
AWS Resource Groups and Tags

User Guide



AWS Resource Groups and Tags: User Guide

Copyright © Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon's trademarks and trade dress may not be used in connection with any product or service that is not Amazon's, in any manner that is likely to cause confusion among customers, or in any manner that disparages or discredits Amazon. All other trademarks not owned by Amazon are the property of their respective owners, who may or may not be affiliated with, connected to, or sponsored by Amazon.

Table of Contents

Resource groups	1
What are resource groups?	1
Use cases for resource groups	2
AWS Resource Groups and permissions	2
AWS Resource Groups resources	3
How tagging works	3
Getting started	3
Prerequisites	4
Creating groups	8
Types of resource group queries	8
Build a tag-based query and create a group	10
Create an AWS CloudFormation stack-based group	12
Updating groups	13
Update tag-based query groups	14
Update an AWS CloudFormation stack-based group	15
Deleting groups	17
AWS services that work with AWS Resource Groups	18
Tagging resources	21
Tag Editor	21
Find resources to tag	22
Manage tags	26
Troubleshooting tag changes	32
Tags in IAM policies	33
Tag-related condition keys	33
Example IAM policies that use tags	34
AWS Organizations tag policies	35
Prerequisites and permissions	35
Evaluating compliance for an account	37
Evaluating organization-wide compliance	39
Supported resources	41
Amazon API Gateway	42
Amazon AppStream	42
AWS AppSync	42
Amazon Braket	43
AWS Certificate Manager	43
AWS Certificate Manager Private Certificate Authority	43
AWS Cloud9	43
AWS CloudFormation	44
Amazon CloudFront	44
AWS CloudTrail	44
Amazon CloudWatch	44
Amazon CloudWatch Logs	45
AWS CodeArtifact	45
AWS CodeBuild	45
AWS CodeCommit	45
AWS CodeDeploy	46
Amazon CodeGuru Reviewer	46
AWS CodePipeline	46
Amazon Cognito	46
Amazon Comprehend	47
AWS Config	47
AWS Data Exchange	47
AWS Data Pipeline	47
AWS Database Migration Service	48

Amazon DynamoDB	48
Amazon EMR	48
Amazon EMR Containers	48
Amazon ElastiCache	49
AWS Elastic Beanstalk	49
Amazon Elastic Compute Cloud (Amazon EC2)	49
Amazon Elastic Container Registry	50
Amazon Elastic Container Service	50
Amazon Elastic File System	51
Amazon Elastic Inference	51
Amazon Elastic Kubernetes Service (Amazon EKS)	51
Elastic Load Balancing	51
Amazon OpenSearch Service	52
Amazon CloudWatch Events	52
Amazon FSx	52
Amazon Forecast	52
Amazon Fraud Detector	53
AWS Glue	53
AWS Glue DataBrew	54
AWS Identity and Access Management	54
Amazon Inspector	55
AWS IoT	55
AWS IoT Analytics	55
AWS IoT Events	55
AWS IoT Greengrass	56
AWS Key Management Service	56
Amazon Kinesis	56
Amazon Kinesis Data Analytics	57
Amazon Kinesis Data Firehose	57
AWS Lambda	57
Amazon MQ	57
Amazon Macie	58
Amazon Managed Streaming for Apache Kafka	58
Amazon OpenSearch Service OpenSearch	58
AWS OpsWorks	58
AWS Organizations	59
Amazon Pinpoint	59
Amazon Quantum Ledger Database (Amazon QLDB)	59
Amazon Redshift	60
Amazon Relational Database Service (Amazon RDS)	60
AWS Resource Access Manager	61
AWS Resource Groups	61
AWS Robomaker	61
Amazon Route 53	61
Amazon Route 53 Resolver	62
Amazon S3 Glacier	62
Amazon SageMaker	62
AWS Secrets Manager	63
AWS Service Catalog	63
Service Quotas	63
Amazon Simple Email Service	64
Amazon Simple Notification Service	64
Amazon Simple Queue Service	64
Amazon Simple Storage Service (Amazon S3)	64
AWS Step Functions	65
Storage Gateway	65
AWS Systems Manager	65

Amazon WorkSpaces	65
Deprecated resource types	66
Security	67
Data Protection	67
Data Encryption	68
Internet Traffic Privacy	68
Identity and Access Management	68
Audience	69
Authenticating With Identities	69
Managing Access Using Policies	71
How Resource Groups Works with IAM	73
Identity-Based Policy Examples	76
Troubleshooting	78
Logging and Monitoring	80
CloudTrail Integration	80
Compliance Validation	82
Resilience	83
Infrastructure Security	83
Security Best Practices	83
Reference	85
Service quotas for Resource Groups	85
Service quotas for Tagging (Tag Editor and Resource Groups Tagging API)	85
AWS managed policies available for use with AWS Resource Groups and Tag Editor	86
Document history	88
Earlier updates	92
AWS glossary	93

What are resource groups?

You can use *resource groups* to organize your AWS resources. AWS Resource Groups is the service that lets you manage and automate tasks on large numbers of resources at one time. This guide shows you how to create and manage resource groups in AWS Resource Groups. The tasks that you can perform on a resource vary based on the AWS service you're using. For a list of the services that support AWS Resource Groups and a brief description of what each service allows you to do with a resource group, see [AWS services that work with AWS Resource Groups \(p. 18\)](#).

You can access Resource Groups through any of the following entry points.

- In the [AWS Management Console](#), in the top navigation bar, choose **Services**. Then, under **Management & Governance**, choose **Resource Groups & Tag Editor**.

Direct link: [AWS Resource Groups console](#)

- In the AWS Systems Manager console, from the left navigation pane entry for Resource Groups.
- By using the Resource Groups API, in AWS CLI commands or AWS SDK programming languages.

To work with resource groups on the AWS Management Console home

1. Sign in to the AWS Management Console.
2. On the navigation bar, choose **Services**.
3. Under **Management & Governance**, choose **Resource Groups & Tag Editor**.
4. In the navigation pane on the left, choose **Saved Resource Groups** to work with an existing group, or **Create a Group** to create a new one.

What are resource groups?

In AWS, a *resource* is an entity that you can work with. Examples include an Amazon EC2 instance, an AWS CloudFormation stack, or an Amazon S3 bucket. If you work with multiple resources, you might find it useful to manage them as a group rather than move from one AWS service to another for each task. If you manage large numbers of related resources, such as EC2 instances that make up an application layer, you likely need to perform bulk actions on these resources at one time. Examples of bulk actions include:

- Applying updates or security patches.
- Upgrading applications.
- Opening or closing ports to network traffic.
- Collecting specific log and monitoring data from your fleet of instances.

A *resource group* is a collection of AWS resources that are all in the same AWS Region, and that match the criteria specified in the group's query. In Resource Groups, there are two types of queries you can use to build a group. Both query types include resources that are specified in the format `AWS::service::resource`.

- **Tag-based**

Tag-based queries include lists of resources and tags. *Tags* are keys that help identify and sort your resources within your organization. Optionally, tags include values for keys.

Important

Do not store personally identifiable information (PII) or other confidential or sensitive information in tags. We use tags to provide you with billing and administration services. Tags are not intended to be used for private or sensitive data.

- **AWS CloudFormation stack-based**

In an AWS CloudFormation stack-based query, you choose an AWS CloudFormation stack in your account in the current region, and then choose resource types within the stack that you want to be in the group. You can base your query on only one AWS CloudFormation stack.

Resource groups can be *nested*; a resource group can contain existing resource groups in the same region.

Use cases for resource groups

By default, the AWS Management Console is organized by AWS service. But with Resource Groups, you can create a custom console that organizes and consolidates information based on criteria specified in tags, or the resources in an AWS CloudFormation stack. The following list describes some of the cases in which resource grouping can help organize your resources.

- An application that has different phases, such as development, staging, and production.
- Projects managed by multiple departments or individuals.
- A set of AWS resources that you use together for a common project or that you want to manage or monitor as a group.
- A set of resources related to applications that run on a specific platform, such as Android or iOS.

For example, you are developing a web application, and you are maintaining separate sets of resources for your alpha, beta, and release stages. Each version runs on Amazon EC2 with an Amazon Elastic Block Store storage volume. You use Elastic Load Balancing to manage traffic and Route 53 to manage your domain. Without Resource Groups, you might have to access multiple consoles just to check the status of your services or modify the settings for one version of your application.

With Resource Groups, you use a single page to view and manage your resources. For example, let's say you use the tool to create a resource group for each version—alpha, beta, and release—of your application. To check your resources for the alpha version of your application, open your resource group. Then view the consolidated information on your resource group page. To modify a specific resource, choose the resource's links on your resource group page to access the service console that has the settings that you need.

AWS Resource Groups and permissions

Resource Groups feature permissions are at the account level. As long as users who are sharing your account have the correct IAM permissions, they can work with resource groups that you create. For information about creating IAM users, see [Creating an IAM User](#) in the *IAM User Guide*.

Tags are properties of a resource, so they are shared across your entire account. Users in a department or specialized group can draw from a common vocabulary (tags) to create resource groups that are meaningful to their roles and responsibilities. Having a common pool of tags also means that when users share a resource group, they don't have to worry about missing or conflicting tag information.

AWS Resource Groups resources

In Resource Groups, the only available resource is a group. Groups have unique Amazon Resource Names (ARNs) associated with them. For more information about ARNs, see [Amazon Resource Names \(ARN\)](#) and [AWS Service Namespaces](#) in the *Amazon Web Services General Reference*.

Resource Type	ARN Format
Resource Group	<code>arn:aws:resource-groups:<i>region</i>:<i>account</i>:group/<i>group-name</i></code>

How tagging works

Tags are key and value pairs that act as metadata for organizing your AWS resources. With most AWS resources, you have the option of adding tags when you create the resource, whether it's an Amazon EC2 instance, an Amazon S3 bucket, or other resource. However, you can also add tags to multiple, supported resources at once by using Tag Editor. You build a query for resources of various types, and then add, remove, or replace tags for the resources in your search results. Tag-based queries assign an AND operator to tags, so any resource that matches the specified resource types and all specified tags is returned by the query.

Important

Do not store personally identifiable information (PII) or other confidential or sensitive information in tags. We use tags to provide you with billing and administration services. Tags are not intended to be used for private or sensitive data.

For more information about tagging, see [Tag Editor \(p. 21\)](#) in this guide. You can tag [supported resources \(p. 41\)](#) by using Tag Editor, and some additional resources by using tagging functionality in the service console in which you create and manage the resource.

Getting started with AWS Resource Groups

In AWS, a *resource* is an entity that you can work with. Examples include an Amazon EC2 instance, an Amazon S3 bucket, or an Amazon Route 53 hosted zone. If you work with multiple resources, you might find it useful to manage them as a group rather than move from one AWS service to another for each task.

This section shows you how to get started with AWS Resource Groups. First, organize AWS resources by tagging them in Tag Editor. Then build queries in Resource Groups that include the resource types you want in a group, and tags that you've applied to resources.

After you've created resource groups in Resource Groups, use AWS Systems Manager tools such as Automation to simplify management tasks on your groups of resources.

For more information about getting started with AWS Systems Manager features and tools, see the [AWS Systems Manager User Guide](#).

Topics

- [Prerequisites for working with AWS Resource Groups \(p. 4\)](#)

Prerequisites for working with AWS Resource Groups

Before you get started working with resource groups, be sure you have an active AWS account with existing resources and appropriate rights to tag resources and create groups.

Sign up for AWS

If you do not have an AWS account, complete the following steps to create one.

To sign up for an AWS account

1. Open <https://portal.aws.amazon.com/billing/signup>.
2. Follow the online instructions.

Part of the sign-up procedure involves receiving a phone call and entering a verification code on the phone keypad.

Create resources

You can create an empty resource group, but won't be able to perform any tasks on resource group members until there are resources in the group. For more information about the supported resource types, see [Resources you can use with AWS Resource Groups and Tag Editor \(p. 41\)](#).

Set up permissions

To make full use of Resource Groups and Tag Editor, you might need additional permissions to tag resources or to see a resource's tag keys and values. These permissions fall into the following categories:

- Permissions for individual services so that you can tag resources from those services and include them in resource groups.
- Permissions that are required to use the Tag Editor console
- Permissions that are required to use the AWS Resource Groups console and API.

If you are an administrator, you can provide permissions for your users by creating policies through the AWS Identity and Access Management (IAM) service. You first create IAM users or groups, and then apply the policies with the permissions that they need. For information about creating and attaching IAM policies, see [Working with policies](#).

Permissions for individual services

Important

This section describes permissions that are required if you want to tag resources from other service consoles and APIs, and add those resources to resource groups.

As described in [What are resource groups? \(p. 1\)](#), each resource group represents a collection of resources of specified types that share one or more tag keys or values. To add tags to a resource, you need the permissions required for the service to which the resource belongs. For example, to tag Amazon EC2 instances, you must have permissions to the tagging actions in that service's API, such as those listed in the [Amazon EC2 User Guide](#).

To make full use of the Resource Groups feature, you need other permissions that allow you to access a service's console and interact with the resources there. For examples of such policies for Amazon EC2, see [Example policies for working in the Amazon EC2 console](#) in the *Amazon EC2 User Guide for Linux Instances*.

Required permissions for Resource Groups and Tag Editor

To use Resource Groups and Tag Editor, the following permissions must be added to a user's policy statement in IAM. You can add either AWS-managed policies that are maintained and kept up-to-date by AWS, or you can create and maintain your own custom policy.

Using AWS managed policies for Resource Groups and Tag Editor permissions

AWS Resource Groups and Tag Editor support the following AWS managed policies that you can use to provide a predefined set of permissions to your users. You can attach these managed policies to any user, role or group just as you would any other policy that you create.

ResourceGroupsandTagEditorReadOnlyAccess

This policy grants the attached IAM user or role permission to call the read-only operations for both Resource Groups and Tag Editor. To read a resource's tags, you must also have permissions for that resource through a separate policy (see the following Important note).

ResourceGroupsandTagEditorFullAccess

This policy grants the attached IAM user or role permission to call any Resource Groups operation and the read and write tag operations in Tag Editor. To read or write a resource's tags, you must also have permissions for that resource through a separate policy (see the following Important note).

Important

The two previous policies grant permission to call the Resource Groups and Tag Editor operations and use those consoles. For Resource Groups operations, those policies are sufficient and grant all the permissions needed to work with any resource in the Resource Groups console. However, for tagging operations and the Tag Editor console, permissions are more granular. You must have permissions not only to invoke the operation, but also appropriate permissions to the specific resource whose tags you're trying to access. To grant that access to the tags, you must also attach one of the following policies:

- The AWS-managed policy [ReadOnlyAccess](#) grants permissions to the read-only operations for every service's resources. AWS automatically keeps this policy up to date with new AWS services as they become available.
- Many services provide a service-specific read-only AWS-managed policies that you can use to limit access to only the resources provided by that service. For example, Amazon EC2 provides [AmazonEC2ReadOnlyAccess](#).
- You could create your own policy that grants access to only the very specific read-only operations for the few services and resources you want your users to access. This policy use either an "allow list" strategy or a deny list strategy.

An allow list strategy takes advantage of the fact that access is denied by default until you **explicitly allow** it in a policy. So you can use a policy like the following example:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [ "resource-groups:*" ],
      "Resource": "arn:aws:resource-groups:*:123456789012:group/*"
    }
  ]
}
```

Alternatively, you could use a "deny list" strategy that allows access to all resources except those that you explicitly block.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": [ "resource-groups:*" ],
      "Resource": "arn:aws:resource-groups:*:123456789012:group/*"
    }
  ]
}
```

Adding Resource Groups and Tag Editor permissions manually

- `resource-groups:*` (This permission allows all Resource Groups actions. If you instead want to restrict actions that are available to a user, you can replace the asterisk with a [specific Resource Groups action](#), or to a comma-separated list of actions)
- `cloudformation:DescribeStacks`
- `cloudformation:ListStackResources`
- `tag:GetResources`
- `tag:TagResources`
- `tag:UntagResources`
- `tag:getTagKeys`
- `tag:getTagValues`
- `resource-explorer:*`

To use Resource Groups and Tag Editor in the console, you also need permission to run the `resource-groups:ListGroupResources` action. This permission is necessary for listing available resource types in the current Region. Using policy conditions with `resource-groups:ListGroupResources` is not currently supported.

Granting permissions for using AWS Resource Groups and Tag Editor

To add a policy for using AWS Resource Groups and Tag Editor to a user, do the following.

1. Open the [IAM console](#).
2. In the navigation pane, choose **Users**.
3. Find the user to whom you want to grant AWS Resource Groups and Tag Editor permissions. Choose the user's name to open the user properties page.
4. Choose **Add permissions**.
5. Choose **Attach existing policies directly**.
6. Choose **Create policy**.
7. On the **JSON** tab, paste the following policy statement.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "resource-groups:*",
        "cloudformation:DescribeStacks",
        "cloudformation:ListStackResources",
        "tag:GetResources",

```

```
        "tag:TagResources",
        "tag:UntagResources",
        "tag:getTagKeys",
        "tag:getTagValues",
        "resource-explorer:*"
    ],
    "Resource": "*"
  }
}
```

Note

This example policy statement grants permissions only for AWS Resource Groups and Tag Editor actions. It does not allow access to AWS Systems Manager tasks in the AWS Resource Groups console. For example, this policy does not grant permissions for you to use Systems Manager Automation commands. To perform Systems Manager tasks on resource groups, you must have Systems Manager permissions attached to your policy (such as `ssm:*`). For more information about granting access to Systems Manager, see [Configuring access to Systems Manager](#) in the *AWS Systems Manager User Guide*.

8. Choose **Review policy**.
9. Give the new policy a name and description. (for example, `AWSResourceGroupsQueryAPIAccess`).
10. Choose **Create policy**.
11. Now that the policy is saved in IAM, you can attach it to other users. For more information about how to add a policy to a user, see [Adding permissions by attaching policies directly to the user](#) in the *IAM User Guide*.

Learn more about AWS Resource Groups authorization and access control

Resource Groups supports the following.

- **Action-based policies.** For example, you can create a policy that allows users to perform [ListGroups](#) operations, but no others.
- **Resource-level permissions.** Resource Groups supports using [ARNs](#) to specify individual resources in the policy.
- **Authorization based on tags.** Resource Groups supports using [resource tags \(p. 21\)](#) in the condition of a policy. For example, you can create a policy that allows Resource Groups users full access to a group that you have tagged.
- **Temporary credentials.** Users can assume a role with a policy that allows AWS Resource Groups operations.

Resource Groups doesn't support resource-based policies.

Resource Groups doesn't use any service-linked roles.

For more information about how Resource Groups and Tag Editor integrate with AWS Identity and Access Management (IAM), see the following topics in the *AWS Identity and Access Management User Guide*.

- [AWS services that work with IAM](#)
- [Actions, resources, and condition keys for AWS Resource Groups](#)
- [Controlling access using policies](#)

Creating query-based groups in AWS Resource Groups

Topics

- [Types of resource group queries \(p. 8\)](#)
- [Build a tag-based query and create a group \(p. 10\)](#)
- [Create an AWS CloudFormation stack-based group \(p. 12\)](#)

Types of resource group queries

In AWS Resource Groups, a *query* is the foundation of a query-based group. You can base a resource group on one of two types of queries.

Tag-based

Tag-based queries include lists of resource types that are specified in the following format `AWS::service::resource`, and tags. *Tags* are keys that help identify and sort your resources in your organization. Optionally, tags include values for keys.

For a tag-based query, you also specify the tags that are shared by the resources that you want to be members of the group. For example, if you want to create a resource group that has all of the Amazon EC2 instances and Amazon S3 buckets that you are using to run the testing stage of an application, and you have instances and buckets that are tagged this way, choose the `AWS::EC2::Instance` and `AWS::S3::Bucket` resource types from the drop-down list, and then specify the tag key **Stage**, with a tag value of **Test**.

The syntax of the `ResourceQuery` parameter of a tag-based resource group contains the following elements:

- **Type**

This element indicates which kind of query defines this resource group. To create a tag-based resource group, specify the value `TAG_FILTERS_1_0`, as follows:

```
"Type": "TAG_FILTERS_1_0"
```

- **Query**

This element defines the actual query used to match against resources. It contains a string representation of a JSON structure with the following elements:

- **ResourceTypeFilters**

This element limits the results to only those resource types that match the filter. You can specify the following values:

- `"AWS::AllSupported"` – to specify that the results can include resources of any type that match the query and that are currently supported by the &ARG; service.
- `"AWS::service-id::resource-type"` – a comma separated list of resource-type specification strings with this format: , such as `"AWS::EC2::Instance"`.

- **TagFilters**

This element specifies key/value string pairs that are compared to the tags attached to your resources. Those with a tag key and value that match the filter are included in the group. Each filter consists of these elements:

- "Key" – a string with a key name. Only resources that have tags with a matching key name match and can be included in the group.
- "Values" – a string with a comma separated list of values for the specified key. Only resources with a matching tag key and a value that matches one in this list are members of the group.

All of these JSON elements must be combined into a single-line string representation of the JSON structure. For example, consider a `Query` with the following example JSON structure. This query is meant to match only Amazon EC2 instances that have a tag "Stage" with a value "Test".

