `dynamodb:DescribeTable`
- `dynamodb:UpdateTable`

- `application-autoscaling:RegisterScalableTarget`

- `application-autoscaling:DescribeScalableTargets`

- `application-autoscaling:DeregisterScalableTarget`

- `application-autoscaling:PutScalingPolicy`

- `application-autoscaling:DescribeScalingPolicies`

- `application-autoscaling:DeleteScalingPolicy`

**Aurora DB clusters**

To add Aurora DB clusters to a scaling plan, users must have the following permissions from Amazon Aurora and Application Auto Scaling:

- `rds:AddTagsToResource`

- `rds:CreateDBInstance`

- `rds:DeleteDBInstance`

- `rds:DescribeDBClusters`

- `rds:DescribeDBInstances`

- `application-autoscaling:RegisterScalableTarget`

- `application-autoscaling:DescribeScalableTargets`

- `application-autoscaling:DeregisterScalableTarget`

- `application-autoscaling:PutScalingPolicy`

- `application-autoscaling:DescribeScalingPolicies`

- `application-autoscaling:DeleteScalingPolicy`

## Permissions required to create a service-linked role

AWS Auto Scaling requires permissions to create a service-linked role the first time any user in your AWS account creates a scaling plan with predictive scaling enabled. If the service-linked role does not exist already, AWS Auto Scaling creates it in your account. The service-linked role grants permissions to AWS Auto Scaling so that it can call other services on your behalf.

For automatic role creation to succeed, users must have permissions for the `iam:CreateServiceLinkedRole` action.

```
"Action": "iam:CreateServiceLinkedRole"
```

