User Guide

AWS Resource Groups

Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

AWS Resource Groups: User Guide

Copyright © 2025 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

What are resource groups? 1
Resources and their group types 1
Use cases for resource groups 3
AWS Resource Groups and permissions 3
AWS Resource Groups resources 4
How tagging works 4
Getting started 5
Prerequisites 5
Resource Groups authorization and access control 11
AWS services that work with AWS Resource Groups 12
Service configurations 16
Accessing 16
Syntax & structure 17
Configuration types and parameters 18
Creating groups 37
Types of resource group queries 37
Build a tag-based query and create a group 41
Create an AWS CloudFormation stack-based group 44
Updating groups 46
Update tag-based query groups 46
Update an AWS CloudFormation stack-based group 49
Monitoring resource groups for changes 52
Turning on group lifecycle events 54
Creating a group lifecycle events rule 56
Creating a rule to capture only specific group lifecycle event types 59
Turning off group lifecycle events 59
Structure and syntax of events 61
Structure of the `detail` field 63
Example custom event patterns 70
Deleting groups 74
Supported resource types 75
AWS DeepComposer 76
Amazon API Gateway 77
Amazon API Gateway V2 77

IAM Access Analyzer 78
AWS Amplify 78
AWS App Runner 78
AWS AppConfig 79
AWS AppFabric 80
Amazon AppFlow 80
AppIntegrations 80
AWS App Mesh 81
Amazon AppStream 82
AWS AppSync 82
Application Auto Scaling 83
AWS Application Migration Service 83
Artificial intelligence operations (AIOps) 84
Amazon Athena 84
AWS Audit Manager 84
AWS B2B Data Interchange 85
AWS Backup 86
AWS Backup gateway 86
AWS Backup search 87
AWS Batch 87
Amazon Bedrock 88
AWS Billing Conductor 89
AWS Billing and Cost Management 89
Amazon Braket 90
AWS Budgets 90
AWS BugBust 90
AWS Certificate Manager 91
AWS Certificate Manager Private Certificate Authority 91
Amazon Q Developer in chat applications 91
Amazon Chime 92
AWS Clean Rooms 93
AWS Clean Rooms ML 93
Amazon Cloud Directory 94
AWS Cloud9 94
AWS CloudFormation 95
Amazon CloudFront 95

AWS CloudHSM ... 96
AWS Cloud Map ... 96
Amazon CloudSearch ... 97
AWS CloudTrail ... 97
Amazon CloudWatch ... 97
Amazon CloudWatch Application Insights ... 98
CloudWatch Application Signals ... 99
CloudWatch Evidently ... 99
Amazon CloudWatch Logs ... 100
Amazon CloudWatch Observability Manager ... 100
Amazon CloudWatch RUM ... 101
Amazon CloudWatch Synthetics ... 101
AWS CodeArtifact ... 101
AWS CodeBuild ... 102
Amazon CodeCatalyst ... 102
AWS CodeCommit ... 103
AWS CodeConnections ... 103
AWS CodeDeploy ... 103
Amazon CodeGuru Reviewer ... 104
Amazon CodeGuru Profiler ... 104
AWS CodePipeline ... 105
AWS CodeStar Notifications ... 105
AWS CodeConnections ... 105
Amazon CodeWhisperer ... 106
Amazon Cognito ... 106
Amazon Comprehend ... 107
AWS Config ... 108
Amazon Connect ... 109
Amazon Connect Cases ... 110
Amazon Connect Customer Profiles ... 111
Amazon Connect Outbound Campaigns ... 111
Amazon Connect Voice ID ... 111
Amazon Connect Wisdom ... 112
AWS Control Tower ... 113
AWS Cost Explorer ... 113
AWS Cost and Usage Report ... 114

AWS Data Exchange ... 114
AWS Data Exports ... 114
Amazon Data Lifecycle Manager ... 115
AWS Data Pipeline ... 115
AWS DataSync ... 115
Amazon DataZone ... 116
AWS Database Migration Service ... 116
AWS Deadline Cloud ... 117
Amazon Detective ... 118
AWS Device Farm ... 118
AWS Diode Messaging ... 119
AWS Diode Object Transfer ... 119
AWS Direct Connect ... 119
AWS Directory Service ... 120
Amazon DocumentDB Elastic Clusters ... 120
Amazon DynamoDB ... 121
DynamoDB Accelerator ... 121
Amazon EMR ... 121
Amazon EMR Containers ... 122
Amazon EMR Serverless ... 122
Amazon ElastiCache ... 123
AWS Elastic Beanstalk ... 124
Amazon Elastic Compute Cloud (Amazon EC2) ... 124
Amazon Elastic Container Registry ... 129
Amazon Elastic Container Service ... 130
AWS Elastic Disaster Recovery ... 130
Amazon Elastic File System ... 131
Amazon Elastic Kubernetes Service (Amazon EKS) ... 131
Elastic Load Balancing ... 132
Amazon OpenSearch Service ... 133
AWS Elemental MediaLive ... 133
AWS Elemental MediaConvert ... 135
AWS Elemental MediaPackage V2 ... 135
AWS Elemental MediaStore ... 136
MediaTailor ... 136
AWS Elemental Support Cases ... 137

AWS End User Messaging Social ... 137
AWS Entity Resolution ... 137
Amazon CloudWatch Events ... 138
Amazon EventBridge Pipes ... 139
Amazon EventBridge Scheduler ... 139
Amazon EventBridge Schemas ... 139
Amazon FSx ... 140
AWS Fault Injection Service ... 140
Amazon FinSpace schemas ... 141
AWS Firewall Manager ... 141
AWS IoT Fleet Hub ... 142
Amazon Forecast ... 142
Amazon Fraud Detector ... 143
FreeRTOS ... 144
Amazon GameLift Servers ... 145
AWS Global Accelerator ... 146
AWS Glue ... 146
AWS Glue DataBrew ... 147
AWS Ground Station ... 148
Amazon GuardDuty ... 149
AWS HealthImaging ... 149
AWS HealthLake ... 150
AWS HealthOmics ... 150
Amazon Interactive Video Service ... 151
IAM ... 152
AWS Identity and Access Management ... 152
EC2 Image Builder ... 153
Amazon Inspector ... 154
Internet Monitor ... 154
AWS IoT ... 154
AWS IoT Analytics ... 156
AWS IoT Core Device Advisor ... 157
AWS IoT Events ... 157
AWS IoT FleetWise ... 158
AWS IoT Greengrass ... 158
AWS IoT Greengrass Version 2 ... 159

AWS IoT SiteWise console ... 160
AWS IoT Wireless ... 160
Amazon Kendra ... 161
Amazon Kendra Intelligent Ranking ... 162
AWS Key Management Service ... 162
Amazon Keyspaces (for Apache Cassandra) ... 163
Amazon Kinesis ... 163
Amazon Managed Service for Apache Flink ... 163
Amazon Data Firehose ... 164
Amazon Kinesis Video Streams ... 164
AWS Lambda ... 164
AWS Launch Wizard ... 165
Amazon Lex ... 165
AWS License Manager ... 166
Amazon Lightsail ... 166
Linux subscriptions in AWS License Manager ... 167
Amazon Location Service ... 168
Lookout for Equipment ... 168
Amazon Lookout for Metrics ... 169
Lookout for Vision ... 169
Amazon MQ ... 169
Amazon Machine Learning ... 170
Amazon Macie ... 170
AWS Mainframe Modernization ... 171
AWS Mainframe Modernization Application Testing ... 171
Amazon Managed Blockchain ... 172
Amazon Managed Grafana ... 172
Amazon Managed Service for Prometheus ... 173
Amazon Managed Streaming for Apache Kafka ... 173
Amazon Managed Streaming for Apache Kafka Connect ... 174
Amazon Managed Workflows for Apache Airflow ... 174
AWS Marketplace Catalog API ... 174
AWS Elemental MediaConnect ... 175
AWS Elemental MediaPackage ... 175
Amazon MemoryDB ... 176
AWS Migration Hub Orchestrator ... 177