```
{
  "ResourceTypeFilters": [ "AWS::EC2::Instance" ],
  "TagFilters": [
    {
      "Key": "Stage",
      "Values": [ "Test" ]
    }
  ]
}
```

That JSON can be represented as the following single-line string, and used as the value of the `Query` element. Because the value of a JSON structure must be a double-quoted string, you must escape any embedded double-quote characters or forward slash characters by preceding each with a backslash as shown here:

```
"Query": "{ \"ResourceTypeFilters\": [ \"AWS::AllSupported\" ], \"TagFilters\": [ { \"Key\": \"Stage\", \"Values\": [ \"Test\" ] } ] }"
```

The complete `ResourceQuery` string is then represented as shown here, as a CLI command parameter:

```
--resource-query '{"Type": "TAG_FILTERS_1_0", "Query": "{ \"ResourceTypeFilters\": [ \"AWS::AllSupported\" ], \"TagFilters\": [ { \"Key\": \"Stage\", \"Values\": [ \"Test\" ] } ] }"}
```

AWS CloudFormation stack-based

In an AWS CloudFormation stack-based query, you choose an AWS CloudFormation stack in your account in the current region, and then choose resource types in the stack that you want to be in the group. You can base your query on only one AWS CloudFormation stack. Resource Groups supports queries based on AWS CloudFormation stacks that have one of the following statuses.

Important

Only resources that are directly created as part of the stack in the query are included in the resource group. Resources created later by members of the AWS CloudFormation stack do not become members of the group. For example, if an auto-scaling group is created by AWS CloudFormation as part of the stack, then that auto-scaling group *is* a member of the group. However, an Amazon EC2 instance created by that auto-scaling group as part of its operation *is not* a member of the AWS CloudFormation stack-based resource group.

If you create a group based on an AWS CloudFormation stack, and the stack's status changes to one that is no longer supported as a basis for a group query, such as `DELETE_COMPLETE`, the resource group still exists, but it has no member resources.

After you create a resource group, you can perform tasks on the resources in the group.

The syntax of the `ResourceQuery` parameter of a CloudFormation stack-based resource group contains the following elements:

- **Type**

This element indicates which kind of query defines this resource group.

To create a AWS CloudFormation stack-based resource group, specify the value `CLOUDFORMATION_STACK_1_0`, as follows:

```
"Type": "CLOUDFORMATION_STACK_1_0"
```

- **Query**

This element defines the actual query used to match against resources. It contains a string representation of a JSON structure with the following elements:

- **ResourceTypeFilters**

This element limits the results to only those resource types that match the filter. You can specify the following values:

- `"AWS::AllSupported"` – to specify that the results can include resources of any type that match the query.
- `"AWS::service-id::resource-type"` – a comma separated list of resource-type specification strings with this format: , such as `"AWS::EC2::Instance"`.

- **StackIdentifier**

This element specifies the Amazon Resource Name (ARN) of the AWS CloudFormation stack whose resources you want to include in the group.

All of these JSON elements must be combined into a single-line string representation of the JSON structure. For example, consider a `Query` with the following example JSON structure. This query is meant to match only Amazon S3 buckets that are part of the specified AWS CloudFormation stack.

```
{
  "ResourceTypeFilters": [ "AWS::S3::Bucket" ],
  "StackIdentifier": "arn:aws:cloudformation:us-west-2:123456789012:stack/MyCloudFormationStackName/fb0d5000-aba8-00e8-aa9e-50d5cEXAMPLE"
}
```

That JSON can be represented as the following single-line string, and used as the value of the `Query` element. Because the value of a JSON structure must be a double-quoted string, you must escape any embedded double-quote characters or forward slash characters by preceding each with a backslash as shown here:

```
"Query": "{ \"ResourceTypeFilters\": [ \"AWS::S3::Bucket\" ], \"StackIdentifier\": \"arn:aws:cloudformation:us-west-2:123456789012:stack/MyCloudFormationStackName/fb0d5000-aba8-00e8-aa9e-50d5cEXAMPLE\" }
```

The complete `ResourceQuery` string is then represented as shown here, as a CLI command parameter:

```
--resource-query '{"Type": "CLOUDFORMATION_STACK_1_0", "Query": "{ \"ResourceTypeFilters\": [ \"AWS::S3::Bucket\" ], \"StackIdentifier\": \"arn:aws:cloudformation:us-west-2:123456789012:stack/MyCloudFormationStackName/fb0d5000-aba8-00e8-aa9e-50d5cEXAMPLE\" }' }
```

Build a tag-based query and create a group

The following procedures show you how to build a tag-based query and use it to create a resource group.

Console

1. Sign in to the [AWS Resource Groups console](#).
2. In the navigation pane, choose [Create Resource Group](#).
3. On the **Create query-based group** page, under **Group type**, choose the **Tag based** group type.
4. Under **Grouping criteria**, choose the resource types that you want to be in your resource group. You can have a maximum of 20 resource types in a query. For this walkthrough, choose **AWS::EC2::Instance** and **AWS::S3::Bucket**.
5. Still under **Grouping criteria**, for **Tags**, specify a tag key, or a tag key and value pair, to limit the matching resources to include only those that are tagged with your specified values. Choose **Add** or press **Enter** when you've finished your tag. In this example, filter for resources that have a tag key of **Stage**. The tag value is optional, but narrows the results of the query further. You can add multiple values for a tag key by adding an **OR** operator between tag values. To add more tags, choose **Add**. Queries assign an **AND** operator to tags, so any resource that matches the specified resource types and all specified tags is returned by the query.
6. Still under **Grouping criteria**, choose **Preview group resources** to return the list of EC2 instances and S3 buckets in your account that match the specified tag key or keys.
7. After you have the results that you want, create a group based on this query.
 - a. Under **Group details**, for **Group name**, type a name for your resource group.

A resource group name can have a maximum of 128 characters, including letters, numbers, hyphens, periods, and underscores. The name cannot start with **AWS** or **aws**. These are reserved. A resource group name must be unique in the current Region in your account.
 - b. (Optional) In **Group description**, enter a description of your group.
 - c. (Optional) In **Group tags**, add tag key and value pairs that apply only to the resource group, not the member resources in the group.

Group tags are useful if you plan to make this group a member of a larger group. Because specifying at least a tag key is required to create a group, be sure to add at least a tag key in **Group tags** to groups that you plan to nest into larger groups.
8. When you're finished, choose **Create group**.

AWS CLI & AWS SDKs

A tag-based group is based on a query of type `TAG_FILTERS_1_0`.

1. In an AWS CLI session, type the following, and then press **Enter**, replacing the values for group name, description, resource types, tag keys, and tag values with your own. Descriptions can have a maximum of 512 characters, including letters, numbers, hyphens, underscores, punctuation, and spaces. You can have a maximum of 20 resource types in a query. A resource group name can have a maximum of 128 characters, including letters, numbers, hyphens, periods, and underscores. The name cannot start with **AWS** or **aws**. These are reserved. A resource group name must be unique in your account.

At least one value for `ResourceTypeFilters` is required. To specify all resource types, use `AWS::AllSupported` as the `ResourceTypeFilters` value.

```
$ aws resource-groups create-group \
  --name resource-group-name \
  --resource-query '{"Type":"TAG_FILTERS_1_0","Query":{"ResourceTypeFilters\
":["resource_type1","resource_type2"],"TagFilters":{"Key":"Key1",\
"Values":["Value1","Value2"]},"Key":"Key2","Values":["Value1",\
"Value2"]}}}'
```


The following command is an example.

```
$ aws resource-groups create-group \
  --name my-resource-group \
  --resource-query '{"Type":"TAG_FILTERS_1_0","Query":{"ResourceTypeFilters\":[\"AWS::EC2::Instance\"],\"TagFilters\":{\"Key\":\"Stage\",\"Values\":[\"Test\"]}}}'
```

The following command is an example that includes all supported resource types.

```
$ aws resource-groups create-group \
  --name my-resource-group \
  --resource-query '{"Type":"TAG_FILTERS_1_0","Query":{"ResourceTypeFilters\":[\"AWS::AllSupported\"],\"TagFilters\":{\"Key\":\"Stage\",\"Values\":[\"Test\"]}}}'
```

2. The following are returned in the response to the command.
 - A full description of the group you have created.
 - The resource query that you used to create the group.
 - The tags that are associated with the group.

Create an AWS CloudFormation stack-based group

The following procedures show you how to build a stack-based query and use it to create a resource group.

Console

1. Sign in to the [AWS Resource Groups console](#).
2. In the navigation pane, choose **Create Resource Group**.
3. On **Create query-based group**, under **Group type**, choose the **CloudFormation stack based** group type.
4. Choose the stack that you want to be the basis of your group. A resource group can be based on only one stack. To filter the list of stacks, start typing the name of the stack. Only stacks with supported statuses appear in the list.
5. Choose resource types in the stack that you want to include in the group. For this walkthrough, keep the default, **All supported resource types**. For more information about which resource types are supported and can be in the group, see [Resources you can use with AWS Resource Groups and Tag Editor \(p. 41\)](#).
6. Choose **View group resources** to return the list of resources in the AWS CloudFormation stack that match your selected resource types.
7. After you have the results that you want, create a group based on this query.
 - a. Under **Group details**, for **Group name**, type a name for your resource group.

A resource group name can have a maximum of 128 characters, including letters, numbers, hyphens, periods, and underscores. The name cannot start with **AWS** or **aws**. These are reserved. A resource group name must be unique in the current Region in your account.
 - b. (Optional) In **Group description**, enter a description of your group.
 - c. (Optional) In **Group tags**, add tag key and value pairs that apply only to the resource group, not the member resources in the group.

Group tags are useful if you plan to make this group a member of a larger group. Because specifying at least a tag key is required to create a group, be sure to add at least a tag key in **Group tags** to groups that you plan to nest into larger groups.

8. When you're finished, choose **Create group**.

AWS CLI & AWS SDKs

An AWS CloudFormation stack-based group is based on a query of type `CLOUDFORMATION_STACK_1_0`.

1. Run the following command, replacing the values for group name, description, stack identifier, and resource types with your own. Descriptions can have a maximum of 512 characters, including letters, numbers, hyphens, underscores, punctuation, and spaces.

If you do not specify resource types, Resource Groups includes all supported resource types in the stack. You can have a maximum of 20 resource types in a query. A resource group name can have a maximum of 128 characters, including letters, numbers, hyphens, periods, and underscores. The name cannot start with `AWS` or `aws`. These are reserved. A resource group name must be unique in your account.

The `stack_identifier` is the stack ARN, as shown in the example command.

```
$ aws resource-groups create-group \
  --name group_name \
  --description "description" \
  --resource-query
  '{"Type":"CLOUDFORMATION_STACK_1_0","Query":{"StackIdentifier\":"
  \"stack_identifier\",\"ResourceTypeFilters\":[\"resource_type1\",
  \"resource_type2\"]}}'
```

The following command is an example.

```
$ aws resource-groups create-group \
  --name My-CFN-stack-group \
  --description "My first CloudFormation stack-based group" \
  --resource-query
  '{"Type":"CLOUDFORMATION_STACK_1_0","Query":{"StackIdentifier\":"
  \\"arn:aws:cloudformation:us-west-2:123456789012:stack/AWStestuseraccount/
  fb0d5000-aba8-00e8-aa9e-50d5cEXAMPLE\", \"ResourceTypeFilters\":"
  \["AWS::EC2::Instance\", \"AWS::S3::Bucket\"]}}'
```

2. The following are returned in the response to the command.
 - A full description of the group you have created.
 - The resource query that you used to create the group.

Updating groups in AWS Resource Groups

To update a tag-based resource group in Resource Groups, you can edit the query and tags that are the basis of your group. You can add and remove resources from your group only by applying changes to the query or tags. You cannot select specific resources to add to or remove from your group. The best way to add or remove a specific resource from a group is to edit the resource's tags. Then verify that your resource group tag query either includes or omits the tag, depending on whether you want the resource in your group.

To update an AWS CloudFormation stack-based resource group, you can choose a different stack. You can also add or remove resource types from the stack that you want to be part of the group. To change the resources that are available in the stack, update the AWS CloudFormation template used to create the stack, and then update the stack in AWS CloudFormation. For more information about how to update an AWS CloudFormation stack, see [AWS CloudFormation stacks updates](#) in the *AWS CloudFormation User Guide*.

In the AWS CLI, you update groups in two commands.

- `update-group`, which you run to update a group's description.
- `update-group-query`, which you run to update the resource query and tags that determine the group's member resources.

In the console, you cannot change an AWS CloudFormation stack-based group to a tag-based query group, or vice versa. However, you can do this by using the Resource Groups API, including in the AWS CLI.

Update tag-based query groups

Console

Update a tag-based group by changing the resource types or tags in the query on which the group is based. You can also add or change the group's description.

1. Sign in to the [AWS Resource Groups console](#).
2. In the navigation pane, under **Saved resource groups**, choose the name of the group, and then choose **Edit**.

Note

You can update only resource groups that you own. The **Owner** column shows account ownership for each resource group. Any groups with an account owner other than the one you're signed in to were created in AWS License Manager. For more information, see [Host resource groups in AWS License Manager](#) in the *License Manager User Guide*.

3. On the **Edit group** page, under **Grouping criteria**, add or remove resource types. You can have a maximum of 20 resource types in a query. To remove a resource type, choose **X** on the resource type's label. Choose **View group resources** to see how the changes affect your group's resource members. In this walkthrough, we add the resource type **AWS::RDS::DBInstance** to the query.
4. Still under **Grouping criteria**, edit the tags as needed. In this example, we filter for resources that have a tag key of **Stage** and add a tag value of **Test**. The tag value is optional, but narrows the results of the query further. To remove a tag, choose **X** on the tag's label.
5. In **Additional information**, you can edit the group description. You cannot edit a group's name after the group has been created.
6. (Optional) In **Group tags**, you can add or remove tags. Group tags are metadata about your resource group. They do not affect member resources. To change the resources that are returned by the resource group's query, edit the tags found under **Grouping criteria**.

Group tags are useful if you plan to make this group a member of a larger group. Specifying at least a tag key is required to create a group. Therefore, be sure to add at least a tag key in **Group tags** to groups that you plan to nest into larger groups.

7. Choose **Preview group resources** to retrieve the updated list of EC2 instances, S3 buckets, and Amazon RDS database instances in your account that match the specified tag keys. If you do not see resources in the list that you expect, be sure that the resources are tagged with tags that you specified in **Grouping criteria**.
8. When you are finished, choose **Save changes**.

AWS CLI & AWS SDKs

In the AWS CLI, you update a group's query and update a resource group's description by using two different commands. You cannot edit an existing group's name. In the AWS CLI, you can change a tag-based group to a CloudFormation stack-based group, or vice versa.

1. If you do not want to change the description of your group, skip this step and go on to the next. In an AWS CLI session, type the following, and then press **Enter**, replacing the values for group name and description with your own.

```
$ aws resource-groups update-group \
  --group-name resource-group-name \
  --description "description_text"
```

The following command is an example.

```
$ aws resource-groups update-group \
  --group-name my-resource-group \
  --description "EC2 instances, S3 buckets, and RDS DBs that we are using for the
  test stage."
```

The command returns a full, updated description of the group.

2. To update the query and tags of a group, type the following command. Replace the values for group name, resource types, tag keys, and tag values with your own. Then press **Enter**. You can have a maximum of 20 resource types in a query.

```
$ aws resource-groups update-group-query \
  --group-name resource-group-name \
  --resource-query '{"Type":"TAG_FILTERS_1_0","Query":{"ResourceTypeFilters\
  \":[\"resource_type1\",\"resource_type2\"],\"TagFilters\":{\"Key\":\"Key1\",\
  \"Values\":[\"Value1\",\"Value2\"]},{\"Key\":\"Key2\",\"Values\":[\"Value1\",\
  \"Value2\"]}}}'
```

The following command is an example.

```
$ aws resource-groups update-group-query \
  --group-name my-resource-group \
  --resource-query '{"Type":"TAG_FILTERS_1_0","Query":{"ResourceTypeFilters\
  \":[\"AWS::EC2::Instance\", \"AWS::S3::Bucket\", \"AWS::RDS::DBInstance\"],\"TagFilters\
  \":{\"Key\":\"Stage\", \"Values\":[\"Test\"]}}}'
```

The command returns the updated query as a result.

Update an AWS CloudFormation stack-based group

Console

You cannot change an AWS CloudFormation stack-based group to a tag-based group in the AWS Management Console. However, you can change the stack on which the group is based, or change the stack resource types that you want to include in the group. You can also add or change the group's description.

1. Sign in to the [AWS Resource Groups console](#).
2. In the navigation pane, under **Saved resource groups**, choose the name of the group, and then choose **Edit**.

3. **Note**

You can update only resource groups that you own. The **Owner** column shows account ownership for each resource group. Any groups with an account owner other than the one you're signed in to were created in AWS License Manager. For more information, see [Host resource groups in AWS License Manager](#) in the *License Manager User Guide*.

4. On the **Edit group** page, under **Grouping criteria**, to change the stack on which your group is based, choose the stack from the drop-down list. A resource group can be based on only one stack. To filter the list of stacks, start typing the name of the stack. Only stacks with supported statuses appear in the list. For a list of supported statuses, see [Creating query-based groups in AWS Resource Groups](#) (p. 8) in this guide.
5. Add or remove resource types. Only resource types that are available in the stack are shown in the drop-down list. The default is **All supported resource types**. You can have a maximum of 20 resource types in a query. To remove a resource type, choose **X** on the resource type's label. For more information about which resource types are supported and can be in the group, see [Resources you can use with AWS Resource Groups and Tag Editor](#) (p. 41).
6. Choose **Preview group resources** to retrieve the list of resources in the AWS CloudFormation stack that match your selected resource types.
7. In **Additional information**, you can edit the group description. You cannot edit a group's name after the group has been created.
8. In **Group tags**, add or remove tags. Group tags are metadata about your resource group. They do not affect member resources. To change the resources that are returned by the resource group's query, edit tags in **Grouping criteria**.

Group tags are useful if you plan to make this group a member of a larger group. Specifying at least a tag key is required to create a group. Therefore, be sure to add at least a tag key in **Group tags** to groups that you plan to nest into larger groups.

9. When you are finished, choose **Save changes**.

AWS CLI & AWS SDKs

In the AWS CLI, you update a group's query and update a resource group's description by using two different commands. You cannot edit an existing group's name. In the AWS CLI, you can change a tag-based group to a CloudFormation stack-based group, or vice versa.

1. If you do not want to change the description of your group, skip this step and go on to the next. Run the following command, replacing the values for group name and description with your own.

```
$ aws resource-groups update-group \
  --group-name "resource-group-name" \
  --description "description_text"
```

The following command is an example.

```
$ aws resource-groups update-group \
  --group-name "My-CFN-stack-group" \
  --description "EC2 instances, S3 buckets, and RDS DBs that we are using for the
test stage."
```

The command returns a full, updated description of the group.

2. To update the query and tags of a group, run the following command. Replace the values for group name, stack identifier, and resource types with your own. To add resource types, provide the full list of resource types in the command, not only resource types you are adding. You can have a maximum of 20 resource types in a query.

The `stack_identifier` is the stack ARN, as shown in the example command.

```
$ aws resource-groups update-group-query \
  --group-name resource-group-name \
  --description "description" \
  --resource-query
'{"Type":"CLOUDFORMATION_STACK_1_0","Query":{"\"StackIdentifier\":
\"stack_identifier\",\"ResourceTypeFilters\":[\"resource_type1\",
\"resource_type2\"]}}'
```

The following command is an example.

```
$ aws resource-groups update-group-query \
  --group-name "my-resource-group" \
  --description "Updated CloudFormation stack-based group" \
  --resource-query
'{"Type":"CLOUDFORMATION_STACK_1_0","Query":{"\"StackIdentifier\":
\"arn:aws:cloudformation:us-west-2:810000000000:stack/AWStestuseraccount
\/fb0d5000-aba8-00e8-aa9e-50d5cEXAMPLE\",\"ResourceTypeFilters\":
[\"AWS::EC2::Instance\", \"AWS::S3::Bucket\"]}}'
```

The command returns the updated query as a result.

Deleting resource groups from AWS Resource Groups

You can use the [AWS Resource Groups console](#) or the AWS CLI to delete resource groups from AWS Resource Groups. Deleting a resource group does not delete the resources that are members of the group or tags on member resources. It deletes only the group structure and any group-level tags.

Console

To delete resource groups

1. Sign in to the [AWS Resource Groups console](#).
2. Choose the name of the resource group that you want to delete.
3. On the group's detail page, choose **Delete**.
4. When you are prompted to confirm the deletion, choose **Delete**.

AWS CLI & AWS SDKs

To delete resource groups

1. Run the following command, replacing `resource_group_name` with the name of your group.