The following shows an example of an identity-based policy that grants permissions to create a service-linked role.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::*:role/aws-service-role/autoscaling-
plans.amazonaws.com/AWSServiceRoleForAutoScalingPlans_EC2AutoScaling",
            "Condition": {
                "StringLike": {
                    "iam:AWSServiceName":"autoscaling-plans.amazonaws.com"
                }
            }
        }
    ]
}
```

For more information, see Predictive scaling service-linked role.

# Compliance validation for scaling plans

To learn whether an AWS service is within the scope of specific compliance programs, see AWS services in Scope by Compliance Program and choose the compliance program that you are interested in. For general information, see AWS Compliance Programs.

You can download third-party audit reports using AWS Artifact. For more information, see Downloading Reports in AWS Artifact.

Your compliance responsibility when using AWS services is determined by the sensitivity of your data, your company's compliance objectives, and applicable laws and regulations. AWS provides the following resources to help with compliance:

- Security and Compliance Quick Start Guides – These deployment guides discuss architectural considerations and provide steps for deploying baseline environments on AWS that are security and compliance focused.

- **Architecting for HIPAA Security and Compliance on Amazon Web Services** – This whitepaper describes how companies can use AWS to create HIPAA-eligible applications.

  > **ⓘ Note**
  >
  > Not all AWS services are HIPAA eligible. For more information, see the HIPAA Eligible Services Reference.

- **AWS Compliance Resources** – This collection of workbooks and guides might apply to your industry and location.

- **AWS Customer Compliance Guides** – Understand the shared responsibility model through the lens of compliance. The guides summarize the best practices for securing AWS services and map the guidance to security controls across multiple frameworks (including National Institute of Standards and Technology (NIST), Payment Card Industry Security Standards Council (PCI), and International Organization for Standardization (ISO)).

- **Evaluating Resources with Rules** in the *AWS Config Developer Guide* – The AWS Config service assesses how well your resource configurations comply with internal practices, industry guidelines, and regulations.

- **AWS Security Hub** – This AWS service provides a comprehensive view of your security state within AWS. Security Hub uses security controls to evaluate your AWS resources and to check your compliance against security industry standards and best practices. For a list of supported services and controls, see Security Hub controls reference.

- **Amazon GuardDuty** – This AWS service detects potential threats to your AWS accounts, workloads, containers, and data by monitoring your environment for suspicious and malicious activities. GuardDuty can help you address various compliance requirements, like PCI DSS, by meeting intrusion detection requirements mandated by certain compliance frameworks.

- **AWS Audit Manager** – This AWS service helps you continuously audit your AWS usage to simplify how you manage risk and compliance with regulations and industry standards.

# Infrastructure security for scaling plans

As a managed service, AWS Auto Scaling is protected by AWS global network security. For information about AWS security services and how AWS protects infrastructure, see AWS Cloud Security. To design your AWS environment using the best practices for infrastructure security, see Infrastructure Protection in *Security Pillar AWS Well-Architected Framework*.

You use AWS published API calls to access AWS Auto Scaling through the network. Clients must support the following:

- Transport Layer Security (TLS). We require TLS 1.2 and recommend TLS 1.3.
- Cipher suites with perfect forward secrecy (PFS) such as DHE (Ephemeral Diffie-Hellman) or ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Most modern systems such as Java 7 and later support these modes.

Additionally, requests must be signed by using an access key ID and a secret access key that is associated with an IAM principal. Or you can use the AWS Security Token Service (AWS STS) to generate temporary security credentials to sign requests.

# Quotas for your scaling plans

Your AWS account has the default quotas (previously referred to as limits) related to scaling plans. Unless otherwise noted, each quota is Region-specific. You can request increases for some quotas, and other quotas cannot be increased.

To view the quotas for Application Auto Scaling, open the Service Quotas console. In the navigation pane, choose **AWS services** and select **AWS Auto Scaling Plans**.

To request a quota increase, see Requesting a quota increase in the *Service Quotas User Guide*.

Your AWS account has the following quotas related to scaling plans.

| Name | Default | Adjustable |
|------|---------|------------|
| Scalable resources per resource type | Amazon DynamoDB: 3,000 \| Amazon EC2 Auto Scaling groups: 200 \| All other resource types: 500 | Yes |
| Scaling plans | 100 | Yes |
| Scaling instructions per scaling plan | 500 | No |
| Target tracking configurations per scaling instruction | 10 | No |

Keep service quotas in mind as you scale out your workloads. For example, when you reach the maximum number of capacity units allowed by a service, scaling out will stop. If demand drops and the current capacity decreases, AWS Auto Scaling can scale out again. To avoid reaching this service quota limit again, you can request an increase. Each service has its own default quotas for the maximum capacity of the resource. For information about the default quotas for other Amazon Web Services, see Service endpoints and quotas in the *Amazon Web Services General Reference*.

# Document history for scaling plans

The following table describes important additions to the AWS Auto Scaling documentation. For notification about updates to this documentation, you can subscribe to the RSS feed.

| Change | Description | Date |
|--------|-------------|------|
| New content for migrating from AWS Auto Scaling to alternative options | You can now migrate from AWS Auto Scaling to Amazon EC2 Auto Scaling predictive scaling, which offers more functionality. For more information, see Migrate your scaling plan. | April 5, 2024 |
| New security content | We released an updated Security chapter. As part of this update, we replaced "Authentication and Access Control" with the Identity and access management for AWS Auto Scaling. | March 12, 2020 |
| Support for Amazon VPC endpoints | You can now establish a private connection between your VPC and AWS Auto Scaling. For migration considerations and instructions, see Scaling plans and interface VPC endpoints. | November 22, 2019 |
| Support for increasing maximum capacity above forecast capacity | Adds console support for allowing the scaling plan to increase maximum capacity above forecast capacity by a specified buffer value. | March 9, 2019 |

| | For more information, see [Predictive scaling settings](). | |
|---|---|---|
| [Predictive scaling and enhancements]() | You can now use predictive scaling to proactively scale your Amazon EC2 Auto Scaling groups. This release also adds support for replacing scaling policies created outside of the scaling plan (such as from other consoles) and controlling whether you enable your plan's dynamic scaling feature. | November 20, 2018 |
| [Support for custom resource settings]() | Added support for customizing various settings for each individual resource or multiple resources at the same time. | October 9, 2018 |
| [Tags as an application source]() | This release adds support for specifying a set of tags as an application source. | April 23, 2018 |
| [New service]() | Initial release of AWS Auto Scaling. | January 16, 2018 |