AWS Migration Hub Refactor Spaces ... 177
Amazon Neptune ... 178
AWS Network Firewall ... 178
Network Synthetic Monitor ... 178
AWS Network Manager ... 179
Amazon One ... 180
Amazon OpenSearch Service OpenSearch ... 180
OpenSearch Serverless ... 181
Amazon OpenSearch Service ... 181
Amazon OpenSearch Service Ingestion ... 181
AWS OpsWorks ... 182
AWS Organizations ... 182
AWS Outposts ... 183
AWS Panorama ... 183
AWS Parallel Computing Service ... 183
AWS Payment Cryptography ... 184
Amazon Payments ... 184
Amazon Relational Database Service Performance Insights ... 184
Amazon Personalize ... 185
Amazon Pinpoint ... 186
Amazon Pinpoint SMS and Voice API ... 186
AWS Pricing Calculator ... 187
AWS Private CA Connector for Active Directory ... 187
AWS Private CA Connector for SCEP ... 188
AWS Proton ... 188
Amazon Q Business Apps ... 189
Amazon Q Business ... 189
Amazon Quantum Ledger Database (Amazon QLDB) ... 190
Amazon QuickSight ... 190
AWS DeepRacer ... 191
Recycle Bin ... 192
Amazon Redshift ... 192
Amazon Redshift Serverless ... 193
Amazon Rekognition ... 194
Amazon Relational Database Service (Amazon RDS) ... 194
AWS Resilience Hub ... 196

AWS Resource Access Manager 196
AWS Resource Groups 197
AWS Robomaker 197
Amazon Route 53 198
Amazon Route 53 199
Amazon Route 53 Profiles 199
Amazon Route 53 Recovery Readiness in Application Recovery Controller (ARC) 200
Amazon Route 53 Resolver 200
Amazon S3 Glacier 201
AWS SQL Workbench 202
Amazon SageMaker AI 202
Amazon SageMaker AI geospatial 206
Savings Plans 206
AWS Secrets Manager 207
AWS Security Hub 207
AWS Service Catalog 208
AWS Service Catalog AppRegistry 208
Service Quotas 209
AWS Shield 209
AWS SimSpace Weaver 209
Amazon Simple Email Service 210
Amazon Simple Notification Service 210
Amazon Simple Queue Service 211
Amazon Simple Storage Service (Amazon S3) 211
Amazon Simple Workflow Service 212
AWS Snowball Edge Device Management 212
AWS Step Functions 212
Storage Gateway 213
AWS Supply Chain 213
AWS Systems Manager 214
AWS Systems Manager Incident Manager 214
AWS Systems Manager Incident Manager Contacts 215
AWS Systems Manager Quick Setup 215
AWS Systems Manager for SAP 216
AWS Telco Network Builder 216
Amazon Textract 216

Amazon Timestream ... 217
Amazon Transcribe ... 217
AWS Transfer Family ... 218
Amazon Translate ... 218
AWS User Notifications ... 219
User subscriptions in AWS License Manager ... 219
Amazon VPC Lattice ... 220
AWS Marketplace Vendor Insights ... 221
AWS WAF ... 221
AWS WAF Classic Regional ... 222
AWS Well-Architected Tool ... 222
AWS Wickr ... 223
Amazon WorkMail ... 223
Amazon WorkSpaces ... 223
Amazon WorkSpaces Secure Browser ... 224
Amazon WorkSpaces Thin Client ... 225
AWS X-Ray ... 225
Deprecated resource types ... 226
Creating groups with AWS CloudFormation resources ... 227
Resource Groups and AWS CloudFormation templates ... 227
Learn more about AWS CloudFormation ... 227
Security ... 228
Data protection ... 229
Data encryption ... 230
Internetwork traffic privacy ... 230
Identity and access management ... 230
Audience ... 231
Authenticating with identities ... 231
Managing access using policies ... 234
How Resource Groups works with IAM ... 237
AWS managed policies ... 242
Using service-linked roles ... 247
Identity-based policy examples ... 250
Troubleshooting ... 254
Logging and monitoring ... 256
CloudTrail Integration ... 256

Compliance validation 259
Resilience 260
Infrastructure security 260
AWS PrivateLink 261
Considerations 261
Create an interface endpoint 261
Create an endpoint policy 262
Security best practices 263
Service quotas 265
Document history 266
Earlier updates 277

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.

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
- By using the Resource Groups API, in AWS CLI commands or AWS SDK programming languages. See the *AWS Resource Groups API Reference* for more information.

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.

Resources and their group types

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**

  A tag-based resource group bases its membership on a query that specifies a list of resource types 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**

  An AWS CloudFormation stack-based resource group bases its membership on a query that specifies an AWS CloudFormation stack in your account in the current region. You can optionally 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.

Service-linked resource groups

Some AWS services define resource groups that you can create and manage only by using that service's console and APIs. You are limited in what you can do with these groups in the Resource Groups console. For more information, see Service configurations for resource groups in the *AWS Resource Groups API Reference Guide*.

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 IAM principals, such as roles and users, who are sharing your account have the correct IAM permissions, they can work with resource groups that you create.

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	`arn:aws:resource-groups:` *`region`*`:`*`account`*`:group/`*`group-name`*

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 the Tag Editor User Guide. You can tag supported resources 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
- Learn more about AWS Resource Groups authorization and access control

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.

Topics

- Sign up for AWS
- Create resources
- Set up permissions

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 or text message and entering a verification code on the phone keypad.

   When you sign up for an AWS account, an *AWS account root user* is created. The root user has access to all AWS services and resources in the account. As a security best practice, assign administrative access to a user, and use only the root user to perform tasks that require root user access.

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 Resource types you can use with AWS Resource Groups and Tag Editor.

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 your principals, such as IAM roles or users, or associate external identities with your AWS environment using a service like AWS IAM Identity Center. Then you apply policies with the permissions that your users 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 Resources and their group types, 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, your 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*.

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 role or user 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 role or user 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:

  JSON

```
{
    "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.

JSON

```
{
    "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:*`

Note

The `resource-groups:SearchResources` permission allows Tag Editor to list resources when you filter your search using tag keys or values.

The `resource-explorer:ListResources` permission allows Tag Editor to list resources when you search resources without defining search tags.

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.

   JSON

```
{
  "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 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.

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

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.	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. For more information, see ResourceGroups resource type reference in the *AWS CloudFormation User Guide*.
CloudTrail – Capture all resource group actions using AWS CloudTrail.	Capture information about actions performed on your resource groups including details like who performed the action (IAM principal,

AWS service	Using with Resource Groups
	such as a role, user, 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. For more information, see Viewing events with CloudTrail Event history.
Amazon CloudWatch – Enable real-time monitoring of your AWS resources and the applications you run on AWS.	Focus your view to display metrics and alarms from a single resource group. For more information, see Focus on metrics and alarms in a resource group in the *Amazon CloudWatch User Guide.*
Amazon CloudWatch application insights – Detect common problems with your .NET and SQL Server-based applications.	Monitor your .NET and SQL Server application resources that belong to a resource group. For more information, see Supported application components in the *Amazon CloudWatch User Guide*.
Amazon DynamoDB table groups – Organize your DynamoDB tables into logical groupings so you can more easily manage your resources.	Create, edit, and delete groups of DynamoDB tables from the DynamoDB **Action** menu. For more information, see the *Amazon DynamoDB Developer Guide.*
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.	Launch Amazon EC2 instances into host resource groups to help maximize your utilization of Dedicated Hosts. For more information, see Working with dedicated hosts in the *Amazon EC2 User Guide.*

AWS service	Using with Resource Groups
Amazon EC2 capacity reservations – Reserve capacity for your Amazon EC2 instances to be 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.	Launch your Amazon EC2 instances into resource groups that contain one or more capacity 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. For more information, see Work with Capacity Reservation groups in the *Amazon EC2 User Guide.*
AWS License Manager – Streamline the process of bringing software vendor licenses to the cloud.	Configure a host resource group to enable License Manager to manage your Dedicated Hosts. For more information, see Host Resource Groups in License Manager in the *License Manager User Guide.*
AWS Resilience Hub – Prepare and protect your applications from disruptions.	Discover your applications that are defined using Resource Groups. For more information, see Measure and Improve Your Application Resilience with AWS Resilience Hub in the *AWS News Blog*.
AWS Resource Access Manager – Share specified AWS resources that you own with other accounts.	Share host resource groups using AWS RAM. For more information, see Shareable resources in the *AWS RAM User Guide.*

AWS service	Using with Resource Groups
AWS Service Catalog AppRegistry – Define and manage your applications and their metadata.	When you create an application in 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. For more information, see Using AppRegistry in the *AWS Service Catalog Administrator Guide*.
AWS Systems Manager – Enable visibility and control of your AWS resources.	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. For more information, see Working with applications in Application Manager in the *AWS Systems Manager User Guide*.

AWS service	Using with Resource Groups
Amazon VPC Network Access Analyzer – Identify unwanted network access to your resources on AWS.	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 environme nt, independent of how you configure your network. For more information, see Use Resource Groups with Network Access Scopes in the *Amazon Virtual Private Cloud User Guide*.

Service configurations for resource groups

Resource groups enable you to manage collections of your AWS resources as a unit. Some AWS services support this by performing requested operations on all members of the group. Such services can store the settings to be applied to group members as a *configuration* in the form of a JSON data structure that is attached to the group.

This topic describes the available configuration settings for supported AWS services.

Topics

- How to access the service configuration attached to a resource group
- JSON syntax of a service configuration
- Supported configuration types and parameters

How to access the service configuration attached to a resource group

Services that support service-linked groups typically set the configuration for you when you use the tools provided by that service, such as that service's management console or its AWS CLI and AWS SDK operations. Some services fully manage their service-linked groups and you can't modify them in any way except as allowed by the console or commands provided by the owning AWS service. However, in some cases, you can interact with the service configuration by using the following API operations in the AWS SDKs or their AWS CLI equivalents:

- You can attach your own configuration to a group when you create the group by using the CreateGroup operation.
- You can modify the current configuration attached to a group by using the PutGroupConfiguration operation.
- You can view the current configuration of a resource group by calling the GetGroupConfiguration operation.

JSON syntax of a service configuration

A resource group can contain a *configuration* that defines service-specific settings that apply to the resources that are members of that group.

A configuration is expressed as a JSON object. At the top-most level, a configuration is an array of group configuration items. Each group configuration item contains two elements: a `Type` for the configuration and a set of `Parameters` defined by that type. Each parameter contains a `Name` and an array of one or more `Values`. The following example with *`placeholders`* shows the basic syntax for a configuration for a single sample resource type. This example shows a type with two parameters, and each parameter with two values. The actual valid types, parameters, and values are discussed in the next section.