```
$ aws resource-groups delete-group \
  --group-name resource_group_name
```

2. When you are prompted to confirm the deletion, type yes, and then press **Enter**.

AWS services that work with AWS Resource Groups

You can use the following AWS services with AWS Resource Groups.

AWS service	Using with Resource Groups
AWS CloudFormation – Create resource groups in AWS CloudFormation by using a stack template.	<p>Provision and organize AWS resources at the same time. Organize resources by tags. Organize resources from another stack. Gather insights on your AWS resources in resource groups using Amazon CloudWatch or take operational actions using AWS Systems Manager.</p> <p>For more information, see ResourceGroups resource type reference in the <i>AWS CloudFormation User Guide</i>.</p>
CloudTrail – Capture all resource group actions using AWS CloudTrail.	<p>Capture information about actions performed on your resource groups including details like who performed the action (IAM user or role, or an AWS service), when the action was performed, where the action occurred (the source IP address) and more. These records can then be used for analysis or to trigger follow-up actions.</p> <p>For more information, see Viewing events with CloudTrail Event history.</p>
Amazon CloudWatch – Enable real-time monitoring of your AWS resources and the applications you run on AWS.	<p>Focus your view to display metrics and alarms from a single resource group.</p> <p>For more information, see Focus on metrics and alarms in a resource group in the <i>Amazon CloudWatch User Guide</i>.</p>
Amazon CloudWatch application insights – Detect common problems with your .NET and SQL Server-based applications.	<p>Monitor your .NET and SQL Server application resources that belong to a resource group.</p> <p>For more information, see Supported application components in the <i>Amazon CloudWatch User Guide</i>.</p>
Amazon DynamoDB table groups – Organize your DynamoDB tables into logical groupings so you can more easily manage your resources.	<p>Create, edit, and delete groups of DynamoDB tables from the DynamoDB Action menu.</p> <p>For more information, see the Amazon DynamoDB Developer Guide.</p>
Amazon EC2 dedicated hosts – Use your existing per-socket, per-core, or per-VM software licenses, including Windows Server, Microsoft SQL Server, SUSE, and Linux Enterprise Server.	<p>Launch Amazon EC2 instances into host resource groups to help maximize your utilization of Dedicated Hosts.</p> <p>For more information, see Working with dedicated hosts in the <i>Amazon EC2 User Guide for Linux Instances</i>.</p>
Amazon EC2 capacity reservations – Reserve capacity for your Amazon EC2 instances to be	<p>Launch your Amazon EC2 instances into resource groups that contain one or more capacity</p>

AWS service	Using with Resource Groups
used when you need it. You can specify attributes for the capacity reservation so that it only works with Amazon EC2 instances that launch with matching attributes.	<p>reservations. If the group doesn't have a capacity reservation with matching attributes and available capacity for a requested instance, the instance runs as an on-demand instance. If you later add a matching capacity reservation to the targeted group, the instance is automatically matched with and moved into the reserved capacity.</p> <p>For more information, see Work with Capacity Reservation groups in the <i>Amazon EC2 User Guide for Linux Instances</i>.</p>
AWS License Manager – Streamline the process of bringing software vendor licenses to the cloud.	<p>Configure a host resource group to enable License Manager to manage your Dedicated Hosts.</p> <p>For more information, see Host Resource Groups in License Manager in the <i>License Manager User Guide</i>.</p>
AWS Resilience Hub – Prepare and protect your applications from disruptions.	<p>Discover your applications that are defined using Resource Groups.</p> <p>For more information, see Measure and Improve Your Application Resilience with AWS Resilience Hub in the <i>AWS News Blog</i>.</p>
AWS Resource Access Manager – Share specified AWS resources that you own with other accounts.	<p>Share host resource groups using AWS RAM.</p> <p>For more information, see Shareable resources in the <i>AWS RAM User Guide</i>.</p>
AWS Service Catalog AppRegistry – Define and manage your applications and their metadata.	<p>When you create an application in AWS Service Catalog AppRegistry, that service automatically creates an resource group for that application. The application resource group is a collection of all of the resources in your application. The service also creates a AWS CloudFormation stack-based resource group for every stack associated with the application.</p> <p>For more information, see Using AppRegistry in the <i>AWS Service Catalog Administrator Guide</i>.</p>

AWS service	Using with Resource Groups
AWS Systems Manager – Enable visibility and control of your AWS resources.	<p>Gather operational insights and take bulk actions on your applications that are based on resource groups. In the AWS Systems Manager console, the Application Manager Custom applications page automatically imports and displays operations data for applications that are based on resource groups. You can use the information in Application Manager to help you determine which resources in an application are compliant and working correctly and which resources require action.</p> <p>For more information, see Working with applications in Application Manager in the <i>AWS Systems Manager User Guide</i>.</p>
Amazon VPC Network Access Analyzer – Identify unwanted network access to your resources on AWS.	<p>You can specify the sources and destinations for your network access requirements by using AWS Resource Groups. This lets you govern network access across your AWS environment, independent of how you configure your network.</p> <p>For more information, see Use Resource Groups with Network Access Scopes in the <i>Amazon Virtual Private Cloud User Guide</i>.</p>

Tagging your AWS resources

Tags are key and value pairs that act as metadata for organizing your AWS resources. With most AWS resources, you have the option of adding tags when you create the resource, whether it's an Amazon EC2 instance, an Amazon S3 bucket, or other resource. However, you can also add tags to multiple, supported resources at once by using Tag Editor. You build a query for resources of various types, and then add, remove, or replace tags for the resources in your search results. Tag-based queries assign an AND operator to tags, so any resource that matches the specified resource types and all specified tags is returned by the query.

Important

Do not store personally identifiable information (PII) or other confidential or sensitive information in tags. We use tags to provide you with billing and administration services. Tags are not intended to be used for private or sensitive data.

You can tag [supported resources \(p. 41\)](#) by using Tag Editor, and some additional resources by using the tagging functionality provided by the service console in which you create and manage the resource. See the documentation for each service to discover what tagging functionality that service provides.

Tags and Attribute-based access control (ABAC)

Tags can be an important part of your AWS access control strategy. To use tags as the "attributes" in an attribute-based access control strategy, see [Controlling access to AWS resources using tags](#) and [Controlling access to and for IAM users and roles using tags](#), both in the *IAM User Guide*.

There is also a much more comprehensive tutorial that shows how to grant access to different projects and groups using tags at [IAM tutorial: Define permissions to access AWS resources based on tags](#) in the *IAM User Guide*.

Also, if you use a SAML-based identity provider (IdP) for single sign-in to AWS, you can configure the SAML IdP to attach the tags to the assumed roles that it delivers to your users for access. For more information, see [IAM tutorial: Use SAML session tags for ABAC](#) in the *AWS Identity and Access Management User Guide*.

Topics

- [Tag Editor \(p. 21\)](#)
- [Using tags in IAM permission policies \(p. 33\)](#)
- [AWS Organizations tag policies \(p. 35\)](#)

Tag Editor

Tags are words or phrases that act as metadata that you can use to identify and organize your AWS resources. A resource can have up to 50 user-applied tags. It can also have read-only system tags. Each tag consists of a key and one optional value. For more information, see [AWS tagging strategies](#) in *AWS Answers*, and [Using cost allocation tags](#) in the *AWS Billing and Cost Management User Guide*.

Important

Do not store personally identifiable information (PII) or other confidential or sensitive information in tags. We use tags to provide you with billing and administration services. Tags are not intended to be used for private or sensitive data.

You can add tags to resources when you create the resource. You can use the resource's service console or API to add, change, or remove those tags one resource at a time. To add tags to—or edit or delete tags of—multiple resources at once, use Tag Editor. With Tag Editor, you search for the resources that you want to tag, and then manage tags for the resources in your search results.

To start Tag Editor

1. Sign in to the AWS Management Console.
2. On the navigation bar, choose **Services**. Then, under **Management & Governance**, choose **Resource Groups & Tag Editor**. In the navigation pane on the left, choose **Tag Editor**.

Direct link: [AWS Tag Editor console](#)

Not all resources can have tags applied. For information about which resources Tag Editor supports, see [Resources you can use with AWS Resource Groups and Tag Editor \(p. 41\)](#). If a resource type that you want to tag is not supported, be sure to let AWS know by choosing the **Feedback** tool in the lower left corner of the console window.

For information about permissions and roles that are required to tag resources, see [Set up permissions \(p. 4\)](#).

Topics

- [Find resources to tag \(p. 22\)](#)
- [Manage tags \(p. 26\)](#)
- [Troubleshooting tag changes \(p. 32\)](#)

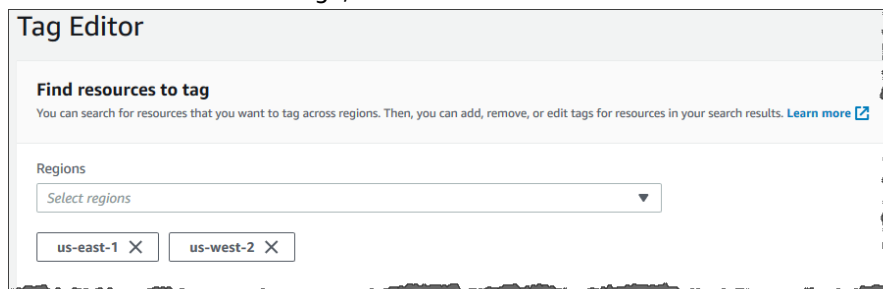
Find resources to tag

With Tag Editor, you build a query to find resources in one or more AWS Regions that are available for tagging. You can choose up to 20 individual resource types, or build a query on **All resource types**. Your query can include resources that already have tags, or resources that have no tags. For more information, see [Resources you can use with AWS Resource Groups and Tag Editor \(p. 41\)](#).

After you find resources to tag, you can use Tag Editor to add tags, or view, edit, or delete tags.

To find resources to tag

1. Sign in to the [AWS Management Console](#), choose **Resource Groups**, and then choose **Tag Editor**.
2. *(optional)* Choose regions in which to search for resources to tag. By default, your current region is selected. For this walkthrough, choose **us-east-1** and **us-west-2**.



3. Choose at least one resource type from the **Resource type** drop-down list. You can add or edit tags for up to 20 individual resource types at a time, or choose **All resource types**. For this walkthrough, choose **AWS::EC2::Instance** and **AWS::S3::Bucket**.

4. (optional) In the **Tags** fields, enter a tag key, or a tag key and value pair, to limit the resources in the current AWS Region to only those that are tagged with your specified values. As you type a tag key, matching tag keys in the current region appear in a list below; you can choose a tag key from the list. Tag Editor auto-completes the tag key for you as you type enough characters to match an existing key. Choose **Add** or press **Enter** when you've finished your tag. In this example, filter for resources that have a tag key of **Stage**. The tag value is optional, but narrows the results of the query further. To add more tags, choose **Add**. Queries assign an AND operator to tags, so any resource that matches the specified resource type and all specified tags is returned by the query.

To find resources with multiple values for a tag key, add another tag with the same key to the query, but specify a different value. The results include all resources that are tagged with the same tag key and that have any of the selected values. The search is case sensitive.

Leave the **Tags** boxes empty to find all resources of the specified type in the current AWS Region. This query returns resources that have any tags, and includes those that have no tags. To remove a tag from your query, choose **X** on the tag's label.

To find resources that have an empty value for a tag, choose **(empty value)** when your cursor is in the tag value box.


Note

Before you can find resources with the specified tags, they must have been applied to at least one resource of the specified type in the current AWS Region.



5. When your query is ready, choose **Search resources**. Results are displayed in the **Resource search results** area.

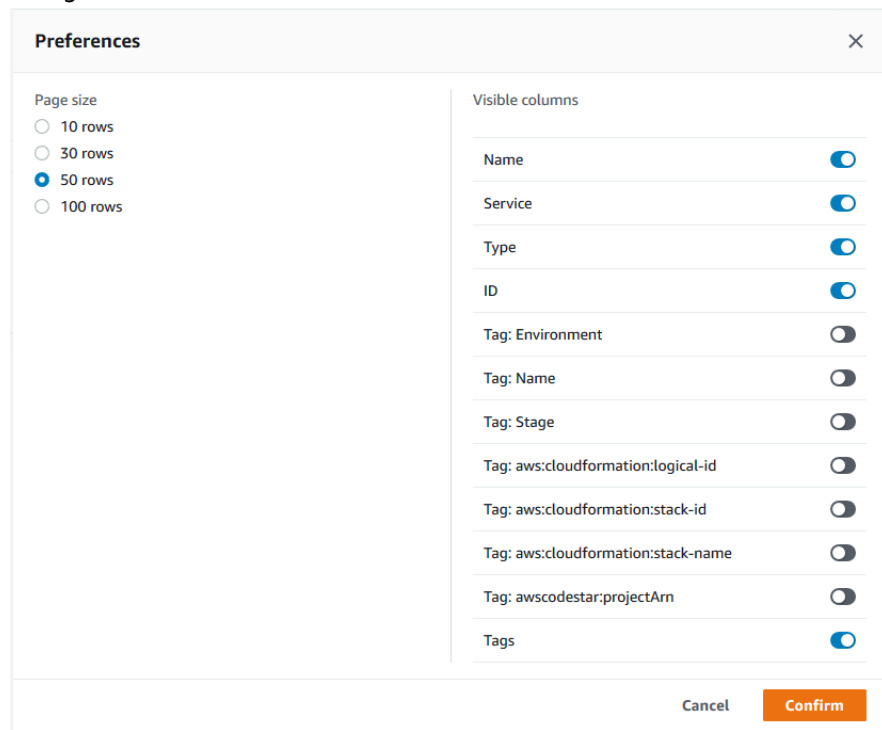
Name	Service	Type	Region	ID	Tag Name	Total tags
<input checked="" type="checkbox"/> EC2 Instance i-0...	EC2	Instance	us-east-1	i-0...	-test-ubuntu-pi	2
<input checked="" type="checkbox"/> EC2 Instance i-0...	EC2	Instance	us-east-1	i-0...	-java-ec2-web-WebApp	4
<input checked="" type="checkbox"/> S3 Bucket codestar-us-east-1-...	S3	Bucket	us-east-1	codestar-us-east-1-...	-java-ec2-web-S3Bucket	4
<input checked="" type="checkbox"/> S3 Bucket codestar-us-east-1-...	S3	Bucket	us-east-1	codestar-us-east-1-...	-nodewebapp-S3Bucket	3
<input type="checkbox"/> S3 Bucket codestar-us-east-1-...	S3	Bucket	us-east-1	codestar-us-east-1-...	-S3Bucket	1
<input type="checkbox"/> EC2 Instance i-0...	EC2	Instance	us-east-1	i-0...	-feb-node-ec2-WebApp	2
<input type="checkbox"/> S3 Bucket codepipeline-consolehookup-us-east-1-...	S3	Bucket	us-east-1	codepipeline-consolehookup-us-east-1-...	-consolehookup-S3Bucket	1
<input checked="" type="checkbox"/> S3 Bucket cloudtrail-test-2018	S3	Bucket	us-west-2	cloudtrail-test-2018	-	4

To filter a large number of resources, enter any filter text, such as part of the name of a resource, in **Filter resources**.

6. (optional) To configure the columns that Tag Editor displays in your resource search results, choose the **Preferences** gear icon  in the **Resource search results**.

On the **Preferences** page, choose the number of rows that you want displayed in your search results.

Turn on columns that you want Tag Editor to display in your results. You can show a column for each tag that occurs in your search results or a selected subset of your search results. You can do this any time after you find resources to tag. To enable a column, choose the switch icon next to the tag and change it from off  to on .



When you are finished configuring visible columns and number of displayed rows, choose **Confirm**.

View and edit tags for a selected resource

Tag Editor shows you the existing tags on selected resources that are in the results of your **Find resources to tag** query.

If you enabled any of the **Tag** columns as described in the previous section, then you can see the current value of that tag for each resource in the search results.

Note

This topic explains how to edit the tag for an *individual* resource. You can also bulk edit tags for several selected resources at the same time. For more information, see [Manage tags \(p. 26\)](#).

To edit tags inline in the search results table

1. Choose the value for the tag on the resource that you want to edit.

Note

- If the chosen resource currently does not have a tag with the chosen key, the value displays as **(not tagged)**.
- If the chosen resource does have a tag with the chosen key but without a value, the value displays as '-'.

In the following example, the column for the tag **Env** and with the current value of **Prod** was chosen.

Change value for tag: **Env**

Q Prod

Enter a new tag value or choose an existing value.

Remove tag

Tag: Env

Prod

2. You can enter a new value or choose from any of the values already present on other resources with this tag. You can also delete the tag from this one resource by choosing **Remove tag**.

To view all tags for an individual resource

1. In the results of your **Find resources to tag** query, choose the number in the **Tags** column for any resource for which you want to view existing tags. Resources with a dash in the **Tags** column do not have existing tags.

Resource search results (8)

Choose up to 500 resources for which you want to edit tags.

Export 8 resources to CSV Manage tags of selected resources

Filter resources

<input type="checkbox"/>	Name	Service	Type	Region	ID	Tag Name	Tags
<input type="checkbox"/>	EC2 Instance I-O	EC2	Instance	us-east-1	I-O	-test-ubuntu-ps	3
<input type="checkbox"/>	EC2 Instance I-O	EC2	Instance	us-east-1	I-O	-java-ec2-web-WebApp	3

2. View existing tags in **Resource tags**. You can also open this window on the **Manage tags** page, when you are changing or removing tags.

Resource tags

Tags that are assigned to the resource.

Custom tags

Tag key	Tag value
Environment	jm-java-ec2-web-WebApp
Name	jm-java-ec2-web-WebApp
Stage	Test

System tags (Read only)

Tag key	Tag value
aws:cloudformation:logical-id	WebApp01
aws:cloudformation:stack-id	arn:aws:cloudformation:us-east-1:;stack/awscodestar-jm-java-ec2-
aws:cloudformation:stack-name	awscodestar-jm-java-ec2-web-infrastructure

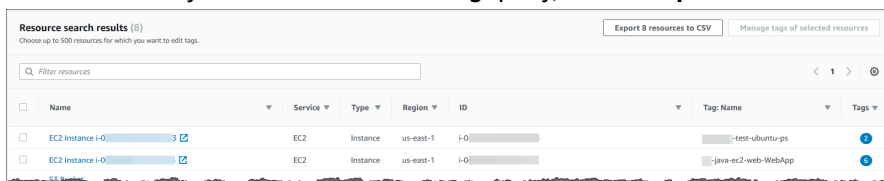
Note

If you don't see a tag that you recently applied to a resource, try refreshing your browser window.

Export Results to CSV

You can export the results of a **Find resources to tag** query to a comma-separated values (CSV) file. The CSV file includes the resource names, services, region, resource IDs, the total number of tags, and a column for each unique tag key in the collection. The CSV file can help you develop a tagging strategy for resources in your organization, or determine where there are overlaps or inconsistencies in tagging across resources.

1. In the results of your **Find resources to tag** query, choose **Export resources to CSV**.



2. When you are prompted by your browser, choose to open the CSV file (typically in Microsoft Excel), or save it to a convenient location.

Related information

- [AWS Tagging Strategies](#)
- [Using cost allocation tags](#)
- [Tag Editor \(p. 21\)](#)

Manage tags

After you [find the resources \(p. 22\)](#) that you want to tag, you can add, remove, and edit the tags for some or all of your search results. Tag Editor shows you any tags that are attached to resources, and whether those tags were added in Tag Editor or by using the resource's service console or by using the API.

Important

Do not store personally identifiable information (PII) or other confidential or sensitive information in tags. We use tags to provide you with billing and administration services. Tags are not intended to be used for private or sensitive data.

Other ways to manage your tags

This topic discusses tagging resources by using the Tag Editor in the AWS Management Console. However, there are other tools you can use to manage the tags on your AWS resources:

- You can type or script commands at your shell prompt by using the [resourcegroupstaggingapi commands](#) in the AWS Command Line Interface (AWS CLI).
- You can create and run PowerShell scripts by using the [AWS Resource Groups Tagging API](#) in the AWS Tools for PowerShell.
- You can create and run programs with any of the available [AWS SDKs](#) by using the [Resource groups tagging APIs](#), such as the [tagging APIs for Python](#) or the [tagging APIs for Java](#).

When you add, remove, or edit existing tags, you are changing tags only on those resources that you select in of the results of your **Find resources to tag** query. You can select up to 500 resources on which to manage tags.

Topics

- [Add tags to selected resources \(p. 27\)](#)
- [Edit tags of selected resources \(p. 29\)](#)
- [Remove tags from selected resources \(p. 31\)](#)
- [Retry failed tag changes \(p. 32\)](#)
- [Related information \(p. 32\)](#)

Add tags to selected resources

You can use Tag Editor to add tags to selected resources that are in the results of your **Find resources to tag** query.

Note

This topic describes how to bulk edit the tags for *multiple* resources. You can also edit the tag values for an individual resource. For more information, see [View and edit tags for a selected resource \(p. 24\)](#).

1. In the results of your **Find resources to tag** query, select the check boxes next to the resources you want to add tags to. Enter a text string in **Filter resources** to filter for part of a resource's name, ID, tag keys, or tag values. In the **Tags** column, note that resources in the results already have tags applied to them. In the following example, the first selected EC2 instance already has two tags.

Resource search results (4 selected of 8)
Choose up to 500 resources for which you want to edit tags.

Export 8 resources to CSV Manage tags of selected resources

Filter resources

	Name	Service	Type	Region	ID	Tag Name	Total tags
<input checked="" type="checkbox"/>	EC2 Instance i-O-1	EC2	Instance	us-east-1	i-O-1	-test-ubuntu-pis	2
<input checked="" type="checkbox"/>	EC2 Instance i-O-1	EC2	Instance	us-east-1	i-O-1	-java-ec2-web-WebApp	6
<input checked="" type="checkbox"/>	S3 Bucket codestar-us-east-1-jm-java-ec2-web-pipe	S3	Bucket	us-east-1	codestar-us-east-1-jm-java-ec2-web-pipe	-java-ec2-web-S3Bucket	6
<input type="checkbox"/>	S3 Bucket codestar-us-east-1-jm-nodewebappla-app	S3	Bucket	us-east-1	codestar-us-east-1-jm-nodewebappla-app	-nodewebappla-WebsiteS3Bucket	3
<input type="checkbox"/>	S3 Bucket codestar-us-east-1-jm-mc-ca-pipe	S3	Bucket	us-east-1	codestar-us-east-1-jm-mc-ca-pipe	-S3Bucket	1
<input type="checkbox"/>	EC2 Instance i-O-1-c	EC2	Instance	us-east-1	i-O-1-c	-feb-node-ec2-WebApp	7
<input type="checkbox"/>	S3 Bucket codepipeline-consolehookup-us-east-1	S3	Bucket	us-east-1	codepipeline-consolehookup-us-east-1	consolehookup-S3Bucket	1
<input checked="" type="checkbox"/>	S3 Bucket -cloudtrail-test-2018	S3	Bucket	us-west-2	-cloudtrail-test-2018	-	4

2. Check the box for one or more resources, and then choose **Manage tags of the selected resources**.
3. On the **Manage tags** page, view the tags on the resources that you selected. Although your original query returned more resources, note that you are adding tags only to the resources that you selected in step 1. Choose **Add tag**.

Manage tags

Selected resources (4)
View and edit the tags of selected resources.

Filter resources

Name	Service	Type	Region	ID	Tag Name	Total tags
EC2 Instance i-0...	EC2	Instance	us-east-1	i-0...	test-ubuntu-ps	2
EC2 Instance i-0...	EC2	Instance	us-east-1	i-0...	jm-java-ec2-web-WebApp	1
S3 Bucket aws-codestar-us-east-1...	S3	Bucket	us-east-1	aws-codestar-us-east-1...	jm-java-ec2-web-pipe	1
S3 Bucket arg-cloudtrail-test-2018	S3	Bucket	us-west-2	arg-cloudtrail-test-2018	-	1

Edit tags of all selected resources
You can override the tags of all selected resources, or add new tags to them. [Learn more](#)

Tag key

Department

Environment

Key

Name

Stage

Value

awscodestar:projectArn

Add tag

Tag value - optional

Selected resources have different tag values

Remove tag

Test

Cancel

Review and apply tag changes

- Enter a tag key and an optional tag value. In this walkthrough, we add the tag key **Team** and the tag value **Development**.

Edit tags of selected resources
You can override the tags of all selected resources, or add new tags to them.

Tag key

Name

Purpose

Stage

Team

Add tag

Tag value - optional

Linux

Kinesis Agent Test

Test

Development

Remove tag

Remove tag

Remove tag

Remove tag

Cancel

Review and apply tag changes

Note

A resource can have a maximum of 50 user-applied tags. You might not be able to add new tags to a resource if you are approaching 50 user-applied tags. Typically, read-only system tags do not apply to the 50-tag limit. Tag keys must also be unique within your selected resources. You cannot add a new tag with a key that matches an existing tag key in your selected resources.

- When you are finished adding tags, choose **Review and apply changes**.
- If you accept the changes, choose **Apply changes to all selected**.

Apply tag changes to the selected resource

Review the following tag changes and then choose Apply.

▼ The following tags will be applied to the selected resource.

Tag key	Tag value
Team	Development

Cancel

Apply changes to all selected

- Depending on the number of resources you selected, applying new tags can take a few minutes. Do not leave the page or open a different page in the same browser tab. If changes were successful, a green success banner is displayed at the top of the page. Wait for a success or failure banner to appear on the page before you continue.

If tag changes to some or all resources were not successful, see [Troubleshooting Tag Changes](#) (p. 32). After you troubleshoot and resolve the root causes of unsuccessful tag changes (such as insufficient permissions), you can retry tag changes on resources for which tag changes failed. For more information, see [the section called “Retry failed tag changes”](#) (p. 32).

Edit tags of selected resources

You can use Tag Editor to change existing tag values on selected resources that are in the results of your [Find resources to tag](#) (p. 22) query. Editing a tag changes the tag's value on all selected resources that have the same tag key. You cannot edit a tag key, but you can delete a tag and create a new tag to replace a tag key. This deletes all tags with that key on selected resources.

Important

Do not store personally identifiable information (PII) or other confidential or sensitive information in tags. We use tags to provide you with billing and administration services. Tags are not intended to be used for private or sensitive data.

- In the results of your **Find resources to tag** query, select the check boxes next to the resources for which you want to change existing tags. Enter a text string in **Filter resources** to filter for part of a resource's name or ID. In the **Tags** column, note that resources in the results already have tags applied to them. In the following example, the first selected EC2 instance already has two tags.

Name	Service	Type	Region	ID	Tag Name	Total tags
<input checked="" type="checkbox"/> EC2 Instance i-0...	EC2	Instance	us-east-1	i-0...	test-ubuntu-ps	2
<input checked="" type="checkbox"/> EC2 Instance i-0...	EC2	Instance	us-east-1	i-0...	java-ec2-web-WebApp	1
<input checked="" type="checkbox"/> S3 Bucket codestar-us-east-1...	S3	Bucket	us-east-1	codestar-us-east-1...	java-ec2-web-S3Bucket	1
<input type="checkbox"/> S3 Bucket codestar-us-east-1...	S3	Bucket	us-east-1	codestar-us-east-1...	nodewebapp-WebsiteS3Bucket	1
<input type="checkbox"/> S3 Bucket codestar-us-east-1...	S3	Bucket	us-east-1	codestar-us-east-1...	S3Bucket	1
<input type="checkbox"/> EC2 Instance i-0...	EC2	Instance	us-east-1	i-0...	feb-node-ec2-WebApp	1
<input type="checkbox"/> S3 Bucket codestrapline-consolehookup-us-east-1...	S3	Bucket	us-east-1	codestrapline-consolehookup-us-east-1...	consolehookup-S3Bucket	1
<input checked="" type="checkbox"/> S3 Bucket cloudtrail-test-2018	S3	Bucket	us-west-2	cloudtrail-test-2018	-	1

- Choose **Manage tags of the selected resources**.
- On the **Manage tags** page, in **Edit tags of selected resources**, view the tags on the resource that you selected. Although your original query returned more resources, note that you are changing tags only on the resources that you selected in step 1.

Edit tags of selected resources
You can override the tags of all selected resources, or add new tags to them.

Tag key	Tag value - optional	Action
Name	Linux	Remove tag
Purpose	Kinesis Agent Test	Remove tag
Stage	Test	Remove tag
Team	Development	Remove tag

- Change, add, or delete tag values. Existing tags must have a tag key, but tag values are optional. In this walkthrough, we change the value of the **Team** tag to **QA**.

Edit tags of selected resources
You can override the tags of all selected resources, or add new tags to them.

Tag key	Tag value - optional	
Name	Linux	Remove tag
Purpose	Kinesis Agent Test	Remove tag
Stage	Test	Remove tag
Team	QA	Remove tag

Add tag

Cancel Review and apply tag changes

Selected resources have different tag values is displayed in the tag value field if resources in your selection have different values for the same key. In this case, placing your cursor in the box opens a drop-down list of all available values for this tag key in your selected resources.

Tag value - optional

Q Selected resources have different tag values

Remove tag

acd-wp-ec2 (1 resource has this tag value)

aws-cloud9-dk-cloud9-env-us-east-1- (1 resource has this tag value)

Remove tag

DK-Instance-us-east-1 (1 resource has this tag value)

-test-ubuntu-ps (1 resource has this tag value)

Remove tag

SUSEhostname (1 resource has this tag value)

Jim (4 resources have this tag value)

(empty value) (1 resource has this tag value)

Cancel Review and apply tag changes

If resources in your selection have the tag value you want, the tag value is highlighted as you type it. For example, if resources in your selection already have the tag value **QA**, the value is highlighted as you type **Q**. The values in the drop-down list help keep tag values consistent across resources. The tag value is changed on all selected resources. In this example, the tag value is changed to **QA** for all selected resources that had a **Team** tag key. For selected resources that did not have the **Team** tag, the **Team** tag with the value **QA** is added.

5. When you are finished changing tags, choose **Review and apply changes**.
6. If you accept the changes, choose **Apply changes to all selected**.

Apply tag changes to the selected resource X

Review the following tag changes and then choose Apply.

▼ The following tags will be applied to the selected resource.

Tag key	Tag value
Team	QA

Cancel Apply changes to all selected

7. Depending on the number of resources you selected, editing tags can take a few minutes. Do not leave the page or open a different page in the same browser tab. If changes were successful, a green

success banner is displayed at the top of the page. Wait for a success or failure banner to appear on the page before you continue.

If tag changes to some or all resources were not successful, see [Troubleshooting Tag Changes \(p. 32\)](#). After you troubleshoot and resolve the root causes of unsuccessful tag changes (such as insufficient permissions), you can retry tag changes on resources for which tag changes failed. For more information, see [the section called "Retry failed tag changes" \(p. 32\)](#).

Remove tags from selected resources

You can use Tag Editor to remove tags from selected resources that are in the results of your [Find resources to tag \(p. 22\)](#) query. Removing a tag deletes the tag from all selected resources that have the tag. Because you cannot edit tag keys, you can remove tags and replace them with new tags if you need to edit a tag key. This deletes all tags with that key on selected resources.

1. In the results of your **Find resources to tag** query, select the check boxes next to the resources you want to remove tags from. Enter a text string in **Filter resources** to filter for part of a resource's name or ID.

Resource search results (4 selected of 8)
Choose up to 500 resources for which you want to edit tags.

Export 8 resources to CSV Manage tags of selected resources

Filter resources

Name	Service	Type	Region	ID	Tag Name	Total tags
<input checked="" type="checkbox"/> EC2 Instance i-O-3	EC2	Instance	us-east-1	i-O-3	-test-ubuntu-ps	2
<input checked="" type="checkbox"/> EC2 Instance i-O-3	EC2	Instance	us-east-1	i-O-3	-java-ec2-web-WebApp	1
<input checked="" type="checkbox"/> S3 Bucket -codestar-us-east-1-jm-java-ec2-web-pipe	S3	Bucket	us-east-1	-codestar-us-east-1-jm-java-ec2-web-pipe	-java-ec2-web-S3Bucket	1
<input type="checkbox"/> S3 Bucket -codestar-us-east-1-jm-nodewebapp-app	S3	Bucket	us-east-1	-codestar-us-east-1-jm-nodewebapp-app	-nodewebapp-WebsiteS3Bucket	1
<input type="checkbox"/> S3 Bucket -codestar-us-east-1-jm-mc-ca-pipe	S3	Bucket	us-east-1	-codestar-us-east-1-jm-mc-ca-pipe	-S3Bucket	1
<input type="checkbox"/> EC2 Instance i-O-c	EC2	Instance	us-east-1	i-O-c	-feb-node-ec2-WebApp	2
<input type="checkbox"/> S3 Bucket codepipeline-consolehookup-us-east-1	S3	Bucket	us-east-1	codepipeline-consolehookup-us-east-1	consolehookup-S3Bucket	1
<input checked="" type="checkbox"/> S3 Bucket -cloudtrail-test-2018	S3	Bucket	us-west-2	-cloudtrail-test-2018	-	1

2. Choose **Manage tags of the selected resources**.
3. On the **Manage tags** page, in **Edit tags of selected resources**, view the tags on the resources that you selected. Although your original query returned more resources, note that you are changing tags only on the resources that you selected in step 1.

Edit tags of selected resources
You can override the tags of all selected resources, or add new tags to them.

Tag key

Name Remove tag

Purpose Remove tag

Stage Remove tag

Team Remove tag

Add tag

Cancel Review and apply tag changes

4. Choose **Remove tag** next to any tags that you want to delete. In this walkthrough, we remove the **Team** tag.

Note

Choosing **Remove tag** removes a tag from all selected resources that have the tag. In the example shown, this removes the **Team** tag from all selected resources that currently have the **Team** tag, regardless of the tag's value.

Edit tags of selected resources
You can override the tags of all selected resources, or add new tags to them.

Tag key	Tag value - optional	
Name	Linux	Remove tag
Purpose	Kinesis Agent Test	Remove tag
Stage	Test	Remove tag
Team	QA	Remove tag

Add tag

Cancel Review and apply tag changes

5. Choose **Review and apply changes**.
6. On the confirmation page, choose **Apply changes to all selected**.
7. Depending on the number of resources you selected, removing tags can take a few minutes. Do not leave the page or open a different page in the same browser tab. If changes were successful, a green success banner is displayed at the top of the page. Wait for a success or failure banner to appear on the page before you continue.

If tag changes to some or all resources were not successful, see [Troubleshooting Tag Changes \(p. 32\)](#). After you troubleshoot and resolve the root causes of unsuccessful tag changes (such as insufficient permissions), you can retry tag changes on resources for which tag changes failed. For more information, see [the section called “Retry failed tag changes” \(p. 32\)](#).

Retry failed tag changes

If tag changes fail on at least one of your selected resources, Tag Editor displays a red banner at the bottom of the page. The banner shows error messages for each type of failure that occurred. For each error, the banner identifies the specific resources on which Tag Editor could not make tag changes. After you review and [troubleshoot the errors \(p. 32\)](#), choose **Retry failed tag changes on resources** to retry changes only on those resources on which tag changes failed.

Related information

- [AWS tagging strategies](#)
- [Using cost allocation tags](#)
- [Tag Editor \(p. 21\)](#)

Troubleshooting tag changes

The following checklist might be helpful if errors occur when you try to apply or change tags on selected resources in [Find resources to tag \(p. 22\)](#) query results.

- The resource might already have the maximum number of tags. Generally, resources can have a maximum of 50 user-applied tags. Read-only system tags do not usually count toward the 50-tag maximum. Other users might also be adding tags to the same resource at the same time, which could raise the resource's tags to the maximum.
- Some services allow a different character set (or restrict the character set that is allowed) for creating tags. If you have added or changed tags using special characters, review the tag requirements in the resource's service documentation to verify that those characters are allowed by the service.
- You might not have permissions to modify the tags for the resource. If you do not have permissions to view existing tags on a resource, you cannot make changes to the resource's tags.

- You might not have adequate permissions to change the resource. Changes to the resource's metadata might be restricted by another administrator.
- The resource might have been edited or deleted by another user or process. For example, if the resource was launched as part of the creation of an AWS CloudFormation stack, and the stack was deleted or is no longer in an active state, the resource might no longer be available.
- Tag changes might not be possible if a resource is offline or terminated, or if other updates (such as software upgrades) to the resource are in progress.
- Tag changes can fail if you did not allow the changes to finish before leaving the page. Let tag changes finish, and wait for the success or failure banner to appear on the page, before you leave the page.

Related information

- [AWS tagging strategies](#)
- [Using cost allocation tags](#)
- [Tag Editor \(p. 21\)](#)

Using tags in IAM permission policies

[AWS Identity and Access Management \(IAM\)](#) is the AWS service that you use to create and manage permission policies that determine who can access your AWS resources. Every attempt to access an AWS service or read or write an AWS resource is access controlled by such an IAM policy.

These policies allow you to provide granular access to your resources. One of the features you can use to fine tune this access is the `Condition` element of the policy. This element lets you specify the additional conditions that must match the request to determine if the request can proceed. Among the things you can check with the `Condition` element are the tags that are attached to the user or role making the request and the tags attached to the resource that is the object of the request.

Tag-related condition keys

These are the condition keys that you can use in an IAM permission policy to control access based on tags. These condition keys let you compare the tags on the principal calling the operation, the tags provided to operation as a parameter, and the tags attached to the resource that would be access by the operation.

See the page linked by the **Condition key name** for complete details about that condition key and how to use it.

Condition key name	Description
aws:PrincipalTag	Compares the tag attached to the principal (IAM user or role) making the request with the tag that you specify in the policy.
aws:RequestTag	Compares the tag key-value pair that was passed to the request as a parameter with the tag pair that you specify in the policy.
aws:ResourceTag	Compares the tag key-value pair that you specify in the policy with the key-value pair that is attached to the resource.
aws:TagKeys	Compares only the tag <i>keys</i> in a request with the keys that you specify in the policy.

Example IAM policies that use tags

Example Example 1: Force users to attach a specific tag when they create a resource

The following example IAM permission policy shows how to force the user who creates or modifies an IAM policy's tags to include a tag with the key `Owner` and the value set to the individual's AWS user name. If those conditions are not met, then the policy does not match and operation is not allowed.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "TagCustomerManagedPolicies",
      "Effect": "Allow",
      "Action": [
        "iam:CreatePolicy",
        "iam:TagPolicy"
      ],
      "Resource": "arn:aws:iam::123456789012:policy/*",
      "Condition": {
        "StringEquals": {
          "aws:RequestTag/Owner": "${aws:username}"
        }
      }
    }
  ]
}
```