```
[
  {
    "Type": "configuration-type",
    "Parameters": [
      {
        "Name": "parameter1-name",
        "Values": [
          "value1",
          "value2"
        ]
      },
      {
        "Name": "parameter2-name",
        "Values": [
          "value3",
          "value4"
        ]
      }
    ]
  }
```

```
]
```

Supported configuration types and parameters

Resource Groups supports using the following configuration types. Each configuration type has a set of parameters that are valid for that type.

Topics

- AWS::ResourceGroups::Generic
- AWS::AppRegistry::Application
- AWS::CloudFormation::Stack
- AWS::EC2::CapacityReservationPool
- AWS::EC2::HostManagement
- AWS::NetworkFirewall::RuleGroup

`AWS::ResourceGroups::Generic`

This configuration type specifies settings that enforce membership requirements on the resource group, rather than configuring the behavior of a specific resource type for an AWS service. This configuration type is automatically added by those service-linked groups that need it, such as the `AWS::EC2::CapacityReservationPool` and `AWS::EC2::HostManagment` types.

The following `Parameters` are valid for the `AWS::ResourceGroups::Generic` service-linked group `Type`.

- **`allowed-resource-types`**

  This parameter specifies that the resource group can consist of resources of only the specified type or types.

  Data type of values: String

  Permitted values:

  - `AWS::EC2::Host` – A `Configuration` with this parameter and value is required when the service configuration also contains a `Configuration` of type `AWS::EC2::HostManagement`. This ensures that the `HostManagement` group can contain only Amazon EC2 dedicated hosts.

- `AWS::EC2::CapacityReservation` – A `Configuration` with this parameter and value is required when the service configuration also contains a `Configuration` item of type `AWS::EC2::CapacityReservationPool`. This ensures that a `CapacityReservation` group can contain only Amazon EC2 capacity reservation capacity.

Required: Conditional, based on other `Configuration` elements that are attached to the resource group. See the previous entry for **Permitted values**.

The following example restricts group members to only Amazon EC2 host instances.

```
[
  {
    "Type": "AWS::ResourceGroups::Generic",
    "Parameters": [
      {
        "Name": "allowed-resource-types",
        "Values": [
          "AWS::EC2::Host"
        ]
      }
    ]
  }
]
```

- **`deletion-protection`**

This parameter specifies that the resource group can't be deleted unless it contains no members. For more information, see Delete a host resource group in the *License Manager User Guide*

Data type of values: Array of string

Permitted values: The only permitted value is `[ "UNLESS_EMPTY" ]` (the value must be upper case).

Required: Conditional, based on other `Configuration` elements that are attached to the resource group. This parameter is required only when the resource group also has another `Configuration` element with the `Type` of `AWS::EC2::HostManagement`.

The following example enables delete protection for the group unless the group has no members.

```
[
  {
    "Type": "AWS::ResourceGroups::Generic",
    "Parameters": [
      {
        "Name": "deletion-protection",
        "Values": [
          "UNLESS_EMPTY"
        ]
      }
    ]
  }
]
```

AWS::AppRegistry::Application

This `Configuration` type specifies that the resource group represents an application created by AWS Service Catalog AppRegistry.

Resource groups of this type are fully managed by the AppRegistry service, and can't be created, updated, or deleted by users other than by using the tools provided by AppRegistry.

> **Note**
>
> Because resource groups of this type are automatically created and maintained by AWS and not managed by the user, these resource groups do not count against your quota limit for the maximum number of resource groups that you can create in your AWS account.

For more information, see Using AppRegistry in the *Service Catalog User Guide*.

When AppRegistry creates a service-linked resource group of this type, it also automatically creates a separate, additional AWS CloudFormation service-linked group for each AWS CloudFormation stack associated with the application.

AppRegistry automatically names the service-linked groups of this type that its creates with the prefix `AWS_AppRegistry_Application-` followed by the name of the application: `AWS_AppRegistry_Application-`*`MyAppName`*

The following parameters are supported for the `AWS::AppRegistry::Application` service-linked group type.

- **`Name`**

  This parameter specifies the friendly name of the application that was assigned by the user when it was created in AppRegistry.

  Data type of values: String

  Permitted values: any text string permitted by the AppRegistry service for an application name.

  Required: Yes

- **`Arn`**

  This parameter specifies the Amazon Resource Name (ARN) path of the application assigned by AppRegistry.

  Data type of values: String

  Permitted values: a valid ARN.

  Required: Yes

> **Note**
>
> To change any of these elements, you must modify the application using the AppRegistry console or that service's AWS SDK and AWS CLI operations.

This application resource group automatically includes as group members the resource groups created for the AWS CloudFormation stacks that are associated with the AppRegistry application. You can use the ListGroupResources operation to see those child groups.

The following example shows what the configuration section of a `AWS::AppRegistry::Application` service-linked group looks like.

```
[
  {
```

```
    "Type": "AWS::AppRegistry::Application",
    "Parameters": [
      {
        "Name": "Name",
        "Values": [
          "MyApplication"
        ]
      },
      {
        "Name": "Arn",
        "Values": [
          "arn:aws:servicecatalog:us-east-1:123456789012:/applications/<application-
id>"
        ]
      }
    ]
  }
]
```

AWS::CloudFormation::Stack

This `Configuration` type specifies that the group represents an AWS CloudFormation stack and its members are the AWS resources created by that stack.

Resource groups of this type are automatically created for you when you associate a AWS CloudFormation stack with the AppRegistry service. You can't create, update, or delete these groups except by using the tools provided by AppRegistry.

AppRegistry automatically names the service-linked groups of this type that its creates with the prefix `AWS_CloudFormation_Stack-` followed by the name of the stack: `AWS_CloudFormation_Stack-`*`MyStackName`*

> **Note**
>
> Because resource groups of this type are automatically created and maintained by AWS and not managed by the user, these resource groups do not count against your quota limit for the maximum number of resource groups that you can create in your AWS account.

For more information, see Using AppRegistry in the *Service Catalog User Guide*.

AppRegistry automatically creates a service-linked resource group of this type for every AWS CloudFormation stack that you associate with the AppRegistry application. These resource groups become child members of the parent resource group for the AppRegistry application.

The members of this AWS CloudFormation resource group are the AWS resources created as part of the stack.

The following parameters are supported for the `AWS::CloudFormation::Stack` service-linked group type.

- **`Name`**

  This parameter specifies the friendly name of the AWS CloudFormation stack assigned by the user when the stack was created.

  Data type of values: String

  Permitted values: any text string permitted by the AWS CloudFormation service for a stack name.

  Required: Yes

- **`Arn`**

  This parameter specifies the Amazon Resource Name (ARN) path of the AWS CloudFormation stack attached to the application in AppRegistry.

  Data type of values: String

  Permitted values: a valid ARN.

  Required: Yes

> **Note**
>
> To change any of these elements, you must modify the application using the AppRegistry console or equivalent AWS SDK and AWS CLI operations.

The following example shows what the configuration section of an `AWS::CloudFormation::Stack` service-linked group looks like.