Example Example 2: Use tags to limit access to a resource to its "owner"

The following example IAM permission policy lets the user stop a running Amazon EC2 instance only if the user is tagged as the owner of that instance.

This example is an example of "attribute-based access control". For more information and additional examples of using IAM policies to implement a tag-based access control strategy, see [Controlling access to AWS resources using tags](#) and [Controlling access to and for IAM users and roles using tags](#), both in the *IAM User Guide*. There is also a much more comprehensive tutorial that shows how to grant access to different projects and groups using multiple tags at [IAM tutorial: Define permissions to access AWS resources based on tags](#)

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "VisualEditor1",
      "Effect": "Allow",
      "Action": [
        "ec2:StopInstances"
      ],
      "Resource": [
        "arn:aws:iam::123456789012:instance/*"
      ],
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Owner": "${aws:username}"
        }
      }
    }
  ]
}
```

AWS Organizations tag policies

An *tag policy* is a type of policy that you create in AWS Organizations. You can use tag policies to help standardize tags across the resources in your organization's accounts. To use tag policies, AWS recommends that you follow the workflows described in [Getting started with tag policies](#) in the *AWS Organizations User Guide*. As mentioned on that page, the recommended workflows include finding and correcting noncompliant tags. To accomplish these tasks, you use the Resource Groups console.

Topics

- [Prerequisites and permissions](#) (p. 35)
- [Evaluating compliance for an account](#) (p. 37)
- [Evaluating organization-wide compliance](#) (p. 39)

Prerequisites and permissions

Before you can evaluate compliance with tag policies in AWS Resource Groups, you need to meet the requirements and set the necessary permissions.

Prerequisites for evaluating compliance with tag policies

Evaluating compliance with tag policies requires the following:

- You must first enable the feature in AWS Organizations and create and attach tag policies. For more information, see the following pages in the *AWS Organizations User Guide*:
 - [Prerequisites and permissions for managing tag policies](#)
 - [Enabling tag policies](#)
 - [Getting started with tag policies](#)
- To [find noncompliant tags on an account's resources](#) (p. 37), you need sign-in credentials for that account and the permissions listed in [Permissions for evaluating compliance for an account](#) (p. 35).
- To [evaluate organization-wide compliance](#) (p. 39), you need sign-in credentials for the organization's management account and the permissions listed in [Permissions for evaluating organization-wide compliance](#) (p. 36).

Permissions for evaluating compliance for an account

Finding noncompliant tags on an account's resources requires these permissions:

- `organizations:DescribeEffectivePolicy` – To get the contents of the effective tag policy for the account.
- `tag:GetResources` – To get a list of resources that do not comply with the attached tag policy.
- `tag:TagResources` – To add or update tags. You also need service-specific permissions to create tags. For example, to tag resources in Amazon EC2, you need permissions for `ec2:CreateTags`.
- `tag:UntagResources` – To remove a tag. You also need service-specific permissions to remove tags. For example, to untag resources in Amazon EC2, you need permissions for `ec2:DeleteTags`.

The following example IAM policy provides permissions for evaluating tag compliance for an account.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
```



```
        "Sid": "EvaluateAccountCompliance",
        "Effect": "Allow",
        "Action": [
            "organizations:DescribeEffectivePolicy",
            "tag:GetResources",
            "tag:TagResources",
            "tag:UntagResources"
        ],
        "Resource": "*"
    }
}
```

For more information on IAM policies and permissions, see the [IAM User Guide](#).

Permissions for evaluating organization-wide compliance

Evaluating organization-wide compliance with tag policies requires the following permissions:

- `organizations:DescribeEffectivePolicy` – To get the contents of the tag policy that's attached to the organization, OU, or account.
- `tag:GetComplianceSummary` – To get a summary of noncompliant resources in all accounts in the organization.
- `tag:StartReportCreation` – To export the results of the most recent compliance evaluation to a file. Organization-wide compliance is evaluated every 48 hours.
- `tag:DescribeReportCreation` – To check the status of report creation.

The following example IAM policy provides permissions for evaluating organization-wide compliance.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "EvaluateOrgCompliance",
      "Effect": "Allow",
      "Action": [
        "organizations:DescribeEffectivePolicy",
        "tag:GetComplianceSummary",
        "tag:StartReportCreation",
        "tag:DescribeReportCreation"
      ],
      "Resource": "*"
    }
  ]
}
```

For more information on IAM policies and permissions, see the [IAM User Guide](#).

Amazon S3 bucket policy for storing report

To create an organization-wide compliance report, you must grant access for the tag policies service principal to an Amazon S3 bucket in the US East (N. Virginia) Region for report storage. Attach the following bucket policy to the bucket, replacing the placeholders with your actual S3 bucket name, the ID number of the organization, and the account ID number of the organization's management account for the organization in which you're applying the policy.

```
{
  "Version": "2012-10-17",
  "Statement": [
```

```
{
  "Sid": "TagPolicyACL",
  "Effect": "Allow",
  "Principal": {
    "Service": [
      "tagpolicies.tag.amazonaws.com"
    ]
  },
  "Action": "s3:GetBucketAcl",
  "Resource": "arn:aws:s3:::<your-bucket-name>",
  "Condition": {
    "StringEquals": {
      "aws:SourceAccount": "<organization-management-account-id>",
      "aws:SourceArn": "arn:aws:tag:us-east-1:<organization-management-
account-id>:*"
    }
  }
},
{
  "Sid": "TagPolicyBucketDelivery",
  "Effect": "Allow",
  "Principal": {
    "Service": [
      "tagpolicies.tag.amazonaws.com"
    ]
  },
  "Action": [
    "s3:PutObject",
    "s3:PutObjectAcl"
  ],
  "Resource": "arn:aws:s3:::<your-bucket-name>/AwsTagPolicies/<your-organization-
id>/**",
  "Condition": {
    "StringEquals": {
      "aws:SourceAccount": "<organization-management-account-id>",
      "aws:SourceArn": "arn:aws:tag:us-east-1:<organization-management-
account-id>:*"
    }
  }
}
]
```

Evaluating compliance for an account

You can evaluate the compliance of an account in your organization with its effective tag policy.

Important

Untagged resources don't appear as noncompliant in results.

The *effective tag policy* specifies the tagging rules that apply to an account. It is the aggregation of any tag policies the account inherits, plus any tag policy directly attached to the account. When you attach a tag policy to the organization root, it applies to all accounts in your organization. When you attach a tag policy to an organizational unit (OU), it applies to all accounts and OUs that belong to the OU.

Note

If you haven't yet created tag policies, see [Getting started with tag policies](#) in the *AWS Organizations User Guide*.

To find noncompliant tags, you must have the following permissions:

- `organizations:DescribeEffectivePolicy`
- `tag:GetResources`

- `tag:TagResources`
- `tag:UntagResources`

To evaluate an account's compliance with its effective tag policy (console)

1. While signed in to the account whose compliance you want to check, open the [AWS Resource Groups console](#), and then in the navigation pane, choose **Tag policies**.


Direct link: [Tag Policies console](#)

2. The **Effective tag policy** section shows when the policy was last updated and the defined tag keys. You can expand a tag key to see information about its values, case treatment, and whether the values are enforced for specific resources types.

Note

If you're signed in to the management account, you need to choose an account to see its effective policy and view compliance information.

3. In the **Resources with noncompliant tags** section, specify which Region to search for noncompliant tags. Optionally, you can also search by resource type. Then choose **Search resources**.

Real-time results are shown in the **Search results** section. To change the number of results returned per page or the columns to display, choose the settings icon ()

4. In the search results, select a resource with noncompliant tags.
5. In the dialog box that lists the resource's tags, choose the hyperlink to open the AWS service where the resource was created. From that console, correct the noncompliant tag.

Tip

If you're not sure which tags are noncompliant, go to the **Effective tag policy** section for the account in the Resource Groups console. You can expand a tag key to view its tagging rules.

6. Repeat the process of finding and correcting tags until the account resources that you care about are compliant in each Region.

To find noncompliant tags (AWS CLI, AWS API)

Use the following commands and operations to find noncompliant tags:

- AWS CLI:
 - [aws resourcegroupstaggingapi get-resources](#)
 - [aws resourcegroupstaggingapi tag-resources](#)
 - [aws resourcegroupstaggingapi untag-resources](#)

For the complete procedure for using tag policies in the AWS CLI, see [Using tag policies in the AWS CLI](#) in the *AWS Organizations User Guide*.

- AWS API:
 - [GetResources](#)
 - [TagResources](#)
 - [UntagResources](#)

What to Do Next

AWS recommends that you repeat the process of finding and correcting compliance issues. Continue until the account's resources that you care about are compliant with the effective tag policy in each Region.

Finding and correcting noncompliant tags is an iterative process for multiple reasons, including the following:

- Your organization's use of tag policies can evolve over time.
- It takes time to effect change in your organization when creating resources.
- Compliance can change anytime a new resource is created or when new tags are assigned to a resource.
- An account's effective tag policy is updated whenever a tag policy is attached to or detached from it. The effective tag policy is also updated whenever changes occur to tag the policies the account inherits.

If you are signed in as the management account in the organization, you can also generate a report. This report shows information on all tagged resources in your organization's accounts. For information, see [Evaluating organization-wide compliance \(p. 39\)](#).

Evaluating organization-wide compliance

You can generate a report that lists all tagged resources in accounts across your organization and whether each resource is compliant with the effective tag policy.

Important

Untagged resources don't appear as noncompliant in results.

You can generate the report from your organization's management account in the `us-east-1` Region only, and it must have access to an Amazon S3 bucket in the US East (N. Virginia) Region. The bucket must have an attached bucket policy as shown in [Amazon S3 Bucket Policy for Storing Report \(p. 36\)](#).

To generate an organization-wide compliance report, you must have the following permissions:

- `organizations:DescribeEffectivePolicy`
- `tag:StartReportCreation`
- `tag:DescribeReportCreation`
- `tag:GetComplianceSummary`

To generate an organization-wide compliance report (console)

1. In the [AWS Resource Groups console](#) navigation pane, choose **Tag policies**.

Direct link: [Tag Policies console](#)

2. From the **Tag policies** pane, choose the **This organization root** tab.
3. Near the bottom of the page, choose **Generate report**.
4. On the **Generate report** screen, specify where to store the report.
5. Choose **Start exporting**.

When the report is complete, you can download it from the **Noncompliance report** section on the **Organization root** tab.

The following shows an example report excerpt:

#	A	B	C	D	E	F	G	H	I
1	AccountId	Region	ResourceType	ComplianceStatus	NoncompliantKeys	KeysWithNoncompliantValues	ResourceARN	Tags	Last
2	1111122223333	ap-southeast-1	rds:db	TRUE			arn:aws:rds:ap-southeast-1:1111122223333:db:database	{ Name=database }	201
3	44445556666	ap-southeast-1	ec2:route-table	TRUE			arn:aws:ec2:ap-southeast-1:44445556666:route-table/rtb-1a2s3d4f5g6h	{ Name=route-table }	201
4	123456789012	ap-southeast-2	ec2:snapshot	TRUE			arn:aws:ec2:ap-southeast-2:123456789012:snapshot/snap-1a2s3d4f5g6h	{ Name=snapshot }	201
5	77778889999	ap-southeast-2	ec2:image	TRUE			arn:aws:ec2:ap-southeast-2:77778889999:image/ami-1a2s3d4f5g6h	{ Name=image }	201
6	234567890123	us-west-1	ec2:image	TRUE			arn:aws:ec2:us-west-1:234567890123:image/ami-1a2s3d4f5g6h	{ Name=image }	201
7	111111111111	us-west-1	ec2:instance	FALSE		Name,CostCenter	arn:aws:ec2:us-west-1:111111111111:instance/i-1a2s3d4f5g6h	{ Name=instance2, CostCenter=00002 }	201
8	222222222222	us-west-2	ec2:security-group	TRUE			arn:aws:ec2:us-west-2:222222222222:security-group/sg-1a2s3d4f5g6h	{ Name=security-group, CostCenter=0001 }	201
9	333333333333	us-west-2	ec2:instance	TRUE			arn:aws:ec2:us-west-2:333333333333:instance/i-1a2s3d4f5g6h	{ Name=instance }	201
10	444444444444	us-east-1	ec2:instance	FALSE		CostCenter	arn:aws:ec2:us-east-1:444444444444:instance/i-1a2s3d4f5g6h	{ Name=instance, CostCenter=0002w }	201
11	555555555555	us-east-1	ec2:snapshot	TRUE			arn:aws:ec2:us-east-1:555555555555:snapshot/snap-1a2s3d4f5g6h	{ Name=instance }	201
12	666666666666	us-east-1	ec2:volume	FALSE	name	Name	arn:aws:ec2:us-east-1:666666666666:volume/vol-1a2s3d4f5g6h	{ name=volume, Name=volume }	201
13	777777777777	us-east-2	ec2:snapshot	TRUE			arn:aws:ec2:us-east-2:777777777777:snapshot/snap-1a2s3d4f5g6h	{ name=snapshot }	201
14	888888888888	us-east-2	elasticmapreduce:cluster	TRUE			arn:aws:elasticmapreduce:us-east-2:888888888888:cluster/j-1a2s3d4f5g6h	{ Name=cluster, CostCenter=0001 }	201
15	999999999999	us-east-2	rds:snapshot	FALSE		Name	arn:aws:rds:us-east-2:999999999999:snapshot:rds:1a2s3d4f5g6h	{ Name=snapsho }	201

Notes

Organization-wide compliance is evaluated every 48 hours. This results in the following:

- It can take up to 48 hours for changes to a tag policy or resources to be reflected in the organization-wide compliance report. For example, assume that you have a tag policy that defines a new standardized tag for a resource type. Resources of that type that don't have this tag are shown as compliant in the report for up to 48 hours.
- Although you can generate the report at any time, report results aren't updated until the next evaluation is complete.
- The **NoncompliantKeys** column lists tag keys on the resource that are noncompliant with the effective tag policy.
- The **KeysWithNonCompliantValues** column lists keys defined in the effective policy that are on the resource with either incorrect case treatment or noncompliant values.
- If you close an AWS account that was a member of the organization, it can continue to appear in the tag compliance report for up to 90 days.

To generate an organization-wide compliance report (AWS CLI, AWS API)

Use the following commands and operations to generate an organization-wide compliance report, check on its status, and view the report:

- AWS CLI:
 - [aws resourcegroupstaggingapi start-report-creation](#)
 - [aws resourcegroupstaggingapi describe-report-creation](#)
 - [aws resourcegroupstaggingapi get-compliance-summary](#)

For the complete procedure for using tag policies in the AWS CLI, see [Using tag policies in the AWS CLI](#) in the *AWS Organizations User Guide*.

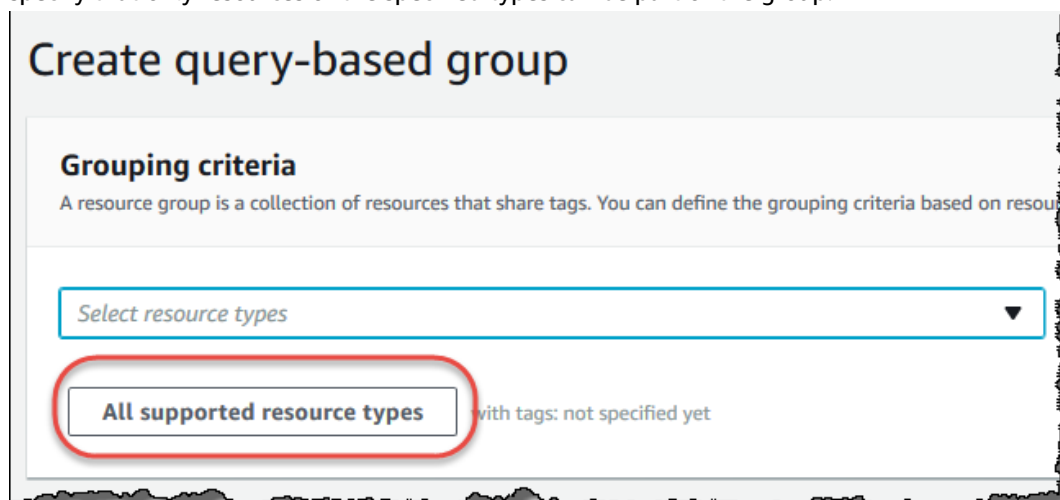
- AWS API:
 - [StartReportCreation](#)
 - [DescribeReportCreation](#)
 - [GetComplianceSummary](#)

Resources you can use with AWS Resource Groups and Tag Editor

You can use the AWS Management Console or the AWS CLI to create resource groups and then interact with the member resources through those groups. You can add tags to many AWS resources and then use those tags to manage group membership. This topic describes the AWS resources that you can include in resource groups by using AWS Resource Groups, and the AWS resources that you can tag by using Tag Editor.

Important

A resource group based on a query for **All supported resource types** can add members automatically over time, as new resources are supported by Resource Groups. When you run automations or other bulk tasks on an existing resource group based on **All supported resource types**, be aware that the actions might run on many more resources than were in the group when you first created the group. This might also mean that automations or tasks that you created for other resources are applied to possibly unintended resources, or resources on which the tasks cannot be successfully completed. In those cases, you can add a resource type filter to specify that only resources of the specified types can be part of the group.



The following tables list which resource types are supported for tagging in Tag Editor, for membership in tag query-based groups, and for membership in AWS CloudFormation stack-based groups.

Column definitions

- **Tag Editor Tagging** – You can tag resources of this type by using the [Tag Editor console](#). Otherwise, you must use either the [AWS Resource Groups Tagging API](#) or the tagging services supported natively by that resource's owning service.
- **Tag-based Groups** – You can include resources of this type in [resource groups whose membership is determined by the tags attached to the resources](#). The group specifies tag key names and values, and any resources with tags that match are automatically part of the group.
- **AWS CloudFormation Stack-based Groups** – You can include resources of this type in [resource groups whose membership consists of the resources created as part of a CloudFormation stack](#). The group specifies the stack's ARN, and all of its resources are automatically members of the group.

Note

Adding tags to a AWS CloudFormation stack causes an update of the stack.

For a list of resource types that are deprecated and no longer supported by Resource Groups, see the section [Deprecated resource types \(p. 66\)](#) at the end of this topic.

Amazon API Gateway

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::ApiGateway::Account	✗ No	✗ No	✓ Yes
AWS::ApiGateway::ApiKey	✗ No	✗ No	✓ Yes
AWS::ApiGateway::DomainName	✗ No	✗ No	✓ Yes
AWS::ApiGateway::RestApi	✗ No	✓ Yes	✓ Yes
AWS::ApiGateway::Stage	✗ No	✓ Yes	✗ No
AWS::ApiGateway::UsagePlan	✗ No	✗ No	✓ Yes

Amazon AppStream

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::AppStream::Fleet	✓ Yes	✓ Yes	✓ Yes
AWS::AppStream::ImageBuilder	✓ Yes	✓ Yes	✓ Yes
AWS::AppStream::Stack	✓ Yes	✓ Yes	✓ Yes

AWS AppSync

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::AppSync::DataSource	✗ No	✗ No	✓ Yes
AWS::AppSync::GraphQLApi	✗ No	✗ No	✓ Yes

Amazon Braket

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Braket::QuantumTask	✓ Yes	✓ Yes	✗ No

AWS Certificate Manager

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::CertificateManager::Certificate	✓ Yes	✓ Yes	✓ Yes

AWS Certificate Manager Private Certificate Authority

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::ACMPCA::CertificateAuthority	✗ No	✓ Yes	✗ No

AWS Cloud9

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Cloud9::Environment	✓ Yes	✓ Yes	✗ No

AWS CloudFormation

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::CloudFormation::Stack	✓ Yes	✓ Yes	✓ Yes

Amazon CloudFront

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::CloudFront::Distribution	✓ Yes ¹	✓ Yes ²	✓ Yes ²
AWS::CloudFront::StreamingDistribution	✓ Yes ¹	✓ Yes ²	✓ Yes ²

¹ This is a resource for a global service that is hosted in the **US East (N. Virginia)** Region. To use Tag Editor to create or modify tags for this resource type, you must include `us-east-1` from the **Select regions** list under **Find resources to tag** in the Tag Editor console.

² This is a resource for a global service that is hosted in the **US East (N. Virginia)** Region. Because Resource Groups are maintained separately for each region, you must switch your AWS Management Console to the AWS Region that contains the resources you want to include in the group. To create a resource group that contains a global resource, you must configure your AWS Management Console to **US East (N. Virginia) us-east-1** using the Region selector in the upper-right corner of the AWS Management Console.

AWS CloudTrail

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::CloudTrail::Trail	✓ Yes	✓ Yes	✓ Yes

Amazon CloudWatch

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::CloudWatch::Alarm	✓ Yes	✓ Yes	✓ Yes

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::CloudWatch::Dashboard	✗ No	✗ No	✓ Yes

Amazon CloudWatch Logs

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Logs::LogGroup	✗ No	✓ Yes	✓ Yes

AWS CodeArtifact

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::CodeArtifact::Domain	✓ Yes	✓ Yes	✓ Yes
AWS::CodeArtifact::Repository	✓ Yes	✓ Yes	✓ Yes

AWS CodeBuild

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::CodeBuild::Project	✓ Yes	✓ Yes	✗ No

AWS CodeCommit

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::CodeCommit::Repository	✓ Yes	✓ Yes	✗ No

AWS CodeDeploy

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::CodeDeploy::Application	✗ No	✗ No	✓ Yes
AWS::CodeDeploy::DeploymentConfig	✗ No	✗ No	✓ Yes

Amazon CodeGuru Reviewer

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::CodeGuruReviewer::RepositoryAssociation	✓ Yes	✓ Yes	✓ Yes

AWS CodePipeline

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::CodePipeline::CustomActionType	✗ No	✓ Yes	✗ No
AWS::CodePipeline::Pipeline	✓ Yes	✓ Yes	✓ Yes
AWS::CodePipeline::Webhook	✓ Yes	✓ Yes	✓ Yes

Amazon Cognito

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Cognito::IdentityPool	✓ Yes	✓ Yes	✓ Yes
AWS::Cognito::UserPool	✓ Yes	✓ Yes	✓ Yes

Amazon Comprehend

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Comprehend::DocumentClassifier	✓ Yes	✓ Yes	✗ No
AWS::Comprehend::EntityRecognizer	✓ Yes	✓ Yes	✗ No

AWS Config

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Config::ConfigRule	✓ Yes	✓ Yes	✗ No

AWS Data Exchange

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::DataExchange::DataSet	✓ Yes	✓ Yes	✗ No
AWS::DataExchange::Revision	✗ No	✓ Yes	✗ No

AWS Data Pipeline

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::DataPipeline::Pipeline	✓ Yes	✓ Yes	✓ Yes

AWS Database Migration Service

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::DMS::Certificate	✗ No	✓ Yes	✗ No
AWS::DMS::Endpoint	✗ No	✓ Yes	✓ Yes
AWS::DMS::EventSubscription	✗ No	✓ Yes	✗ No
AWS::DMS::ReplicationInstance	✗ No	✓ Yes	✓ Yes
AWS::DMS::ReplicationSubnetGroup	✗ No	✓ Yes	✗ No
AWS::DMS::ReplicationTask	✗ No	✓ Yes	✗ No

Amazon DynamoDB

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::DynamoDB::Table	✓ Yes	✓ Yes	✓ Yes

Amazon EMR

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::EMR::Cluster	✓ Yes	✓ Yes	✓ Yes

Amazon EMR Containers

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::EMRContainers::JobRun	✗ No	✓ Yes	✗ No
AWS::EMRContainers::VirtualCluster	✓ Yes	✓ Yes	✓ Yes

Amazon ElastiCache

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::ElastiCache::CacheCluster	✓ Yes	✓ Yes	✓ Yes
AWS::ElastiCache::Snapshot	✓ Yes	✓ Yes	✗ No

AWS Elastic Beanstalk

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::ElasticBeanstalk::Application	✓ Yes	✓ Yes	✗ No
AWS::ElasticBeanstalk::ApplicationVersion	✗ No	✓ Yes	✗ No
AWS::ElasticBeanstalk::ConfigurationTemplate	✗ No	✓ Yes	✗ No
AWS::ElasticBeanstalk::Environment	✗ No	✓ Yes	✗ No

Amazon Elastic Compute Cloud (Amazon EC2)

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::EC2::CapacityReservation	✗ No	✓ Yes	✗ No
AWS::EC2::CustomerGateway	✓ Yes	✓ Yes	✓ Yes
AWS::EC2::DHCPOptions	✓ Yes	✓ Yes	✓ Yes
AWS::EC2::EIP	✓ Yes	✓ Yes	✗ No
AWS::EC2::Host	✗ No	✓ Yes	✗ No
AWS::EC2::Image	✓ Yes	✓ Yes	✗ No
AWS::EC2::Instance	✓ Yes	✓ Yes	✓ Yes
AWS::EC2::InternetGateway	✓ Yes	✓ Yes	✓ Yes
AWS::EC2::LaunchTemplate	✗ No	✓ Yes	✓ Yes
AWS::EC2::NatGateway	✓ Yes	✓ Yes	✓ Yes

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::EC2::NetworkAcl	✓ Yes	✓ Yes	✓ Yes
AWS::EC2::NetworkInterface	✓ Yes	✓ Yes	✓ Yes
AWS::EC2::ReservedInstance	✓ Yes	✓ Yes	✗ No
AWS::EC2::RouteTable	✓ Yes	✓ Yes	✓ Yes
AWS::EC2::SecurityGroup	✓ Yes	✓ Yes	✓ Yes
AWS::EC2::Snapshot	✓ Yes	✓ Yes	✗ No
AWS::EC2::SpotInstanceRequest	✓ Yes	✓ Yes	✗ No
AWS::EC2::Subnet	✓ Yes	✓ Yes	✓ Yes
AWS::EC2::TransitGateway	✗ No	✓ Yes	✗ No
AWS::EC2::TransitGatewayRouteTable	✗ No	✓ Yes	✗ No
AWS::EC2::Volume	✓ Yes	✓ Yes	✓ Yes
AWS::EC2::VPC	✓ Yes	✓ Yes	✓ Yes
AWS::EC2::VPCPeeringConnection	✗ No	✓ Yes	✓ Yes
AWS::EC2::VPNConnection	✓ Yes	✓ Yes	✓ Yes
AWS::EC2::VPNGateway	✓ Yes	✓ Yes	✓ Yes

Amazon Elastic Container Registry

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::ECR::Repository	✗ No	✓ Yes	✗ No

Amazon Elastic Container Service

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::ECS::Cluster	✓ Yes	✓ Yes	✗ No
AWS::ECS::ContainerInstance	✗ No	✓ Yes	✗ No

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::ECS::Service	✗ No	✓ Yes	✗ No
AWS::ECS::Task	✗ No	✓ Yes	✗ No
AWS::ECS::TaskDefinition	✓ Yes	✓ Yes	✗ No

Amazon Elastic File System

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::EFS::FileSystem	✓ Yes	✓ Yes	✓ Yes

Amazon Elastic Inference

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::ElasticInference::ElasticInferenceAccelerator	✓ Yes	✓ Yes	✗ No

Amazon Elastic Kubernetes Service (Amazon EKS)

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::EKS::Cluster	✓ Yes	✓ Yes	✓ Yes

Elastic Load Balancing

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::ElasticLoadBalancing::LoadBalancer	✓ Yes	✓ Yes	✓ Yes

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::ElasticLoadBalancingV2::LoadBalancer	✓ Yes	✓ Yes	✓ Yes
AWS::ElasticLoadBalancingV2::TargetGroup	✓ Yes	✓ Yes	✓ Yes

Amazon OpenSearch Service

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Elasticsearch::Domain	✓ Yes	✓ Yes	✓ Yes

Amazon CloudWatch Events

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Events::Rule	✓ Yes	✓ Yes	✓ Yes

Amazon FSx

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::FSx::FileSystem	✓ Yes	✓ Yes	✗ No

Amazon Forecast

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Forecast::Dataset	✓ Yes	✓ Yes	✗ No

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Forecast::DatasetGroup	✓ Yes	✓ Yes	✗ No
AWS::Forecast::DatasetImportJob	✓ Yes	✓ Yes	✗ No
AWS::Forecast::Forecast	✓ Yes	✓ Yes	✗ No
AWS::Forecast::ForecastExportJob	✓ Yes	✓ Yes	✗ No
AWS::Forecast::Predictor	✓ Yes	✓ Yes	✗ No
AWS::Forecast::PredictorBacktestExportJob	✓ Yes	✓ Yes	✗ No

Amazon Fraud Detector

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::FraudDetector::Detector	✓ Yes	✓ Yes	✗ No
AWS::FraudDetector::DetectorVersion	✗ No	✓ Yes	✗ No
AWS::FraudDetector::EntityType	✓ Yes	✓ Yes	✗ No
AWS::FraudDetector::EventType	✓ Yes	✓ Yes	✗ No
AWS::FraudDetector::ExternalModel	✓ Yes	✓ Yes	✗ No
AWS::FraudDetector::Label	✓ Yes	✓ Yes	✗ No
AWS::FraudDetector::Model	✓ Yes	✓ Yes	✗ No
AWS::FraudDetector::ModelVersion	✗ No	✓ Yes	✗ No
AWS::FraudDetector::Outcome	✓ Yes	✓ Yes	✗ No
AWS::FraudDetector::Rule	✗ No	✓ Yes	✗ No
AWS::FraudDetector::Variable	✓ Yes	✓ Yes	✗ No

AWS Glue

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Glue::Crawler	✓ Yes	✓ Yes	✗ No

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Glue::Database	✗ No	✗ No	✓ Yes
AWS::Glue::Job	✓ Yes	✓ Yes	✗ No
AWS::Glue::Trigger	✓ Yes	✓ Yes	✗ No

AWS Glue DataBrew

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::DataBrew::Dataset	✓ Yes	✓ Yes	✓ Yes
AWS::DataBrew::Job	✓ Yes	✓ Yes	✓ Yes
AWS::DataBrew::Project	✓ Yes	✓ Yes	✓ Yes
AWS::DataBrew::Recipe	✓ Yes	✓ Yes	✓ Yes
AWS::DataBrew::Schedule	✓ Yes	✓ Yes	✓ Yes

AWS Identity and Access Management

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::IAM::InstanceProfile	✓ Yes	✓ Yes	✗ No
AWS::IAM::ManagedPolicy	✓ Yes	✓ Yes	✗ No
AWS::IAM::OpenIDConnectProvider	✓ Yes	✓ Yes	✗ No
AWS::IAM::Role	✗ No	✗ No	✓ Yes
AWS::IAM::SAMLProvider	✓ Yes	✓ Yes	✗ No
AWS::IAM::ServerCertificate	✓ Yes	✓ Yes	✗ No
AWS::IAM::VirtualMFADevice	✗ No	✓ Yes	✗ No

Amazon Inspector

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
<code>AWS::Inspector::AssessmentTemplate</code>	✗ No	✓ Yes	✓ Yes

AWS IoT

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
<code>AWS::IoT::TopicRule</code>	✗ No	✓ Yes	✓ Yes

AWS IoT Analytics

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
<code>AWS::IoTAnalytics::Channel</code>	✗ No	✓ Yes	✗ No
<code>AWS::IoTAnalytics::Dataset</code>	✓ Yes	✓ Yes	✗ No
<code>AWS::IoTAnalytics::Datastore</code>	✗ No	✓ Yes	✗ No
<code>AWS::IoTAnalytics::Pipeline</code>	✗ No	✓ Yes	✗ No

AWS IoT Events

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
<code>AWS::IoTEvents::DetectorModel</code>	✓ Yes	✓ Yes	✓ Yes
<code>AWS::IoTEvents::Input</code>	✓ Yes	✓ Yes	✓ Yes

AWS IoT Greengrass

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Greengrass::ConnectorDefinition	✓ Yes	✓ Yes	✗ No
AWS::Greengrass::CoreDefinition	✓ Yes	✓ Yes	✗ No
AWS::Greengrass::DeviceDefinition	✓ Yes	✓ Yes	✗ No
AWS::Greengrass::FunctionDefinition	✓ Yes	✓ Yes	✗ No
AWS::Greengrass::Group	✓ Yes	✓ Yes	✗ No
AWS::Greengrass::LoggerDefinition	✓ Yes	✓ Yes	✗ No
AWS::Greengrass::ResourceDefinition	✓ Yes	✓ Yes	✗ No
AWS::Greengrass::SubscriptionDefinition	✓ Yes	✓ Yes	✗ No

AWS Key Management Service

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::KMS::Alias	✗ No	✗ No	✓ Yes
AWS::KMS::Key	✓ Yes	✓ Yes	✓ Yes

Amazon Kinesis

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Kinesis::Stream	✓ Yes	✓ Yes	✓ Yes

Amazon Kinesis Data Analytics

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::KinesisAnalytics::Application	✓ Yes	✓ Yes	✓ Yes
AWS::KinesisAnalyticsV2::Application	✗ No	✗ No	✓ Yes

Amazon Kinesis Data Firehose

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::KinesisFirehose::DeliveryStream	✗ No	✓ Yes	✓ Yes

AWS Lambda

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Lambda::Alias	✗ No	✗ No	✓ Yes
AWS::Lambda::EventSourceMapping	✗ No	✗ No	✓ Yes
AWS::Lambda::Function	✓ Yes	✓ Yes	✓ Yes
AWS::Lambda::LayerVersion	✗ No	✗ No	✓ Yes
AWS::Lambda::Version	✗ No	✗ No	✓ Yes

Amazon MQ

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::AmazonMQ::Broker	✓ Yes	✓ Yes	✗ No
AWS::AmazonMQ::Configuration	✓ Yes	✓ Yes	✗ No

Amazon Macie

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Macie::ClassificationJob	✓ Yes	✓ Yes	✗ No
AWS::Macie::CustomDataIdentifier	✓ Yes	✓ Yes	✓ Yes
AWS::Macie::FindingsFilter	✓ Yes	✓ Yes	✓ Yes
AWS::Macie::Member	✓ Yes	✓ Yes	✗ No

Amazon Managed Streaming for Apache Kafka

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Kafka::Cluster	✓ Yes	✓ Yes	✗ No

Amazon OpenSearch Service OpenSearch

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::OpenSearchService::Domain	✓ Yes	✓ Yes	✓ Yes

AWS OpsWorks

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::OpsWorks::Instance	✗ No	✓ Yes	✓ Yes
AWS::OpsWorks::Layer	✗ No	✓ Yes	✓ Yes
AWS::OpsWorks::Stack	✗ No	✓ Yes	✓ Yes

AWS Organizations

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Organizations::Account	✓ Yes	✓ Yes	✗ No
AWS::Organizations::OrganizationalUnit	✗ No	✓ Yes	✗ No
AWS::Organizations::Policy	✗ No	✓ Yes	✗ No
AWS::Organizations::Root	✓ Yes	✓ Yes	✗ No

Amazon Pinpoint

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Pinpoint::App	✗ No	✓ Yes	✓ Yes
AWS::Pinpoint::EmailTemplate	✗ No	✓ Yes	✓ Yes
AWS::Pinpoint::PushTemplate	✗ No	✓ Yes	✓ Yes
AWS::Pinpoint::SmsTemplate	✗ No	✓ Yes	✓ Yes
AWS::Pinpoint::VoiceTemplate	✗ No	✓ Yes	✗ No

Amazon Quantum Ledger Database (Amazon QLDB)

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::QLDB::Ledger	✓ Yes	✓ Yes	✓ Yes