```
[
  {
    "Type": "AWS::CloudFormation::Stack",
    "Parameters": [
      {
        "Name": "Name",
        "Values": [
          "MyStack"
        ]
      },
      {
        "Name": "Arn",
        "Values": [
          "arn:aws:cloudformation:us-east-1:123456789012:stack/MyStack/<stack-id>"
        ]
      }
    ]
  }
]
```

`AWS::EC2::CapacityReservationPool`

This `Configuration` type specifies that the resource group represents a common pool of capacity provided by the group's members. The members of this resource group are required to be Amazon EC2 capacity reservations. A resource group can include both capacity reservations that you own in your account and capacity reservations that are shared with you from other accounts by using AWS Resource Access Manager. This lets you launch an Amazon EC2 instance using this resource group as the value for the capacity reservation parameter. When you do this, the instance uses the available reserved capacity in the group.

If the resource group has no available capacity, the instance launches as a stand alone on-demand instance outside of the pool unless you configure the resource group to use Amazon EC2 UltraServer Capacity Blocks. For more information, see Working with Capacity Reservation groups in the *Amazon EC2 User Guide*.

If you configure a service-linked resource group with a `Configuration` item of this type, then you must also specify separate `Configuration` items with the following values:

- An `AWS::ResourceGroups::Generic` type with one parameter:
  - The parameter `allowed-resource-types` and a single value of `AWS::EC2::CapacityReservation`. This ensures that only Amazon EC2 capacity reservations can be members of the resource group.
- A `AWS::EC2::CapacityReservationPool` type with two parameters:
  - `reservation-type`— Only required when you configure a Capacity Reservation Group for Amazon EC2 UltraServer Capacity Blocks. The only allowed value in this field is `capacity-block`.
  - `instance-type`— Only required when you configure a Capacity Reservation Group for Amazon EC2 UltraServer Capacity Blocks. The allowed values in this field are `trn2u.48xlarge` and `p6e-gb200.36xlarge`.

The following example shows the `Configuration` section of an On-Demand Capacity Reservation:

```
{
  "Configuration": [
    {
      "Type": "AWS::EC2::CapacityReservationPool",
      "Parameters": []
    },
    {
      "Type": "AWS::ResourceGroups::Generic",
      "Parameters": [
        {
          "Name": "allowed-resource-types",
          "Values": [
            "AWS::EC2::CapacityReservation"
          ]
        }
      ]
    }
  ]
}
```

The following example shows the `Configuration` section supporting Amazon EC2 UltraServer Capacity Blocks:

```
{
```

```
  "Configuration": [
    {
      "Type": "AWS::EC2::CapacityReservationPool",
      "Parameters": [
        {
          "Name": "instance-type",
          "Values": [
            "trn2u.48xlarge"
          ]
        },
        {
          "Name": "reservation-type",
          "Values": [
            "capacity-block"
          ]
        }
      ]
    },
    {
      "Type": "AWS::ResourceGroups::Generic",
      "Parameters": [
        {
          "Name": "allowed-resource-types",
          "Values": [
            "AWS::EC2::CapacityReservation"
          ]
        }
      ]
    }
  ]
}
```

After adding `instance-type` and `reservation-type` to a resource group configuration when you use Amazon EC2 UltraServer Capacity Blocks, the following behaviors apply to that resource group:

- You can add additional capacity reservations into this resource group configuration but additional reservations must also have the `reservation-type` set to `capacity-block` and the `instance-type` set to `trn2u.48xlarge` or `p6e-gb200.48xlarge`.
- Currently, the only allowable value for `reservation-type` is `capacity-block`, and the only allowable values for `instance-type` are `trn2u.48xlarge` and `p6e-gb200.48xlarge`.

- You can't add Amazon EC2 Capacity Blocks for ML into a resource group that does not include the `reservation-type` and `instance-type` configurations.
- Adding the `reservation-type` and `capacity-block` configuration parameters does not change the process of adding or removing group reservations.
- If you remove the capacity reservation from the group, or delete the group, the reservations inside the group remain in use until the instances are terminated.
- Currently, resource groups with the `reservation-type` and `instance-type` configuration parameters can't be updated after initial setup. To change or remove the configuration, you must delete the group and then create a new group with new configurations.
- You can't launch an instance into an empty group or modify an instance to target an empty group.

`AWS::EC2::HostManagement`

This identifier specifies settings for Amazon EC2 host management and AWS License Manager that are enforced for the group's members. For more information, see Host resource groups in AWS License Manager.

If you configure a service-linked resource group with a `Configuration` item of this type, then you must also specify separate `Configuration` items with the following values:

- An `AWS::ResourceGroups::Generic` type, with a parameter of `allowed-resource-types` and a single value of `AWS::EC2::Host`. This ensures that only Amazon EC2 dedicated hosts can be members of the group.
- An `AWS::ResourceGroups::Generic` type, with a parameter of `deletion-protection` and a single value of `UNLESS_EMPTY`. This ensures that the group can't be deleted unless the group is empty.

The following parameters are supported for the `AWS::EC2::HostManagement` service-linked group type.

- **`auto-allocate-host`**

  This parameter specifies whether instances are launched onto a specific dedicated host, or onto any available host that has a matching configuration. For more information, see Understanding auto-placement and affinity in the *Amazon EC2 User Guide*.

Data type of values: Boolean

Permitted values: "true" or "false" (must be lower case).

Required: No

```
[
  {
    "Type": "AWS::EC2::HostManagement",
    "Parameters": [
      {
        "Name": "auto-allocate-host",
        "Values": [
          "true"
        ]
      },
      {
        "Name": "any-host-based-license-configuration",
        "Values": [
          "true"
        ]
      }
    ]
  },
  {
    "Type": "AWS::ResourceGroups::Generic",
    "Parameters": [
      {
        "Name": "allowed-resource-types",
        "Values": [
          "AWS::EC2::Host"
        ]
      },
      {
        "Name": "deletion-protection",
        "Values": [
          "UNLESS_EMPTY"
        ]
      }
    ]
  }
]
```

- **`auto-release-host`**

  This parameter specifies whether a dedicated host in the group is automatically released after its last running instance is terminated. For more information, see Releasing Dedicated Hosts in the *Amazon EC2 User Guide*.

  Data type of values: Boolean

  Permitted values: "true" or "false" (must be lower case).

  Required: No

```
[
  {
    "Type": "AWS::EC2::HostManagement",
    "Parameters": [
      {
        "Name": "auto-release-host",
        "Values": [
          "false"
        ]
      },
      {
        "Name": "any-host-based-license-configuration",
        "Values": [
          "true"
        ]
      }
    ]
  },
  {
    "Type": "AWS::ResourceGroups::Generic",
    "Parameters": [
      {
        "Name": "allowed-resource-types",
        "Values": [
          "AWS::EC2::Host"
        ]
      },
      {
        "Name": "deletion-protection",
        "Values": [
          "UNLESS_EMPTY"
```

```
        ]
      }
    ]
  }
]
```

- **`allowed-host-families`**

  This parameter specifies which instance type families can be used by instances that are members of this group.

  Data type of values: An array of String.

  Permitted values: Each must be a valid Amazon EC2 instance type family identifier, such as `C4`, `M5`, `P3dn`, or `R5d`.

  Required: No

  The following example configuration item specifies that launched instances can be only members of the C5 or M5 instance type families.

```
[
  {
    "Type": "AWS::EC2::HostManagement",
    "Parameters": [
      {
        "Name": "allowed-host-families",
        "Values": [
          "c5",
          "m5"
        ]
      },
      {
        "Name": "any-host-based-license-configuration",
        "Values": [
          "true"
        ]
      }
    ]
  },
  {
    "Type": "AWS::ResourceGroups::Generic",
    "Parameters": [
```

```
      {
        "Name": "allowed-resource-types",
        "Values": [
          "AWS::EC2::Host"
        ]
      },
      {
        "Name": "deletion-protection",
        "Values": [
          "UNLESS_EMPTY"
        ]
      }
    ]
  }
]
```

- **`allowed-host-based-license-configurations`**

  This parameter specifies the Amazon Resource Name (ARN) paths of one or more core/socket based license configurations that you want applied to members of the group.

  Data type of values: An array of ARNs.

  Permitted values: Each must be a valid License Manager configuration ARN.

  Required: Conditional. You must specify either this parameter or `any-host-based-license-configuration`, but not both. They are mutually exclusive.

  The following example configuration item specifies that group members can use the two specified License Manager configurations.

```
[
  {
    "Type": "AWS::EC2::HostManagement",
    "Parameters": [
      {
        "Name": "allowed-host-based-license-configurations",
        "Values": [
          "arn:aws:license-manager:us-west-2:123456789012:license-
configuration:lic-6eb6586f508a786a2ba41EXAMPLE1111",
          "arn:aws:license-manager:us-west-2:123456789012:license-
configuration:lic-8a786a26f50ba416eb658EXAMPLE2222"
        ]
```

```
      }
    ]
  },
  {
    "Type": "AWS::ResourceGroups::Generic",
    "Parameters": [
      {
        "Name": "allowed-resource-types",
        "Values": [
          "AWS::EC2::Host"
        ]
      },
      {
        "Name": "deletion-protection",
        "Values": [
          "UNLESS_EMPTY"
        ]
      }
    ]
  }
]
```

- **`any-host-based-license-configuration`**

  This parameter specifies that you do not want to associate a specific license configuration to your group. In this case, all core/socket based license configurations are available to your members of your host resource group. Use this setting if you have an unlimited number of licenses and want to optimize for host utilization.

  Data type of values: Boolean

  Permitted values: "true" or "false" (must be lower case).

  Required: Conditional. You must specify either this parameter or `allowed-host-based-license-configurations`, but not both. They are mutually exclusive.

  The following example configuration item specifies that group members can use any core/socket based license configuration.

```
[
  {
    "Type": "AWS::EC2::HostManagement",
    "Parameters": [
```

```
      {
        "Name": "any-host-based-license-configuration",
        "Values": [
          "true"
        ]
      }
    ]
  },
  {
    "Type": "AWS::ResourceGroups::Generic",
    "Parameters": [
      {
        "Name": "allowed-resource-types",
        "Values": [
          "AWS::EC2::Host"
        ]
      },
      {
        "Name": "deletion-protection",
        "Values": [
          "UNLESS_EMPTY"
        ]
      }
    ]
  }
]
```

The following example illustrates how to include all of the host management settings together in a single configuration.

```
[
  {
    "Type": "AWS::EC2::HostManagement",
    "Parameters": [
      {
        "Name": "auto-allocate-host",
        "Values": [
          "true"
        ]
      },
      {
        "Name": "auto-release-host",
```

```
        "Values": [
          "false"
        ]
      },
      {
        "Name": "allowed-host-families",
        "Values": [
          "c5",
          "m5"
        ]
      },
      {
        "Name": "allowed-host-based-license-configurations",
        "Values": [
          "arn:aws:license-manager:us-west-2:123456789012:license-
configuration:lic-6eb6586f508a786a2ba41EXAMPLE1111",
          "arn:aws:license-manager:us-west-2:123456789012:license-
configuration:lic-8a786a26f50ba416eb658EXAMPLE2222"
        ]
      }
    ]
  },
  {
    "Type": "AWS::ResourceGroups::Generic",
    "Parameters": [
      {
        "Name": "allowed-resource-types",
        "Values": [
          "AWS::EC2::Host"
        ]
      },
      {
        "Name": "deletion-protection",
        "Values": [
          "UNLESS_EMPTY"
        ]
      }
    ]
  }
]
```

AWS::NetworkFirewall::RuleGroup

This identifier specifies settings for AWS Network Firewall rule groups that are enforced for the group's members. Firewall administrators can specify the ARN of a resource group of this type to automatically resolve the IP addresses of the group's members for a firewall rule instead of having to list each address manually. For more information, see Using tag-based resource groups in AWS Network Firewall.

You can create resource groups of this configuration type by using the Network Firewall console or by running a AWS CLI command or AWS SDK operation.

Resource groups of this configuration type have the following restrictions:

- The group's members consist of only resources of types supported by Network Firewall.
- The group must contain a tag-based query to manage the group's membership; any resources of supported types with tags that match the query are automatically members of the group.
- There are no `Parameters` supported for this configuration type.
- To delete a resource group of this configuration type, it can't be referenced by any Network Firewall rule group.

The following example illustrates the `Configuration` and `ResourceQuery` sections for a group of this type.

```
{
  "Configuration": [
    {
      "Type": "AWS::NetworkFirewall::RuleGroup",
      "Parameters": []
    }
  ],
  "ResourceQuery": {
    "Query": "{\"ResourceTypeFilters\":[\"AWS::EC2::Instance\"],\"TagFilters\":[{\"Key
\":\"environment\",\"Values\":[\"production\"]}]}",
    "Type": "TAG_FILTERS_1_0"
  }
}
```

The following example AWS CLI command creates a resource group with the previous configuration and query.