AWS::QLDB::Stream	✗ No	✓ Yes	✓ Yes

Amazon Redshift

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Redshift::Cluster	✓ Yes	✓ Yes	✓ Yes
AWS::Redshift::ClusterParameterGroup	✓ Yes	✓ Yes	✓ Yes
AWS::Redshift::ClusterSecurityGroup	✗ No	✓ Yes	✓ Yes
AWS::Redshift::ClusterSubnetGroup	✓ Yes	✓ Yes	✓ Yes
AWS::Redshift::HSMClientCertificate	✓ Yes	✓ Yes	✗ No

Amazon Relational Database Service (Amazon RDS)

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::RDS::DBCluster	✓ Yes	✓ Yes	✓ Yes
AWS::RDS::DBClusterParameterGroup	✓ Yes	✓ Yes	✓ Yes
AWS::RDS::DBClusterSnapshot	✓ Yes	✓ Yes	✗ No
AWS::RDS::DBInstance	✓ Yes	✓ Yes	✓ Yes
AWS::RDS::DBParameterGroup	✓ Yes	✓ Yes	✓ Yes
AWS::RDS::DBSecurityGroup	✓ Yes	✓ Yes	✓ Yes
AWS::RDS::DBSnapshot	✓ Yes	✓ Yes	✗ No
AWS::RDS::DBSubnetGroup	✓ Yes	✓ Yes	✓ Yes
AWS::RDS::EventSubscription	✓ Yes	✓ Yes	✗ No
AWS::RDS::OptionGroup	✓ Yes	✓ Yes	✗ No
AWS::RDS::ReservedDBInstance	✓ Yes	✓ Yes	✗ No

AWS Resource Access Manager

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::RAM::ResourceShare	✓ Yes	✓ Yes	✗ No

AWS Resource Groups

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::ResourceGroups::Group	✓ Yes	✓ Yes	✓ Yes

AWS Robomaker

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::RoboMaker::DeploymentJob	✗ No	✓ Yes	✗ No
AWS::RoboMaker::Fleet	✗ No	✓ Yes	✗ No
AWS::RoboMaker::Robot	✗ No	✓ Yes	✗ No
AWS::RoboMaker::RobotApplication	✓ Yes	✓ Yes	✗ No
AWS::RoboMaker::SimulationApplication	✓ Yes	✓ Yes	✗ No
AWS::RoboMaker::SimulationJob	✓ Yes	✓ Yes	✗ No

Amazon Route 53

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Route53::Domain	✓ Yes ¹	✓ Yes ²	✗ No
AWS::Route53::HealthCheck	✓ Yes ¹	✓ Yes ²	✓ Yes ²
AWS::Route53::HostedZone	✓ Yes ¹	✓ Yes ²	✓ Yes ²

¹ This is a resource for a global service that is hosted in the **US East (N. Virginia)** Region. To use Tag Editor to create or modify tags for this resource type, you must include `us-east-1` from the **Select regions** list under **Find resources to tag** in the Tag Editor console.

² This is a resource for a global service that is hosted in the **US East (N. Virginia)** Region. Because Resource Groups are maintained separately for each region, you must switch your AWS Management Console to the AWS Region that contains the resources you want to include in the group. To create a resource group that contains a global resource, you must configure your AWS Management Console to **US East (N. Virginia) us-east-1** using the Region selector in the upper-right corner of the AWS Management Console.

Amazon Route 53 Resolver

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
<code>AWS::Route53Resolver::ResolverEndpoint</code>	✓ Yes ¹	✓ Yes ²	✗ No
<code>AWS::Route53Resolver::ResolverRule</code>	✓ Yes ¹	✓ Yes ²	✗ No

¹ This is a resource for a global service that is hosted in the **US East (N. Virginia)** Region. To use Tag Editor to create or modify tags for this resource type, you must include `us-east-1` from the **Select regions** list under **Find resources to tag** in the Tag Editor console.

² This is a resource for a global service that is hosted in the **US East (N. Virginia)** Region. Because Resource Groups are maintained separately for each region, you must switch your AWS Management Console to the AWS Region that contains the resources you want to include in the group. To create a resource group that contains a global resource, you must configure your AWS Management Console to **US East (N. Virginia) us-east-1** using the Region selector in the upper-right corner of the AWS Management Console.

Amazon S3 Glacier

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
<code>AWS::Glacier::Vault</code>	✓ Yes	✓ Yes	✗ No

Amazon SageMaker

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
<code>AWS::SageMaker::Endpoint</code>	✗ No	✓ Yes	✓ Yes

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::SageMaker::EndpointConfig	✗ No	✓ Yes	✓ Yes
AWS::SageMaker::HyperParameterTuningJob	✗ No	✓ Yes	✗ No
AWS::SageMaker::LabelingJob	✗ No	✓ Yes	✗ No
AWS::SageMaker::Model	✗ No	✓ Yes	✓ Yes
AWS::SageMaker::NotebookInstance	✓ Yes	✓ Yes	✓ Yes
AWS::SageMaker::Pipeline	✗ No	✓ Yes	✗ No
AWS::SageMaker::TrainingJob	✗ No	✓ Yes	✗ No
AWS::SageMaker::TransformJob	✗ No	✓ Yes	✗ No
AWS::SageMaker::Workteam	✗ No	✓ Yes	✗ No

AWS Secrets Manager

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::SecretsManager::Secret	✓ Yes	✓ Yes	✓ Yes

AWS Service Catalog

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::ServiceCatalog::CloudFormationProduct	✗ No	✓ Yes	✓ Yes
AWS::ServiceCatalog::Portfolio	✗ No	✓ Yes	✓ Yes

Service Quotas

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::ServiceQuotas::Quota	✗ No	✓ Yes	✗ No

Amazon Simple Email Service

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::SES::ConfigurationSet	✓ Yes	✓ Yes	✓ Yes
AWS::SES::ContactList	✓ Yes	✓ Yes	✓ Yes
AWS::SES::DedicatedIpPool	✓ Yes	✓ Yes	✗ No
AWS::SES::Identity	✓ Yes	✓ Yes	✗ No

Amazon Simple Notification Service

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::SNS::Topic	✓ Yes	✓ Yes	✓ Yes

Amazon Simple Queue Service

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::SQS::Queue	✓ Yes	✓ Yes	✓ Yes

Amazon Simple Storage Service (Amazon S3)

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::S3::Bucket	✓ Yes	✓ Yes	✓ Yes

AWS Step Functions

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::StepFunctions::Activity	✓ Yes	✓ Yes	✓ Yes
AWS::StepFunctions::StateMachine	✓ Yes	✓ Yes	✓ Yes

Storage Gateway

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::StorageGateway::Gateway	✓ Yes	✓ Yes	✗ No
AWS::StorageGateway::Volume	✗ No	✓ Yes	✗ No

AWS Systems Manager

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::SSM::Document	✗ No	✓ Yes	✓ Yes
AWS::SSM::MaintenanceWindow	✗ No	✓ Yes	✗ No
AWS::SSM::ManagedInstance	✗ No	✓ Yes	✗ No
AWS::SSM::Parameter	✓ Yes	✓ Yes	✓ Yes
AWS::SSM::PatchBaseline	✗ No	✓ Yes	✓ Yes

Amazon WorkSpaces

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::WorkSpaces::Workspace	✓ Yes	✓ Yes	✓ Yes

Deprecated resource types

The following resource types are no longer supported for the specified functionality.

Service	Resource type	Support change	Date
AWS RoboMaker	AWS::RoboMaker::Robot	No longer supported by Tag Editor.	May 2, 2022
AWS RoboMaker	AWS::RoboMaker::Fleet	No longer supported by Tag Editor.	May 2, 2022
AWS RoboMaker	AWS::RoboMaker::Deployment	No longer supported by Tag Editor.	May 2, 2022

Security in AWS Resource Groups

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 Resource Groups, 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 Resource Groups. The following topics show you how to configure Resource Groups to meet your security and compliance objectives. You also learn how to use other AWS services that help you to monitor and secure your Resource Groups resources.

Topics

- [Data Protection in AWS Resource Groups](#) (p. 67)
- [Identity and Access Management for AWS Resource Groups](#) (p. 68)
- [Logging and Monitoring in Resource Groups](#) (p. 80)
- [Compliance Validation for Resource Groups](#) (p. 82)
- [Resilience in Resource Groups](#) (p. 83)
- [Infrastructure Security in Resource Groups](#) (p. 83)
- [Security Best Practices for Resource Groups](#) (p. 83)

Data Protection in AWS Resource Groups

The AWS [shared responsibility model](#) applies to data protection in AWS Resource Groups. 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. This content includes 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 user accounts with 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 recommend TLS 1.2 or later.
- 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 personal data that is stored in Amazon S3.
- If you require FIPS 140-2 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 \(FIPS\) 140-2](#).

We strongly recommend that you never put confidential or sensitive information, such as your customers' email addresses, into tags or free-form fields such as a **Name** field. This includes when you work with Resource Groups or other AWS services using the console, API, AWS CLI, or AWS SDKs. Any data that you enter into tags or free-form 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.

Data Encryption

Compared to other AWS services, AWS Resource Groups has a minimal attack surface, because it does not provide a way of changing, adding, or deleting AWS resources except for groups. Resource Groups collects the following service-specific information from you.

- Group names (not encrypted, not private)
- Group descriptions (not encrypted, but private)
- Member resources in groups (these are stored in logs, which are not encrypted)

Encryption at Rest

There are no additional ways of isolating service or network traffic specific to Resource Groups. If applicable, use AWS-specific isolation. You can use the Resource Groups API and console in a VPC to help maximize privacy and infrastructure security.

Encryption in Transit

AWS Resource Groups data is encrypted in transit to the service's internal database for backup. This is not user-configurable.

Key Management

AWS Resource Groups is not currently integrated with AWS Key Management Service and does not support AWS KMS keys.

Internetwork Traffic Privacy

AWS Resource Groups uses HTTPS for all transmissions between Resource Groups users and AWS. Resource Groups uses transport layer security (TLS) 1.2, but also supports TLS 1.0 and 1.1.

Identity and Access Management for AWS Resource Groups

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 Resource Groups resources. IAM is an AWS service that you can use with no additional charge.

Topics

- [Audience \(p. 69\)](#)
- [Authenticating With Identities \(p. 69\)](#)
- [Managing Access Using Policies \(p. 71\)](#)
- [How Resource Groups Works with IAM \(p. 73\)](#)
- [AWS Resource Groups Identity-Based Policy Examples \(p. 76\)](#)
- [Troubleshooting AWS Resource Groups Identity and Access \(p. 78\)](#)

Audience

How you use AWS Identity and Access Management (IAM) differs, depending on the work that you do in Resource Groups.

Service user – If you use the Resource Groups service to do your job, then your administrator provides you with the credentials and permissions that you need. As you use more Resource Groups features to do your work, you might need additional permissions. Understanding how access is managed can help you request the right permissions from your administrator. If you cannot access a feature in Resource Groups, see [Troubleshooting AWS Resource Groups Identity and Access \(p. 78\)](#).

Service administrator – If you're in charge of Resource Groups resources at your company, you probably have full access to Resource Groups. It's your job to determine which Resource Groups features and resources your employees should access. You must then submit requests to your IAM administrator to change the permissions of your service users. Review the information on this page to understand the basic concepts of IAM. To learn more about how your company can use IAM with Resource Groups, see [How Resource Groups Works with IAM \(p. 73\)](#).

IAM administrator – If you're an IAM administrator, you might want to learn details about how you can write policies to manage access to Resource Groups. To view example Resource Groups identity-based policies that you can use in IAM, see [AWS Resource Groups Identity-Based Policy Examples \(p. 76\)](#).

Authenticating With Identities

Authentication is how you sign in to AWS using your identity credentials. For more information about signing in using the AWS Management Console, see [Signing in to the AWS Management Console as an IAM user or root user](#) in the *IAM User Guide*.

You must be *authenticated* (signed in to AWS) as the AWS account root user, an IAM user, or by assuming an IAM role. You can also use your company's single sign-on authentication or even sign in using Google or Facebook. In these cases, your administrator previously set up identity federation using IAM roles. When you access AWS using credentials from another company, you are assuming a role indirectly.

To sign in directly to the [AWS Management Console](#), use your password with your root user email address or your IAM user name. You can access AWS programmatically using your root user or IAM users access keys. AWS provides SDK and command line tools to cryptographically sign your request using your credentials. If you don't use AWS tools, you must sign the request yourself. Do this using *Signature Version 4*, a protocol for authenticating inbound API requests. For more information about authenticating requests, see [Signature Version 4 signing process](#) in the *AWS General Reference*.

Regardless of the authentication method that you use, you might also be required to provide additional security information. For example, AWS recommends that you use multi-factor authentication (MFA) to increase the security of your account. To learn more, see [Using multi-factor authentication \(MFA\) in AWS](#) in the *IAM User Guide*.

AWS account root user

When you first create an AWS account, you begin with a single sign-in identity that has complete access to all AWS services and resources in the account. This identity is called the AWS account *root user* and is accessed by signing in with the email address and password that you used to create the account. We strongly recommend that you do not use the root user for your everyday tasks, even the administrative ones. Instead, adhere to the [best practice of using the root user only to create your first IAM user](#). Then securely lock away the root user credentials and use them to perform only a few account and service management tasks.

IAM Users and Groups

An *IAM user* is an identity within your AWS account that has specific permissions for a single person or application. An IAM user can have long-term credentials such as a user name and password or a set of access keys. To learn how to generate access keys, see [Managing access keys for IAM users](#) in the *IAM User Guide*. When you generate access keys for an IAM user, make sure you view and securely save the key pair. You cannot recover the secret access key in the future. Instead, you must generate a new access key pair.

An *IAM group* is an identity that specifies a collection of IAM users. You can't sign in as a group. You can use groups to specify permissions for multiple users at a time. Groups make permissions easier to manage for large sets of users. For example, you could have a group named *IAMAdmins* and give that group permissions to administer IAM resources.

Users are different from roles. A user is uniquely associated with one person or application, but a role is intended to be assumable by anyone who needs it. Users have permanent long-term credentials, but roles provide temporary credentials. To learn more, see [When to create an IAM user \(instead of a role\)](#) in the *IAM User Guide*.

IAM Roles

An *IAM role* is an identity within your AWS account that has specific permissions. It is similar to an IAM user, but is not associated with a specific person. You can temporarily assume an IAM role in the AWS Management Console by [switching roles](#). You can assume a role by calling an AWS CLI or AWS API operation or by using a custom URL. For more information about methods for using roles, see [Using IAM roles](#) in the *IAM User Guide*.

IAM roles with temporary credentials are useful in the following situations:

- **Temporary IAM user permissions** – An IAM user can assume an IAM role to temporarily take on different permissions for a specific task.
- **Federated user access** – Instead of creating an IAM user, you can use existing identities from AWS Directory Service, your enterprise user directory, or a web identity provider. These are known as *federated users*. AWS assigns a role to a federated user when access is requested through an [identity provider](#). For more information about federated users, see [Federated users and roles](#) in the *IAM User Guide*.
- **Cross-account access** – You can use an IAM role to allow someone (a trusted principal) in a different account to access resources in your account. Roles are the primary way to grant cross-account access. However, with some AWS services, you can attach a policy directly to a resource (instead of using a role as a proxy). To learn the difference between roles and resource-based policies for cross-account access, see [How IAM roles differ from resource-based policies](#) in the *IAM User Guide*.
- **Cross-service access** – Some AWS services use features in other AWS services. For example, when you make a call in a service, it's common for that service to run applications in Amazon EC2 or store objects in Amazon S3. A service might do this using the calling principal's permissions, using a service role, or using a service-linked role.

- **Principal permissions** – When you use an IAM user or role to perform actions in AWS, you are considered a principal. Policies grant permissions to a principal. When you use some services, you might perform an action that then triggers another action in a different service. In this case, you must have permissions to perform both actions. To see whether an action requires additional dependent actions in a policy, see [Actions, resources, and condition keys for AWS Resource Groups](#) in the *Service Authorization Reference*.
- **Service role** – A service role is an [IAM role](#) that a service assumes to perform actions on your behalf. An IAM administrator can create, modify, and delete a service role from within IAM. For more information, see [Creating a role to delegate permissions to an AWS service](#) in the *IAM User Guide*.
- **Service-linked role** – A service-linked role is a type of service role that is linked to an AWS service. The service can assume the role to perform an action on your behalf. Service-linked roles appear in your IAM account and are owned by the service. An IAM administrator can view, but not edit the permissions for service-linked roles.
- **Applications running on Amazon EC2** – You can use an IAM role to manage temporary credentials for applications that are running on an EC2 instance and making AWS CLI or AWS API requests. This is preferable to storing access keys within the EC2 instance. To assign an AWS role to an EC2 instance and make it available to all of its applications, you create an instance profile that is attached to the instance. An instance profile contains the role and enables programs that are running on the EC2 instance to get temporary credentials. For more information, see [Using an IAM role to grant permissions to applications running on Amazon EC2 instances](#) in the *IAM User Guide*.

To learn whether to use IAM roles or IAM users, see [When to create an IAM role \(instead of a user\)](#) in the *IAM User Guide*.

Managing Access Using Policies

You control access in AWS by creating policies and attaching them to IAM identities or AWS resources. A policy is an object in AWS that, when associated with an identity or resource, defines their permissions. You can sign in as the root user or an IAM user, or you can assume an IAM role. When you then make a request, AWS evaluates the related identity-based or resource-based policies. Permissions in the policies determine whether the request is allowed or denied. Most policies are stored in AWS as JSON documents. For more information about the structure and contents of JSON policy documents, see [Overview of JSON policies](#) in the *IAM User Guide*.

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**.

Every IAM entity (user or role) starts with no permissions. In other words, by default, users can do nothing, not even change their own password. To give a user permission to do something, an administrator must attach a permissions policy to a user. Or the administrator can add the user to a group that has the intended permissions. When an administrator gives permissions to a group, all users in that group are granted those permissions.

IAM policies define permissions for an action regardless of the method that you use to perform the operation. For example, suppose that you have a policy that allows the `iam:GetRole` action. A user with that policy can get role information from the AWS Management Console, the AWS CLI, or the AWS API.

Identity-Based Policies

Identity-based policies are JSON permissions policy documents that you can attach to an identity, such as an IAM user, group of users, or role. These policies control what actions users and roles can perform, on which resources, and under what conditions. To learn how to create an identity-based policy, see [Creating IAM policies](#) in the *IAM User Guide*.

Identity-based policies can be further categorized as *inline policies* or *managed policies*. Inline policies are embedded directly into a single user, group, or role. Managed policies are standalone policies that

you can attach to multiple users, groups, and roles in your AWS account. Managed policies include AWS managed policies and customer managed policies. To learn how to choose between a managed policy or an inline policy, see [Choosing between managed policies and inline policies](#) in the *IAM User Guide*.

Resource-Based Policies

Resource-based policies are JSON policy documents that you attach to a resource. Examples of resource-based policies are IAM *role trust policies* and Amazon S3 *bucket policies*. In services that support resource-based policies, service administrators can use them to control access to a specific resource. For the resource where the policy is attached, the policy defines what actions a specified principal can perform on that resource and under what conditions. You must [specify a principal](#) in a resource-based policy. Principals can include accounts, users, roles, federated users, or AWS services.

Resource-based policies are inline policies that are located in that service. You can't use AWS managed policies from IAM in a resource-based policy.

Access Control Lists (ACLs)

Access control lists (ACLs) control which principals (account members, users, or roles) have permissions to access a resource. ACLs are similar to resource-based policies, although they do not use the JSON policy document format.

Amazon S3, AWS WAF, and Amazon VPC are examples of services that support ACLs. To learn more about ACLs, see [Access control list \(ACL\) overview](#) in the *Amazon Simple Storage Service Developer Guide*.

Other Policy Types

AWS supports additional, less-common policy types. These policy types can set the maximum permissions granted to you by the more common policy types.

- **Permissions boundaries** – A permissions boundary is an advanced feature in which you set the maximum permissions that an identity-based policy can grant to an IAM entity (IAM user or role). You can set a permissions boundary for an entity. The resulting permissions are the intersection of entity's identity-based policies and its permissions boundaries. Resource-based policies that specify the user or role in the `Principal` field are not limited by the permissions boundary. An explicit deny in any of these policies overrides the allow. For more information about permissions boundaries, see [Permissions boundaries for IAM entities](#) in the *IAM User Guide*.
- **Service control policies (SCPs)** – SCPs are JSON policies that specify the maximum permissions for an organization or organizational unit (OU) in AWS Organizations. AWS Organizations is a service for grouping and centrally managing multiple AWS accounts that your business owns. If you enable all features in an organization, then you can apply service control policies (SCPs) to any or all of your accounts. The SCP limits permissions for entities in member accounts, including each AWS account root user. For more information about Organizations and SCPs, see [How SCPs work](#) in the *AWS Organizations User Guide*.
- **Session policies** – Session policies are advanced policies that you pass as a parameter when you programmatically create a temporary session for a role or federated user. The resulting session's permissions are the intersection of the user or role's identity-based policies and the session policies. Permissions can also come from a resource-based policy. An explicit deny in any of these policies overrides the allow. For more information, see [Session policies](#) in the *IAM User Guide*.

Multiple Policy Types

When multiple types of policies apply to a request, the resulting permissions are more complicated to understand. To learn how AWS determines whether to allow a request when multiple policy types are involved, see [Policy evaluation logic](#) in the *IAM User Guide*.

How Resource Groups Works with IAM

Before you use IAM to manage access to Resource Groups, you should understand what IAM features are available to use with Resource Groups. To get a high-level view of how Resource Groups and other AWS services work with IAM, see [AWS Services That Work with IAM](#) in the *IAM User Guide*.

Topics

- [Resource Groups Identity-Based Policies](#) (p. 73)
- [Resource-Based Policies](#) (p. 75)
- [Authorization Based on Resource Groups Tags](#) (p. 75)
- [Resource Groups IAM Roles](#) (p. 75)

Resource Groups Identity-Based Policies

With IAM identity-based policies, you can specify allowed or denied actions and resources as well as the conditions under which actions are allowed or denied. Resource Groups supports 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.

Policy actions in Resource Groups use the following prefix before the action: `resource-groups:`. Tag Editor actions are performed entirely in the console, but have the prefix `resource-explorer` in log entries.

For example, to grant someone permission to create a Resource Groups group with the Resource Groups `CreateGroup` API operation, you include the `resource-groups:CreateGroup` action in their policy. Policy statements must include either an **Action** or **NotAction** element. Resource Groups defines its own set of actions that describe tasks that you can perform with this service.

To specify multiple Resource Groups and Tag Editor actions in a single statement, separate them with commas as follows:

```
"Action": [
    "resource-groups:action1",
    "resource-groups:action2",
    "resource-explorer:action3"
```

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

```
"Action": "resource-groups:List*"
```

To see a list of Resource Groups actions, see [Actions, Resources, and Condition Keys for AWS Resource Groups](#) in the *IAM User Guide*.

Resources

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 Resource JSON policy element specifies the object or objects to which the action applies. Statements must include either a `Resource` or a `NotResource` element. As a best practice, specify a resource using its [Amazon Resource Name \(ARN\)](#). You can do this for actions that support a specific resource type, known as *resource-level permissions*.

For actions that don't support resource-level permissions, such as listing operations, use a wildcard (*) to indicate that the statement applies to all resources.

```
"Resource": "*" 
```

The only Resource Groups resource is a *group*. The group resource has an ARN in the following format:

```
arn:${Partition}:resource-groups:${Region}:${Account}:group/${GroupName}
```

For more information about the format of ARNs, see [Amazon Resource Names \(ARNs\) and AWS Service Namespaces](#).

For example, to specify the `my-test-group` resource group in your statement, use the following ARN:

```
"Resource": "arn:aws:resource-groups:us-west-2:123456789012:group/my-test-group" 
```

To specify all groups that belong to a specific account, use the wildcard (*):

```
"Resource": "arn:aws:resource-groups:us-west-2:123456789012:group/*" 
```

Some Resource Groups actions, such as those for creating resources, cannot be performed on a specific resource. In those cases, you must use the wildcard (*).

```
"Resource": "*" 
```

Some Resource Groups API actions can involve multiple resources. For example, `DeleteGroup` deletes groups, so an IAM user must have permissions to delete a specific group or all groups. To specify multiple resources in a single statement, separate the ARNs with commas.

```
"Resource": [
    "resource1",
    "resource2" ]
```

To see a list of Resource Groups resource types and their ARNs, and learn with which actions you can specify the ARN of each resource, see [Actions, Resources, and Condition Keys for AWS Resource Groups](#) in the *IAM User Guide*.

Condition Keys

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 `Condition` element (or *Condition block*) lets you specify conditions in which a statement is in effect. The `Condition` element is optional. You can create conditional expressions that use [condition operators](#), such as equals or less than, to match the condition in the policy with values in the request.

If you specify multiple `Condition` elements in a statement, or multiple keys in a single `Condition` element, AWS evaluates them using a logical `AND` operation. If you specify multiple values for a single condition key, AWS evaluates the condition using a logical `OR` operation. All of the conditions must be met before the statement's permissions are granted.

You can also use placeholder variables when you specify conditions. For example, you can grant an IAM user permission to access a resource only if it is tagged with their IAM user name. For more information, see [IAM policy elements: variables and tags](#) in the *IAM User Guide*.

AWS supports global condition keys and service-specific condition keys. To see all AWS global condition keys, see [AWS global condition context keys](#) in the *IAM User Guide*.

Resource Groups defines its own set of condition keys and also supports using some global condition keys. To see all AWS global condition keys, see [AWS Global Condition Context Keys](#) in the *IAM User Guide*.

To see a list of Resource Groups condition keys, and learn with which actions and resources you can use a condition key, see [Actions, Resources, and Condition Keys for AWS Resource Groups](#) in the *IAM User Guide*.

Examples

To view examples of Resource Groups identity-based policies, see [AWS Resource Groups Identity-Based Policy Examples](#) (p. 76).

Resource-Based Policies

Resource Groups does not support resource-based policies.

Authorization Based on Resource Groups Tags

You can attach tags to groups in Resource Groups, or pass tags in a request to Resource Groups. To control access based on tags, you provide tag information in the [condition element](#) of a policy using the `aws:ResourceTag/key-name`, `aws:RequestTag/key-name`, or `aws:TagKeys` condition keys. You can apply tags to a group when you are creating or updating the group. For more information about tagging a group in Resource Groups, see [Creating query-based groups in AWS Resource Groups](#) (p. 8) and [Updating groups in AWS Resource Groups](#) (p. 13) in this guide.

To view an example identity-based policy for limiting access to a resource based on the tags on that resource, see [Viewing Groups Based on Tags](#) (p. 78).

Resource Groups IAM Roles

An [IAM role](#) is an entity within your AWS account that has specific permissions. Resource Groups does not have or use service roles.

Using Temporary Credentials with Resource Groups

In Resource Groups, you can use temporary credentials to sign in with federation, 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](#).

Service-Linked Roles

[Service-linked roles](#) allow AWS services to access resources in other services to complete an action on your behalf.

Resource Groups does not have or use service-linked roles.

Service Roles

This feature allows a service to assume a [service role](#) on your behalf.

Resource Groups does not have or use service roles.

AWS Resource Groups Identity-Based Policy Examples

By default, IAM users and roles don't have permission to create or modify Resource Groups resources. They also can't perform tasks using the AWS Management Console, AWS CLI, or AWS API. An IAM administrator must create IAM policies that grant users and roles permission to perform specific API operations on the specified resources they need. The administrator must then attach those policies to the IAM users or groups that require those permissions.

To learn how to create an IAM identity-based policy using these example JSON policy documents, see [Creating Policies on the JSON Tab](#) in the *IAM User Guide*.

Topics

- [Policy Best Practices](#) (p. 76)
- [Using the Resource Groups Console and API](#) (p. 76)
- [Allow Users to View Their Own Permissions](#) (p. 77)
- [Viewing Groups Based on Tags](#) (p. 78)

Policy Best Practices

Identity-based policies are very powerful. They determine whether someone can create, access, or delete Resource Groups 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 using AWS managed policies** – To start using Resource Groups quickly, use AWS managed policies to give your employees the permissions they need. These policies are already available in your account and are maintained and updated by AWS. For more information, see [Get started using permissions with AWS managed policies](#) in the *IAM User Guide*.
- **Grant least privilege** – When you create custom policies, grant only the permissions required to perform a task. Start with a minimum set of permissions and grant additional permissions as necessary. Doing so is more secure than starting with permissions that are too lenient and then trying to tighten them later. For more information, see [Grant least privilege](#) in the *IAM User Guide*.
- **Enable MFA for sensitive operations** – For extra security, require IAM users to use multi-factor authentication (MFA) to access sensitive resources or API operations. For more information, see [Using multi-factor authentication \(MFA\) in AWS](#) in the *IAM User Guide*.
- **Use policy conditions for extra security** – To the extent that it's practical, define the conditions under which your identity-based policies allow access to a resource. For example, you can write conditions to specify a range of allowable IP addresses that a request must come from. You can also write conditions to allow requests only within a specified date or time range, or to require the use of SSL or MFA. For more information, see [IAM JSON policy elements: Condition](#) in the *IAM User Guide*.

Using the Resource Groups Console and API

To access the AWS Resource Groups and Tag Editor console and API, you must have a minimum set of permissions. These permissions must allow you to list and view details about the Resource Groups resources in your AWS account. If you create an identity-based policy that is more restrictive than the minimum required permissions, the console and API commands won't function as intended for entities (IAM users or roles) with that policy.

To ensure that those entities can still use Resource Groups, attach the following policy (or a policy that contains the permissions listed in the following policy) to the entities. For more information, see [Adding Permissions to a User](#) in the *IAM User Guide*:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "resource-groups:*",
        "cloudformation:DescribeStacks",
        "cloudformation:ListStackResources",
        "tag:GetResources",
        "tag:TagResources",
        "tag:UntagResources",
        "tag:getTagKeys",
        "tag:getTagValues",
        "resource-explorer:List*"
      ],
      "Resource": "*"
    }
  ]
}
```

For more information about granting access to Resource Groups and Tag Editor, see [Granting permissions for using AWS Resource Groups and Tag Editor \(p. 6\)](#) in this guide.

Allow Users to View Their Own Permissions

This example shows how you might create a policy that allows IAM users to view the inline and managed policies that are attached to their user identity. This policy includes permissions to complete this action on the console or programmatically using the AWS CLI or AWS API.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ViewOwnUserInfo",
      "Effect": "Allow",
      "Action": [
        "iam:GetUserPolicy",
        "iam:ListGroupForUser",
        "iam:ListAttachedUserPolicies",
        "iam:ListUserPolicies",
        "iam:GetUser"
      ],
      "Resource": ["arn:aws:iam::*:user/${aws:username}"]
    },
    {
      "Sid": "NavigateInConsole",
      "Effect": "Allow",
      "Action": [
        "iam:GetGroupPolicy",
        "iam:GetPolicyVersion",
        "iam:GetPolicy",
        "iam:ListAttachedGroupPolicies",
        "iam:ListGroupPolicies",
        "iam:ListPolicyVersions",
        "iam:ListPolicies",
        "iam:ListUsers"
      ],
      "Resource": "*"
    }
  ]
}
```

```
}
]
}
```

Viewing Groups Based on Tags

You can use conditions in your identity-based policy to control access to Resource Groups resources based on tags. This example shows how you might create a policy that allows viewing a group. However, permission is granted only if the group tag `Owner` has the value of that user's user name.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ListSpecificGroup",
      "Effect": "Allow",
      "Action": "resource-groups:ListGroup",
      "Resource": "arn:aws:resource-groups::region:account_ID:group/group_name"
    },
    {
      "Sid": "ViewGroupIfOwner",
      "Effect": "Allow",
      "Action": "resource-groups:ListGroup",
      "Resource": "arn:aws:resource-groups::region:account_ID:group/group_name",
      "Condition": {
        "StringEquals": {"aws:ResourceTag/Owner": "${aws:username}"}
      }
    }
  ]
}
```

You can attach this policy to the IAM users in your account. If a user named `richard-ro` attempts to view an Resource Groups group, the group must be tagged `Owner=richard-ro` or `owner=richard-ro`. Otherwise he is denied access. The condition tag key `Owner` matches both `Owner` and `owner` because condition key names are not case-sensitive. For more information, see [IAM JSON Policy Elements: Condition](#) in the *IAM User Guide*.

Troubleshooting AWS Resource Groups Identity and Access

Use the following information to help you diagnose and fix common issues that you might encounter when working with Resource Groups and IAM.

Topics

- [I Am Not Authorized to Perform an Action in Resource Groups \(p. 78\)](#)
- [I Am Not Authorized to Perform iam:PassRole \(p. 79\)](#)
- [I Want to View My Access Keys \(p. 79\)](#)
- [I'm an Administrator and Want to Allow Others to Access Resource Groups \(p. 79\)](#)
- [I Want to Allow People Outside of My AWS Account to Access My Resource Groups Groups \(p. 80\)](#)

I Am Not Authorized to Perform an Action in Resource Groups

If the AWS Management Console tells you that you're not authorized to perform an action, then you must contact your administrator for assistance. Your administrator is the person that provided you with your user name and password.

The following example error occurs when the `mateojackson` IAM user tries to use the console to view details about a group but does not have `resource-groups:ListGroup` permission.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform: resource-  
groups:ListGroup on resource: arn:aws:resource-groups::us-west-2:123456789012:group/my-  
test-group
```

In this case, Mateo asks his administrator to update his policies to allow him to access the `my-test-group` resource using the `resource-groups:ListGroup` action.

I Am Not Authorized to Perform `iam:PassRole`

If you receive an error that you're not authorized to perform the `iam:PassRole` action, then you must contact your administrator for assistance. Your administrator is the person that provided you with your user name and password. Ask that person to update your policies to allow you to pass a role to Resource Groups.

Some AWS services allow you to pass an existing role to that service, instead of creating a new service role or service-linked role. To do this, you must have permissions to pass the role to the service.

The following example error occurs when an IAM user named `marymajor` tries to use the console to perform an action in Resource Groups. However, the action requires the service to have permissions granted by a service role. Mary does not have permissions to pass the role to the service.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform: iam:PassRole
```

In this case, Mary asks her administrator to update her policies to allow her to perform the `iam:PassRole` action.

I Want to View My Access Keys

After you create your IAM user access keys, you can view your access key ID at any time. However, you can't view your secret access key again. If you lose your secret key, you must create a new access key pair.