```
$ aws resource-groups create-group \
    --name test-group \
    --resource-query '{"Type": "TAG_FILTERS_1_0", "Query": "{\"ResourceTypeFilters\":
 [\"AWS::EC2::Instance\"], \"TagFilters\": [{\"Key\": \"environment\", \"Values\":
 [\"production\"]}]}"}' \
    --configuration '[{"Type": "AWS::NetworkFirewall::RuleGroup", "Parameters": []}]'
{
    "Group":{
        "GroupArn":"arn:aws:resource-groups:us-west-2:123456789012:group/test-group",
        "Name":"test-group",
        "OwnerId":"123456789012"
    },
    "Configuration": [
        {
            "Type": "AWS::NetworkFirewall::RuleGroup",
            "Parameters": []
        }
    ],
    "ResourceQuery": {
        "Query": "{\"ResourceTypeFilters\":[\"AWS::EC2::Instance\"],\"TagFilters\":
[{\"Key\":\"environment\",\"Values\":[\"production\"]}]}",
        "Type": "TAG_FILTERS_1_0"
    }
}
```

Creating query-based groups in AWS Resource Groups

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 Resource Groups 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 the filter and are members of 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.

> **Note**
>
> An AWS CloudFormation stack can contain other AWS CloudFormation "child" stacks. However, a resource group based on a "parent" stack doesn't get all of the child stacks' resources as group members. Resource groups adds the child stacks to the parent stack's resource group as single group members and doesn't expand them.

Resource Groups supports queries based on AWS CloudFormation stacks that have one of the following statuses.

- `CREATE_COMPLETE`
- `CREATE_IN_PROGRESS`
- `DELETE_FAILED`
- `DELETE_IN_PROGRESS`
- `REVIEW_IN_PROGRESS`

> **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 Resource types you can use with AWS Resource Groups and Tag Editor.
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

The following procedures show you how to update a tag-based query group.

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 pres **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

The following procedures show you how to update a 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 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 Resource types you can use with AWS Resource Groups and Tag Editor.

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.

Group lifecycle events: Monitoring resource groups for changes

After you use AWS Resource Groups to organize your resources into groups, you can monitor those groups for changes that are exposed to you as *events*. You can receive a notification about a group event as a signal for you to take some kind of action. For example, you could configure a notification that is sent whenever a group's membership changes. You could use an event from adding a new group member to trigger a Lambda function that programmatically reviews the change to ensure that new group members meet compliance requirements set by your organization. Such a Lambda function could perform automatic remediation for any new group members that fail to meet those requirements. An event caused by the removal of a group member could trigger a Lambda function that performs any required cleanup, such as deleting linked resources.

By turning on group lifecycle events for your resource groups, you allow events about changes to your groups to be captured by Amazon EventBridge and made available to all of the various EventBridge supported target services. You can then configure those target services to automatically take whatever actions your scenario requires. These targets include a variety of AWS services such as Amazon Simple Notification Service (Amazon SNS), Amazon Simple Queue Service (Amazon SQS), and AWS Lambda. With services like Lambda, your events can trigger *programmatic* responses that use code to perform whatever actions you require. For a list of the AWS services that you can target with EventBridge, see Amazon EventBridge targets in the *Amazon EventBridge User Guide*.

When you turn on group lifecycle events, AWS Resource Groups creates the following items:

- An AWS Identity and Access Management (IAM) service-linked role that has permission to monitor your resources for any changes to their tags and your AWS CloudFormation stacks for any changes to the resources that are part of a stack.
- A Resource Groups managed EventBridge rule that captures the details of any tag or stack changes to your resources. EventBridge uses this rule to notify Resource Groups about those changes. Then, Resource Groups generates membership events to send to EventBridge for your custom rules to process.

The service-linked role can be assumed by *only* the Resource Groups service. For more information about the service-linked role used by Resource Groups for this feature, see Using service-linked roles for Resource Groups.

When this feature is turned on, Resource Groups generates an event when you make any of the following changes to a resource group:

- Create a new resource group.
- Update the query which defines the membership of query-based resource group.
- Update the configuration of a service linked resource group.
- Update the description of a resource group.
- Delete a resource group.
- Change a resource group's membership by adding or removing a resource from the group. A membership change can also happen when tags change, or when a AWS CloudFormation stack changes.

Important

- To successfully receive and respond to group events, you must make changes to both Resource Groups and EventBridge. You can perform the changes in any order, but no group events are published to EventBridge targets until after you make changes to both services.

- The resource group changes don't include changes to any tags attached to the resource group itself. To generate events based on tag changes to your groups, you must use an EventBridge rule that uses the `aws.tag` source, instead of the `aws.resource-groups` source. For more information, see Tag change events on AWS Resources in the *Amazon EventBridge User Guide*.

Topics

- Turning on group lifecycle events in Resource Groups
- Creating an EventBridge rule to capture group lifecycle events and publish notifications
- Turning off group lifecycle events

- Structure and syntax of Resource Groups lifecycle events

Turning on group lifecycle events in Resource Groups

To receive notifications about lifecycle changes to your resource groups, you can turn on group lifecycle events. Resource Groups then provides information about your groups' changes to Amazon EventBridge. In EventBridge, you can evaluate and act on the changes using rules you define in the EventBridge service.

Minimum permissions

To turn on group lifecycle events in your AWS account, you must sign in as an AWS Identity and Access Management (IAM) principal with the following permissions:

- `resource-groups:UpdateAccountSettings`
- `iam:CreateServiceLinkedRole`
- `events:PutRule`
- `events:PutTargets`
- `events:DescribeRule`
- `events:ListTargetsByRule`
- `cloudformation:DescribeStacks`
- `cloudformation:ListStackResources`
- `tag:GetResources`

When you initially turn on group lifecycle events in an AWS account, Resource Groups creates a service-linked role named `AWSServiceRoleForResourceGroups`. This managed role has permission to use a Resource Groups managed EventBridge rule. The rule monitors the tags attached to your resources and the AWS CloudFormation stacks in your account for any changes. Resource Groups then publishes those changes to the default event bus in Amazon EventBridge. The service also creates an EventBridge managed rule named `Managed.ResourceGroups.TagChangeEvents`. This rule captures the details of tag changes of your resources. This lets Resource Groups generate membership events to send to EventBridge for your custom rules to process. Your EventBridge rules can then respond to events by sending notifications to the rules' configured targets.

After you complete these steps, rules that look for these events should start receiving them in a few minutes.

You can turn on group lifecycle events by using either the AWS Management Console or by using a command from the AWS CLI or one of the SDK APIs.

> **Note**
>
> You can't turn on group lifecycle events if your resource groups quota is too high. For more information, review Viewing service quotas.

AWS Management Console

To turn on group lifecycle events in the Resource Groups console

1. Open the **Settings** page in the Resource Groups console.
2. In the **Group lifecycle events** section, choose the switch next to **Notifications are turned off**.
3. On the confirmation dialog, choose **Turn on notifications**.

   The feature switch displays **Notifications are turned on**.

That completes the first part of the process. After you turn on event notifications, you can create rules in Amazon EventBridge that capture the events and send them to specific AWS services for processing.

AWS CLI

To turn on group lifecycle events by using the AWS CLI or the AWS SDKs

The following example show how to use the AWS CLI to turn on group lifecycle events in Resource Groups. Enter the command with the service principal parameter exactly as shown. The output shows both the current status and the desired status of the feature.

```
$ aws resource-groups update-account-settings \
    --group-lifecycle-events-desired-status ACTIVE
{
    "AccountSettings": {
```

```
        "GroupLifecycleEventsDesiredStatus": "ACTIVE",
        "GroupLifecycleEventsStatus": "IN_PROGRESS"
    }
}
```

You can confirm that the feature is turned on by running the following example command. When both status fields show the same value, then the operation is complete.

```
$ aws resource-groups get-account-settings
{
    "AccountSettings": {
        "GroupLifecycleEventsDesiredStatus": "ACTIVE",
        "GroupLifecycleEventsStatus": "ACTIVE"
    }
}
```

For more information, see the following resources:

- AWS CLI – aws resource-groups update-account-settings and aws resource-groups get-account-settings
- API – UpdateAccountSettings and GetAccountSettings

Creating an EventBridge rule to capture group lifecycle events and publish notifications

You can turn on group lifecycle events for your resource groups in AWS Resource Groups to publish events to Amazon EventBridge. Then, you can create EventBridge rules that respond to those events by sending them to other AWS services for further processing.

AWS CLI

The process to create a rule in EventBridge that captures events and sends them to your desired target service takes two separate CLI commands:

1. Create the EventBridge rule to capture the events you want
2. Attach a target that can process the events to the EventBridge rule

Step 1: Create the EventBridge rule to capture the events

The following AWS CLI `put-rule` example command creates an EventBridge rule that captures ***all*** Resource Groups lifecycle event changes.

```
$ aws events put-rule \
    --name "CatchAllResourceGroupEvents" \
    --event-pattern '{"source":["aws.resource-groups"]}'
{
    "RuleArn": "arn:aws:events:us-east-1:123456789012:rule/
CatchAllResourceGroupEvents"
}
```

The output includes the Amazon Resource Name (ARN) of the new rule.

> **Note**
>
> Parameter values that include quoted strings have different formatting rules based on the operating system and shell that you use. For the examples in this guide, we show commands that work on a Linux BASH shell. For instructions about formatting strings with embedded quotes for other operating systems, such as the Windows command prompt, see Using quotation marks inside strings in the *AWS Command Line Interface User Guide*.
>
> As parameter strings get more complex, it can be easier and less error prone to accept a parameter value from a text file instead of typing it directly on the command line.

The following event pattern restricts the events to only those that are related to the specified group, identified by its ARN. This event pattern is a complex JSON string that is much less readable when compressed into a single-line, properly escaped JSON string. You can store it in a file instead.

Store the event pattern JSON string in a file. In the following code example, the file is `eventpattern.txt`.

```
{
    "source": [ "aws.resource-groups" ],
    "detail": {
```

```
        "group": {
            "arn": [ "my-resource-group-arn" ]
        }
    }
}
```

Then, issue the following command to create the rule, retrieving the custom event pattern from the file.

```
$ aws events put-rule \
    --name "CatchResourceGroupEventsForMyGroup" \
    --event-pattern file://eventpattern.txt
{
    "RuleArn": "arn:aws:events:us-east-1:123456789012:rule/
CatchResourceGroupEventsForMyGroup"
}
```

To capture other types of Resource Groups events, replace the `--event-pattern` string with filters like those presented in the section Example EventBridge custom event patterns for different use cases.

Step 2: Attach a target that can process the events to the EventBridge rule

Now that you have a rule that captures the events of interest to you, you can attach one or more targets to do some type of processing on the events.

The following AWS CLI `put-targets` command attaches an Amazon Simple Notification Service (Amazon SNS) topic named `my-sns-topic` to the rule you created in the previous example. All subscribers to the topic receive a notification when a change occurs to the group specified in the rule.

```
$ aws events put-targets \
    --rule CatchResourceGroupEventsForMyGroup \
    --targets Id=1,Arn=arn:aws:sns:us-east-1:123456789012:my-sns-topic
{
    "FailedEntryCount": 0,
    "FailedEntries": []
}
```

At this point, any group changes that match the event pattern in your rule are automatically sent to the configured target or targets. If, as in the previous example, the target is an

Amazon SNS topic, then all subscribers to the topic receive a message containing the event as described in Structure and syntax of Resource Groups lifecycle events.

For more information, see the following resources:

- AWS CLI – aws events put-rule and aws events put-targets
- API – PutRule and PutTargets

Creating a rule to capture only specific group lifecycle event types

You can create a rule with a custom event pattern that captures only the events that you are interested in. For complete details about how to filter incoming events using a custom event pattern, see Amazon EventBridge events in the *Amazon EventBridge User Guide*.

For example, suppose you want a rule to process only those Resource Groups notifications that indicate the creation of a new resource group. You could use a custom event pattern similar to the following example.

```
{
    "source": [ "aws.resource-groups" ],
    "detail-type": [ "ResourceGroups Group State Change" ],
    "detail": {
        "state-change": "create"
    }
}
```

That filter captures only those events that have those exact values in the specified fields. For a complete list of the fields available for you to match, see Structure and syntax of Resource Groups lifecycle events.

Turning off group lifecycle events

You can turn off group lifecycle events to stop AWS Resource Groups from emitting events to Amazon EventBridge. You can do this by using either the AWS Management Console or by using a command from the AWS CLI or one of the SDK APIs.

Note

Turning off group lifecycle events deletes the Resource Groups managed EventBridge rule used to scan your resource tags and AWS CloudFormation stacks for changes. Resource Groups can no longer pass those changes to EventBridge. Any rules you defined in EventBridge that look for Resource Groups events stop receiving events to process. If you intend to turn on group lifecycle events again in the future, you can disable your rules. If you don't intend to use those rules again, you can delete them. For more information, see Disabling or deleting an EventBridge rule in the *Amazon EventBridge User Guide*.
Turning off group lifecycle events does ***not*** delete the service-linked role. You can delete the service-linked role manually if you wish using IAM. If you later need to turn on group lifecycle events again and the service-linked role doesn't exist, Resource Groups recreates it automatically.

Minimum permissions

To turn off group lifecycle events in your current AWS account, you must sign in as an AWS Identity and Access Management (IAM) principal with the following permissions:

- `resource-groups:UpdateAccountSettings`
- `events:DeleteRule`
- `events:RemoveTargets`
- `events:DescribeRule`
- `events:ListTargetsByRule`

AWS Management Console

To turn off group lifecycle event notifications to EventBridge

1. Open the **Settings** page in the Resource Groups console.
2. In the **Group lifecycle events** section, choose the switch next to **Notifications are turned on**.
3. On the confirmation dialog, choose **Turn off notifications**.

   The feature switch is displayed: **Event notifications are turned off**.

At this point, Resource Groups no longer sends events to the EventBridge default event bus, and any rules that you have no longer receive group notification events to process. You can optionally delete those rules to complete the clean up.

AWS CLI

To turn off group lifecycle event notifications to EventBridge

The following example show how to use the AWS CLI to turn off group lifecycle events in Resource Groups.

```
$ aws resource-groups update-account-settings \
    ----group-lifecycle-events-desired-status INACTIVE
{
    "AccountSettings": {
        "GroupLifecycleEventsDesiredStatus": "INACTIVE",
        "GroupLifecycleEventsStatus": "INACTIVE"
    }
}
```

For more information, see the following resources:

- AWS CLI – aws resource-groups update-account-settings and aws resource-groups get-account-settings
- API – UpdateAccountSettings and GetAccountSettings

Structure and syntax of Resource Groups lifecycle events

Topics

- Structure of the detail field
- Example EventBridge custom event patterns for different use cases

The lifecycle events for AWS Resource Groups take the form of JSON object strings in the following general format.

```
{
    "version": "0",
    "id": "08f00e24-2e30-ec44-b824-8acddf1ac868",
```

```
    "detail-type": "ResourceGroups Group ... Change",
    "source": "aws.resource-groups",
    "account": "123456789012",
    "time": "2020-09-29T09:59:01Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:resource-groups:us-east-1:123456789012:group/MyGroupName"
    ],
    "detail": {
        ...
    }
}
```

For details about the fields common to all Amazon EventBridge events, see Amazon EventBridge events in the *Amazon EventBridge User Guide*. Details that are specific to Resource Groups are explained in the following table.

Field name	Type	Description
`detail-type`	String	For Resource Groups, the `detail-type` field is always one of the following values: • `ResourceGroups Group State Change` – Represents changes to the overall group state and its properties. • `ResourceGroups Group Membership Change` – Represents changes to the group membership.
`source`	String	For Resource Groups, this value is **always** `"aws.resource-groups"`.
`resources`	An array of Amazon Resource Names (ARNs)	This field always includes the Amazon resource name (ARN) of the group with the change that triggered this event. This field can also include the ARNs of any resources added to or removed from the group, if applicable.

Field name	Type	Description
`detail`	JSON object string	This is the payload of the event. The contents of the `detail` field vary based on the value of the `detail-type` . See the next section for more information.

Structure of the `detail` field

The `detail` field includes all of the Resource Groups service-specific details about a specific change. The `detail` field can take one of two forms, a group state change or membership change, based on the value of the `detail-type` field described in the previous section.

> **Important**
>
> Resource groups in these events are identified by a combination of the group's ARN and a `"unique-id"` field that contains a UUID. By including a UUID as part of the identity of a resource group, you can distinguish between a group that is deleted and a different group that is later created with the same name. We recommend that you treat a concatenation of the ARN and unique id as the key for the group in your programs that interact with these events.

Group state change

`"detail-type": "ResourceGroups Group State Change"`

This `detail-type` value indicates that the state of the group itself, including its metadata, has changed. This change occurs when a group is created, updated, or deleted, as indicated by the `"change"` field within the `detail`.

The information included in the `details` section when this `detail-type` is specified include the fields described in the following table.

Field name	Type	Description
`event-sequence`	Double	A monotonically increasing number that specifies the sequence of events for a specific group. The

Field name	Type	Description
		number resets when you delete the group and create another group with the same name.
group	Group JSON object	The group object associated with the event by its ARN, name, and unique ID.
state-change	String	The type of state change that occurred. Can be any of the following values: • create • update • delete
old-state	GroupState JSON object	The state of the group before the change. The object includes only the values of properties that changed.
new-state	GroupState JSON object	The state of the group after the change. The object includes only the values of properties that changed.

The group JSON object contains the elements described in the following table.

Field name	Type	Description
arn	String	The ARN of the group.
name	String	The friendly name of the group.
unique-id	GUID	A unique GUID value that distinguishes between a group that was deleted and a different group that was later created with the same name and ARN. Use the concatenation of ARN and this value as a unique key for the group when consuming these events in your code.

The `GroupState` JSON objects contain the elements described in the following table.

Field name	Type	Description
`description`	String	The customer-provided description of the resource group.
`resource-query`	`ResourceQuery` JSON object	A JSON representation of the query that defines the group's members. This field is present only for groups based on a query. The syntax of this field is defined by the ResourceQuery API data type. Example of this are included in the Create and Update event examples.
`group-configuration`	`Configuration` JSON object	A JSON representation of configuration parameters associated with a service-linked group. For more information, see Service configurations for resource groups in the *AWS Resource Groups API Reference*.

Each of the following code examples illustrates the contents of the `detail` field for each `state-change` type.

Create

`"state-change": "create"`

The event indicates that a new group was created. The event carries all the group metadata properties set during the group's creation. This event is typically followed by one of more group membership events unless the group is empty. Properties that have a null value are not displayed in the event body.

The following example event indicates a newly created resource group named `my-service-group`. In this example, the group uses a tag-based query that matches only Amazon Elastic Compute Cloud (Amazon EC2) instances that have the tag `"project"="my-service"`.

```
{
    "version": "0",
```

```
    "id": "08f00e24-2e30-ec44-b824-8acddf1ac868",
    "detail-type": "ResourceGroups Group State Change",
    "source": "aws.resource-groups",
    "account": "123456789012",
    "time": "2020-09-29T09:59:01Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:resource-groups:us-east-1:123456789012:group/my-service-group"
    ],
    "detail": {
        "event-sequence": 1.0,
        "state-change": "create",
        "group": {
            "arn": "arn:aws:resource-groups:us-east-1:123456789012:group/my-service-
group",
            "name": "my-service-group",
            "unique-id": "3dd07ab7-3228-4410-8cdc-6c4a10fcceea"
        },
        "new-state": {
            "resource-query": {
                "type": "TAG_FILTERS_1_0",
                "query": "{
                    \"ResourceTypeFilters\": [\"AWS::EC2::Instance\"],
                    \"TagFilters\": [{\"Key\":\"project\", \"Values\":[\"my-service\"}]
                }"
            }
        }
    }
}
```

Update

`"state-change": "update"`

The event indicates that an existing group was modified in some way. The event carries only the properties that changed from the previous state. Properties that have not changed are not displayed in the event body.

The following example event indicates that the tag-based query in the previous example's resource group was modified to also include Amazon EC2 volume resources in the group.