Access keys consist of two parts: an access key ID (for example, `AKIAIOSFODNN7EXAMPLE`) and a secret access key (for example, `wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY`). Like a user name and password, you must use both the access key ID and secret access key together to authenticate your requests. Manage your access keys as securely as you do your user name and password.

Important

Do not provide your access keys to a third party, even to help [find your canonical user ID](#). By doing this, you might give someone permanent access to your account.

When you create an access key pair, you are prompted to save the access key ID and secret access key in a secure location. The secret access key is available only at the time you create it. If you lose your secret access key, you must add new access keys to your IAM user. You can have a maximum of two access keys. If you already have two, you must delete one key pair before creating a new one. To view instructions, see [Managing access keys](#) in the *IAM User Guide*.

I'm an Administrator and Want to Allow Others to Access Resource Groups

To allow others to access Resource Groups, you must create an IAM entity (user or role) for the person or application that needs access. They will use the credentials for that entity to access AWS. You must then attach a policy to the entity that grants them the correct permissions in Resource Groups.

To get started right away, see [Creating your first IAM delegated user and group](#) in the *IAM User Guide*.

For more information about granting access to Resource Groups and Tag Editor, see [Granting permissions for using AWS Resource Groups and Tag Editor \(p. 6\)](#) in this guide.

I Want to Allow People Outside of My AWS Account to Access My Resource Groups

You can create a role that users in other accounts or people outside of your organization can use to access your resources. You can specify who is trusted to assume the role. For services that support resource-based policies or access control lists (ACLs), you can use those policies to grant people access to your resources.

To learn more, consult the following:

- To learn whether Resource Groups supports these features, see [How Resource Groups Works with IAM \(p. 73\)](#).
- To learn how to provide access to your resources across AWS accounts that you own, see [Providing access to an IAM user in another AWS account that you own](#) in the *IAM User Guide*.
- To learn how to provide access to your resources to third-party AWS accounts, see [Providing access to AWS accounts owned by third parties](#) in the *IAM User Guide*.
- To learn how to provide access through identity federation, see [Providing access to externally authenticated users \(identity federation\)](#) in the *IAM User Guide*.
- To learn the difference between using roles and resource-based policies for cross-account access, see [How IAM roles differ from resource-based policies](#) in the *IAM User Guide*.

Logging and Monitoring in Resource Groups

All AWS Resource Groups actions are logged in AWS CloudTrail.

Logging AWS Resource Groups API Calls with AWS CloudTrail

AWS Resource Groups and Tag Editor are integrated with AWS CloudTrail, a service that provides a record of actions taken by a user, role, or an AWS service in Resource Groups or Tag Editor. CloudTrail captures all API calls for Resource Groups as events, including calls from the Resource Groups or Tag Editor console and from code calls to the Resource Groups APIs. If you create a trail, you can enable continuous delivery of CloudTrail events to an Amazon S3 bucket, including events for Resource Groups. If you don't configure a trail, you can still view the most recent events in the CloudTrail console in **Event history**. Using the information collected by CloudTrail, you can determine the request that was made to Resource Groups, the IP address from which the request was made, who made the request, when it was made, and additional details.

To learn more about CloudTrail, see the [AWS CloudTrail User Guide](#).

Resource Groups Information in CloudTrail

CloudTrail is enabled on your AWS account when you create the account. When activity occurs in Resource Groups, or in the Tag Editor console, that activity is recorded in a CloudTrail event along with other AWS service events in **Event history**. You can view, search, and download recent events in your AWS account. For more information, see [Viewing Events with CloudTrail Event History](#).

For an ongoing record of events in your AWS account, including events for Resource Groups, create a trail. A trail enables CloudTrail to deliver log files to an Amazon S3 bucket. By default, when you create a trail in the console, the trail applies to all regions. The trail logs events from all regions in the AWS

partition and delivers the log files to the Amazon S3 bucket that you specify. Additionally, you can configure other AWS services to further analyze and act upon the event data collected in CloudTrail logs. For more information, see:

- [Overview for Creating a Trail](#)
- [CloudTrail Supported Services and Integrations](#)
- [Configuring Amazon SNS Notifications for CloudTrail](#)
- [Receiving CloudTrail Log Files from Multiple Regions](#) and [Receiving CloudTrail Log Files from Multiple Accounts](#)

All Resource Groups actions are logged by CloudTrail and are documented in the [AWS Resource Groups API Reference](#). Resource Groups actions in CloudTrail are shown as events with the API endpoint `resource-groups.amazonaws.com` as their source. For example, calls to the `CreateGroup`, `GetGroup`, and `UpdateGroupQuery` actions generate entries in the CloudTrail log files. Tag Editor actions in the console are logged by CloudTrail, and are shown as events with the internal API endpoint `resource-explorer` as their source.

Every event or log entry contains information about who generated the request. The identity information helps you determine the following:

- Whether the request was made with root or IAM user credentials.
- Whether the request was made with temporary security credentials for a role or federated user.
- Whether the request was made by another AWS service.

For more information, see the [CloudTrail `userIdentity` Element](#).

Understanding Resource Groups Log File Entries

A trail is a configuration that enables delivery of events as log files to an Amazon S3 bucket that you specify. CloudTrail log files contain one or more log entries. An event represents a single request from any source and includes information about the requested action, the date and time of the action, request parameters, and so on. CloudTrail log files are not an ordered stack trace of the public API calls, so they do not appear in any specific order.

The following example shows a CloudTrail log entry that demonstrates the action `CreateGroup`.

```
{ "eventVersion": "1.05",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "ID number:AWSResourceGroupsUser",
    "arn": "arn:aws:sts::831000000000:assumed-role/Admin/AWSResourceGroupsUser",
    "accountId": "831000000000", "accessKeyId": "ID number",
    "sessionContext": {
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2018-06-05T22:03:47Z"
      },
      "sessionIssuer": {
        "type": "Role",
        "principalId": "ID number",
        "arn": "arn:aws:iam::831000000000:role/Admin",
        "accountId": "831000000000",
        "userName": "Admin"
      }
    }
  },
  "eventTime": "2018-06-05T22:18:23Z",
  "eventSource": "resource-groups.amazonaws.com",
```

```
"eventName": "CreateGroup",
"awsRegion": "us-west-2",
"sourceIPAddress": "100.25.190.51",
"userAgent": "console.amazonaws.com",
"requestParameters": {
  "Description": "EC2 instances that we are using for application staging.",
  "Name": "Staging",
  "ResourceQuery": {
    "Query": "string",
    "Type": "TAG_FILTERS_1_0"
  },
  "Tags": {
    "Key": "Phase",
    "Value": "Stage"
  }
},
"responseElements": {
  "Group": {
    "Description": "EC2 instances that we are using for application staging.",
    "groupArn": "arn:aws:resource-groups:us-west-2:831000000000:group/Staging",
    "Name": "Staging"
  },
  "resourceQuery": {
    "Query": "string",
    "Type": "TAG_FILTERS_1_0"
  }
},
"requestID": "de7z64z9-d394-12ug-8081-7zz0386fbc6",
"eventID": "8z7z18dz-6z90-47bz-87cf-e8346428zzz3",
"eventType": "AwsApiCall",
"recipientAccountId": "831000000000"
}
```

Compliance Validation for Resource Groups

Third-party auditors assess the security and compliance of AWS services as part of multiple AWS compliance programs, such as SOC, PCI, FedRAMP, and HIPAA.

To learn whether AWS Organizations or other AWS services are in scope of specific compliance programs, see [AWS Services in Scope by Compliance Program](#). 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 Whitepaper](#) – 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.

- [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 that helps you check your compliance with security industry standards and best practices.
- [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.

Resilience in Resource Groups

AWS Resource Groups performs automated backups to internal service resources. These backups are not user-configurable. Backups are encrypted, both at rest and in transit. Resource Groups stores customer data in Amazon DynamoDB.

The AWS global infrastructure is built around AWS Regions and Availability Zones. AWS Regions provide multiple physically separated and isolated Availability Zones, which are connected with low-latency, high-throughput, and highly redundant networking. With Availability Zones, you can design and operate applications and databases that automatically fail over between Availability Zones without interruption. Availability Zones are more highly available, fault tolerant, and scalable than traditional single or multiple data center infrastructures.

Even a complete loss of user resource groups would not result in a loss of customer data, because most customer data is replicated across AWS Availability Zones (AZs). If you delete groups accidentally, contact [AWS Support Center](#).

For more information about AWS Regions and Availability Zones, see [AWS Global Infrastructure](#).

Infrastructure Security in Resource Groups

There are no additional ways of isolating service or network traffic provided by Resource Groups. If applicable, use AWS-specific isolation. You can use the Resource Groups API and console in a VPC to help maximize privacy and infrastructure security.

As a managed service, AWS Resource Groups is protected by the AWS global network security procedures that are described in the [Amazon Web Services: Overview of Security Processes](#) whitepaper.

You use AWS published API calls to access Resource Groups through the network. Clients must support Transport Layer Security (TLS) 1.0 or later. We recommend TLS 1.2 or later, which Resource Groups supports. Clients must also support cipher suites with perfect forward secrecy (PFS) such as Ephemeral Diffie-Hellman (DHE) or Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). 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.

Resource Groups does not support resource-based policies.

Security Best Practices for Resource Groups

The following best practices are general guidelines and don't represent a complete security solution. Because these best practices might not be appropriate or sufficient for your environment, treat them as helpful considerations rather than prescriptions.

- **Use the principle of least privilege** to grant access to groups. Resource Groups supports resource-level permissions. Grant access to specific groups only as required for specific users. Avoid using asterisks in policy statements that assign permissions to all users or all groups. For more information about least privilege, see [Grant Least Privilege](#) in the *IAM User Guide*.
- **Keep private information out of public fields.** The name of a group is treated as service metadata. Group names are not encrypted. Do not put sensitive information in group names. Group descriptions are private.

Do not put private or sensitive information in tag keys or tag values.

- **Use authorization based on tagging** whenever appropriate. Resource Groups supports authorization based on tags. You can tag groups, then update policies that are attached to IAM users and security groups to set their level of access based on the tags that are applied to a group. For more information about how to use authorization based on tags, see [Controlling access to AWS resources using resource tags](#) in the *IAM User Guide*.

Many AWS services support authorization based on tags for their resources. Be aware that tag-based authorization might be configured for member resources in a group. If access to a group's resources is restricted by tags, unauthorized users or groups might not be able to perform actions or automations on those resources. For example, if an Amazon EC2 instance in one of your groups is tagged with a tag key of `Confidentiality` and a tag value of `High`, and you are not authorized to run commands on resources tagged `Confidentiality:High`, actions or automations that you perform on the EC2 instance will fail, even if actions are successful for other resources in the resource group. For more information about which services support tag-based authorization for their resources, see [AWS Services That Work with IAM](#) in the *IAM User Guide*.

For more information about developing a tagging strategy for your AWS resources, see [AWS Tagging Strategies](#).

AWS Resource Groups and Tagging Reference

Use the topics in this section to find reference information for various aspects of AWS Resource Groups.

Service quotas for Resource Groups

Name	Default	Adjust	Description
Resource groups per account	Each supported Region: 100	Yes	The maximum number of resource groups that you can create in this account. A resource group is a collection of AWS resources that match a specific criteria.

Note

You can request changes to quotas marked as adjustable by using the [AWS Resource Groups page in the Service Quotas console](#).

Service quotas for Tagging (Tag Editor and Resource Groups Tagging API)

Name	Default	
Tags attached per resource	50 user created tags (AWS created tags don't count against this limit)	
Tag key name	<p>Minimum of 1, maximum 128 Unicode characters in UTF-8.</p> <p>Allowed characters include: letters, numbers, spaces, and the following characters:</p> <p>_ . : / = + - @</p> <p>Key names can't begin with <code>aws :</code> because that is reserved.</p> <p>Note Some AWS services have some additional character or length</p>	

Name	Default	
	restrictions. For details, see the documentation for the specific service.	
Tag values	<p>Minimum of 0, maximum of 256 Unicode characters in UTF-8.</p> <p>Allowed characters include: letters, numbers, spaces, and the following characters:</p> <p>_ . : / = + - @</p> <p>Note Some AWS services have some additional character or length restrictions. For details, see the documentation for the specific service.</p>	
Rate of calling the GetResources API operation	Maximum of 15 calls per second	
Rate of calling the following API operations: <ul style="list-style-type: none">• TagResources• UntagResources• GetTagKeys• GetTagValues	Maximum of 5 calls per second	

Note

These limits are currently not adjustable using the [Service Quotas console](#). Contact AWS Support.

AWS managed policies available for use with AWS Resource Groups and Tag Editor

[AWS-managed IAM permission policies](#) enable you to grant pre-configured permissions to the IAM users and roles in your account. AWS managed policies are tested and adhere to best practice recommendations, so you can reliably use them in the scenarios for which they're define. As new resource types are supported as members of resource groups, and as new resource types support tagging, AWS automatically updates these policies to support them. You don't need to do anything.

The following table lists the AWS-managed IAM permission policies available for you to use to grant permissions to AWS Resource Groups.

Policy name and ARN	Description
AWSResourceGroupsReadOnlyAccess	Grants read-only access to the AWS Resource Groups management console. It includes permission to view the details of

Policy name and ARN	Description
AWSResourceGroupsReadOnlyAccess arn:aws:iam::aws:policy/ AWSResourceGroupsReadOnlyAccess	a resource, including the list of attached tags. This policy doesn't grant permission to make any changes to resource groups or tags.
ResourceGroupsandTagEditorReadOnlyAccess arn:aws:iam::aws:policy/ ResourceGroupsandTagEditorReadOnlyAccess	Grants read-only access to the AWS Resource Groups management console, including the Tag Editor. It includes permission to view the details of a resource, including its tags. You can use the Tag Editor to view resources that match tag queries. This policy doesn't grant permission to make any changes to resource groups or tags.
ResourceGroupsandTagEditorFullAccess arn:aws:iam::aws:policy/ ResourceGroupsandTagEditorFullAccess	Grants full administrative access to the AWS Resource Groups management console. It includes permissions to view, create, and modify resource groups. It also includes permissions to view, create, and modify tags for any resources that are supported by Tag Editor.

AWS Resource Groups document history

update-history-change	update-history-description	update-history-date
Deprecation of resource types	The following resource types are no supported by Tag Editor: AWS::RoboMaker::Robot, AWS::RoboMaker::Fleet, and AWS::RoboMaker::DeploymentJob. Those types are removed from the Supported resources page.	May 17, 2022
Resource groups can now be used by Amazon VPC Network Access Analyzer to monitor unwanted network traffic to your AWS resources.	You can use AWS Resource Groups to specify the sources and destinations for your network access requirements.	December 3, 2021
Added support for resources of AWS Resilience Hub (p. 88)	AWS Resource Groups now supports including resources for AWS Resilience Hub in a resource group.	November 18, 2021
Added support for resources of Amazon Pinpoint (p. 88)	AWS Resource Groups now supports including resources for Amazon Pinpoint in a resource group.	November 11, 2021
Added support for resource groups that are configured and managed by AWS Service Catalog AppRegistry	AWS Resource Groups now supports resource groups that contain service configurations for resources in applications that you create by using AWS Service Catalog AppRegistry. For more information, see Service Configurations in the <i>AWS Resource Groups API Reference</i> .	September 15, 2021
Added support for resources of Amazon OpenSearch Service (p. 88)	AWS Resource Groups now supports including resources for Amazon OpenSearch Service in a resource group.	August 11, 2021
Added support for resources of AWS Braket (p. 88)	AWS Resource Groups now supports including resources for AWS Braket in a resource group.	June 30, 2021
Added support for resources of Amazon EMR Containers (p. 88)	AWS Resource Groups now supports including resources for Amazon EMR containers in a resource group.	April 27, 2021

Added support for resources of additional AWS services (p. 88)	AWS Resource Groups now supports including resources for the following services in a resource group: Amazon CodeGuru Reviewer, Amazon Elastic Inference, Amazon Forecast, Amazon Fraud Detector, and Service Quotas.	February 25, 2021
Added chapter on security and compliance. (p. 88)	Discusses how Resource Groups protects your information and complies with regulatory standards.	July 30, 2020
Added support for resource groups that are configured for AWS services (p. 88)	You can now create resource groups that are associated with an AWS service and that configure how the service can interact with the resources that are in the group. In this first release of the feature, you can create a resource group that contains Amazon EC2 capacity reservations and then launch Amazon EC2 instances into the group. If there's capacity in one or more of the group's reservations that match your instance, then that instance uses the reservation. If the instance doesn't match any available reservations in the group, then it launches as an on-demand instance. For more information, see Working with capacity reservation groups in the <i>Amazon EC2 User Guide for Linux Instances</i> .	July 29, 2020
Added support for AWS IoT Greengrass resources. (p. 88)	More resource types are now supported by AWS Resource Groups and Tag Editor.	March 25, 2020

View operations data for AWS Resource Groups (p. 88)	In the AWS Systems Manager console, the AWS Resource Groups page displays operations data for a selected group on four tabs: Details , Config , CloudTrail , OpsItems . These tabs are not available when viewing a group in the Resource Groups console. You can use the information on these tabs to help you understand which resources in a group are compliant and working correctly and which resources require action. If you need to take action on a resource, you can use Systems Manager Automation runbooks to perform common operations maintenance and troubleshooting tasks. For more information, see Viewing operations data for AWS Resource Groups in the <i>AWS Systems Manager User Guide</i> .	March 16, 2020
Check for compliance with tag policies (p. 88)	After you create and attach tag policies to accounts using AWS Organizations, you can find noncompliant tags on resources in your organization's accounts.	November 26, 2019
Support for more resource types (p. 88)	More resource types are now supported by AWS Resource Groups and Tag Editor.	October 4, 2019
New resource types supported by AWS Resource Groups (p. 88)	More resource types are now supported by AWS Resource Groups, especially for groups based on an AWS CloudFormation stack.	August 5, 2019
New resource types supported by AWS Resource Groups (p. 88)	Amazon API Gateway REST APIs, Amazon CloudWatch Events events, and Amazon SNS topics are now supported resource types in AWS Resource Groups.	June 27, 2019
Tag Editor now supports finding untagged resources (p. 88)	You can now search for resources in Tag Editor that do not have tag values applied for a specific tag key.	June 18, 2019
New resource types supported by AWS Resource Groups and Tag Editor (p. 88)	Over 50 new resource types have been added to AWS Resource Groups and Tag Editor support.	June 6, 2019

AWS Resource Groups and Tag Editor console moves out of AWS Systems Manager console (p. 88)	The AWS Resource Groups and Tag Editor console is now independent from the Systems Manager console. Although you can still find pointers to the AWS Resource Groups console in the Systems Manager left navigation bar, you can open the Resource Groups and Tag Editor console directly from the drop-down menu at the upper left of the AWS Management Console.	June 5, 2019
New Resource Groups authorization and access control features (p. 88)	Resource Groups now supports action-based policies, resource-level permissions, and authorization based on tags.	May 24, 2019
Older, legacy Resource Groups and Tag Editor tools are no longer available (p. 88)	Mentions of older, classic, or legacy Resource Groups and Tag Editor have been removed; these tools are no longer available in AWS. Use AWS Resource Groups and Tag Editor instead.	May 14, 2019
Tag Editor now supports tagging resources across multiple regions (p. 88)	Tag Editor now lets you search for and manage tags of resources across multiple regions, with your current region added to resource queries by default.	May 2, 2019
Tag Editor now supports exporting query results to a CSV (p. 88)	You can export the results of a query on the Find Resources to tag page to a CSV-formatted file. A new Region column is shown in Tag Editor query results. Tag Editor now lets you search for resources that have empty values for a specific tag key. Tag key values auto-complete as you type a unique value among existing keys.	April 2, 2019
Tag Editor now supports adding all resource types to a query (p. 88)	You can apply tags to up to 20 individual resource types in a single operation, or you can choose All resource types to query all resource types in a region. Autocompletion has been added to the Tag key field of a query to help enable consistent tag keys among resources. If tag changes fail on some resources, you can retry tag changes on just resources for which tag changes failed.	March 19, 2019

Tag Editor now supports multiple resource types in a search (p. 88)	You can apply tags to up to 20 resource types in a single operation. You can also choose the columns that are shown to you in search results, including columns for each unique tag key found in your search results or selected resources from results.	February 26, 2019
Documentation added for new Tag Editor (p. 88)	The "Working with Tag Editor" section describes how to use the new AWS Tag Editor console experience.	February 13, 2019
New resource types supported for groups in Resource Groups (p. 88)	Added new resource types that are now supported in Resource Groups.	February 4, 2019
Improved user experience for adding tags to tag-based Resource Groups queries (p. 88)	Minor changes to the console user experience for addition of tags in a tag-based query.	December 17, 2018
AWS CloudFormation stack-based query support added to Resource Groups (p. 88)	You can create resource groups where the query is based on an AWS CloudFormation stack. After you choose a stack, you can choose which resource types from the stack you want to appear in your group's query.	November 13, 2018
Resource Groups and CloudTrail (p. 88)	Resource Groups now offers AWS CloudTrail support. You can view and work with logs of all Resource Groups API calls in CloudTrail.	June 29, 2018

- **API version:** 2017-11-27
- **Latest documentation update:** September 24, 2019

Earlier updates

The following table describes important changes in each release of the *AWS Resource Groups User Guide* before June 2018.

Change	Description	Date
Initial release	Initial release of the next generation of AWS Resource Groups	November 29, 2017

AWS glossary

For the latest AWS terminology, see the [AWS glossary](#) in the *AWS General Reference*.