```
{
    "version": "0",
```

```
    "id": "08f00e24-2e30-ec44-b824-8acddf1ac868",
    "detail-type": "ResourceGroups Group State Change",
    "source": "aws.resource-groups",
    "account": "123456789012",
    "time": "2020-09-29T09:59:01Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:resource-groups:us-east-1:123456789012:group/my-service-group"
    ],
    "detail": {
        "event-sequence": 3.0,
        "state-change": "update",
        "group": {
            "arn": "arn:aws:resource-groups:us-east-1:123456789012:group/my-service-
group",
            "name": "my-service",
            "unique-id": "3dd07ab7-3228-4410-8cdc-6c4a10fcceea"
        },
        "new-state": {
            "resource-query": {
                "type": "TAG_FILTERS_1_0",
                "query": "{
                    \"ResourceTypeFilters\": [\"AWS::EC2::Instance\",
 \"AWS::EC2::Volume\"],
                    \"TagFilters\": [{\"Key\":\"project\", \"Values\":[\"my-service\"}]
                }"
            }
        },
        "old-state": {
            "resource-query": {
                "type": "TAG_FILTERS_1_0",
                "query": "{
                    \"ResourceTypeFilters\": [\"AWS::EC2::Instance\"],
                    \"TagFilters\": [{\"Key\":\"Project\", \"Values\":[\"my-service\"}]
                }"
            }
        }
    }
}
```

Delete

```
"state-change": "delete"
```

The event indicates that an existing group was deleted. The detail field includes no metadata about the group other than its identification. The `event-sequence` field is reset after this event as it is, by definition, the last event for this `arn` and `unique-id`.

```
{
    "version": "0",
    "id": "08f00e24-2e30-ec44-b824-8acddf1ac868",
    "detail-type": "ResourceGroups Group State Change",
    "source": "aws.resource-groups",
    "account": "123456789012",
    "time": "2020-09-29T09:59:01Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:resource-groups:us-east-1:123456789012:group/my-service"
    ],
    "detail": {
        "event-sequence": 4.0,
        "state-change": "delete",
        "group": {
            "arn": "arn:aws:resource-groups:us-east-1:123456789012:group/my-service",
            "name": "my-service",
            "unique-id": "3dd07ab7-3228-4410-8cdc-6c4a10fcceea"
        }
    }
}
```

Group membership change

`"detail-type": "ResourceGroups Group Membership Change"`

This `detail-type` value indicates that the group's membership was changed by a resource being added to or removed from the group. When this `detail-type` is specified, the top-level `resources` field includes the ARN of the group whose membership was changed and the ARNs of any resources that were added to or removed from the group.

The information included in the `details` section when this `detail-type` is specified include the fields described in the following table.

Field name	Type	Description
event-sequence	Double	A monotonically increasing number that indicates the sequence of events for a specific group. The number resets when the group is deleted and its unique ID changes.
group	Group JSON object	Identifies the group object associated with the event by its ARN, name, and unique ID.
resources	Array of ResourceChange JSON objects	An array of resources whose group membership has changed. This ResourceChange object contains the following fields for each resource: • membership-change – The value is either "add" or "remove". • arn – The ARN of the resource added or removed. • resource-type – The type of resource added or removed.

The following code example illustrates the contents of the event for a typical membership change type. This example shows one resource being added to the group, and one resource being removed from the group.

```
{
    "version": "0",
    "id": "08f00e24-2e30-ec44-b824-8acddf1ac868",
    "detail-type": "ResourceGroups Group Membership Change",
    "source": "aws.resource-groups",
    "account": "123456789012",
    "time": "2020-09-29T09:59:01Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:resource-groups:us-east-1:123456789012:group/my-service",
        "arn:aws:ec2:us-east-1:123456789012:instance/i-abcd1111",
```

```
        "arn:aws:ec2:us-east-1:123456789012:instance/i-efef2222"
    ],
    "detail": {
        "event-sequence": 2.0,
        "group": {
            "arn": "arn:aws:resource-groups:us-east-1:123456789012:group/my-service",
            "name": "my-service",
            "unique-id": "3dd07ab7-3228-4410-8cdc-6c4a10fcceea"
        },
        "resources": [
            {
                "membership-change": "add",
                "arn": "arn:aws:ec2:us-east-1:123456789012:instance/i-abcd1111",
                "resource-type": "AWS::EC2::Instance"
            },
            {
                "membership-change": "remove",
                "arn": "arn:aws:ec2:us-east-1:123456789012:instance/i-efef2222",
                "resource-type": "AWS::EC2::Instance"
            }
        ]
    }
}
```

Example EventBridge custom event patterns for different use cases

The following example EventBridge custom event patterns filter the events generated by Resource Groups to only those that you are interested in for a specific event rule and target.

In the following code examples, if a specific group or resource is needed, replace each *`user input placeholder`* with your own information.

All Resource Groups events

```
{
    "source": [ "aws.resource-groups" ]
}
```

Group state or membership change events

The following code example is for all group *state* changes.

```
{
```

```
    "source": [ "aws.resource-groups" ],
    "detail-type": [ "ResourceGroups Group State Change " ]
}
```

The following code example is for all group *membership* changes.

```
{
    "source": [ "aws.resource-groups" ],
    "detail-type": [ "ResourceGroups Group Membership Change" ]
}
```

Events for a specific group

```
{
    "source": [ "aws.resource-groups" ],
    "detail": {
        "group": {
            "arn": [ "my-group-arn" ]
        }
    }
}
```

The previous example captures changes to the specified group. The following example does the same and also captures changes when the group is a member resource of another group.

```
{
    "source": [ "aws.resource-groups" ],
    "resources": [ "my-group-arn" ]
}
```

Events for a specific resource

You can filter only group membership change events for specific member resources.

```
{
    "source": [ "aws.resource-groups" ],
    "detail-type": [ "ResourceGroups Group Membership Change " ],
    "resources": [ "arn:aws:ec2:us-east-1:123456789012:instance/i-b188560f" ]
}
```

Events for a specific resource type

You can use prefix matching with ARNs to match events for a specific resource type.

```
{
    "source": [ "aws.resource-groups" ],
    "resources": [
        { "prefix": "arn:aws:ec2:us-east-1:123456789012:instance" }
    ]
}
```

Alternatively, you can use exact matching by using `resource-type` identifiers, potentially matching on more than one type concisely. Unlike the previous example, the following example matches only group membership change events because group state change events don't include a `resources` field in their `detail` field.

```
{
    "source": [ "aws.resource-groups" ],
    "detail": {
        "resources": {
                "resource-type": [ "AWS::EC2::Instance", "AWS::EC2::Volume" ]
         }
    }
}
```

All resource removal events

```
{
    "source": [ "aws.resource-groups" ],
    "detail-type": [ "ResourceGroups Group Membership Change" ],
    "detail": {
        "resources": {
                "membership-change": [ "remove" ]
        }
    }
}
```

All resource removal events for a specific resource

```
 {
    "source": [ "aws.resource-groups" ],
```

```
    "detail-type": [ "ResourceGroups Group Membership Change" ],
    "detail": {
        "resources": {
                "membership-change": [ "remove" ],
                "arn": [ "arn:aws:ec2:us-east-1:123456789012:instance/i-b188560f" ]
         }
    }
}
```

You can't use the **top-level** `resources` array that was used in the first example in this section for this type of event filtering. That's because a resource in the top-level `resources` element might be a resource being added to a group and the event would still match. In other words, the following code example might return unexpected events. Instead, use the syntax shown in the previous example.

```
{
    "source": [ "aws.resource-groups" ],
    "detail-type": [ "ResourceGroups Group Membership Change" ],
    "resources": [ "arn:aws:ec2:us-east-1:123456789012:instance/i-b188560f" ],
    "detail": {
        "resources": {
                "membership-change": [ "remove" ]
         }
     }
}
```

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. In the navigation pane, choose **Saved Resource Groups**.
3. Choose the name of the resource group that you want to delete, and then choose **View details**.
4. On the group's detail page, choose **Delete** in the top right corner.
5. 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
```

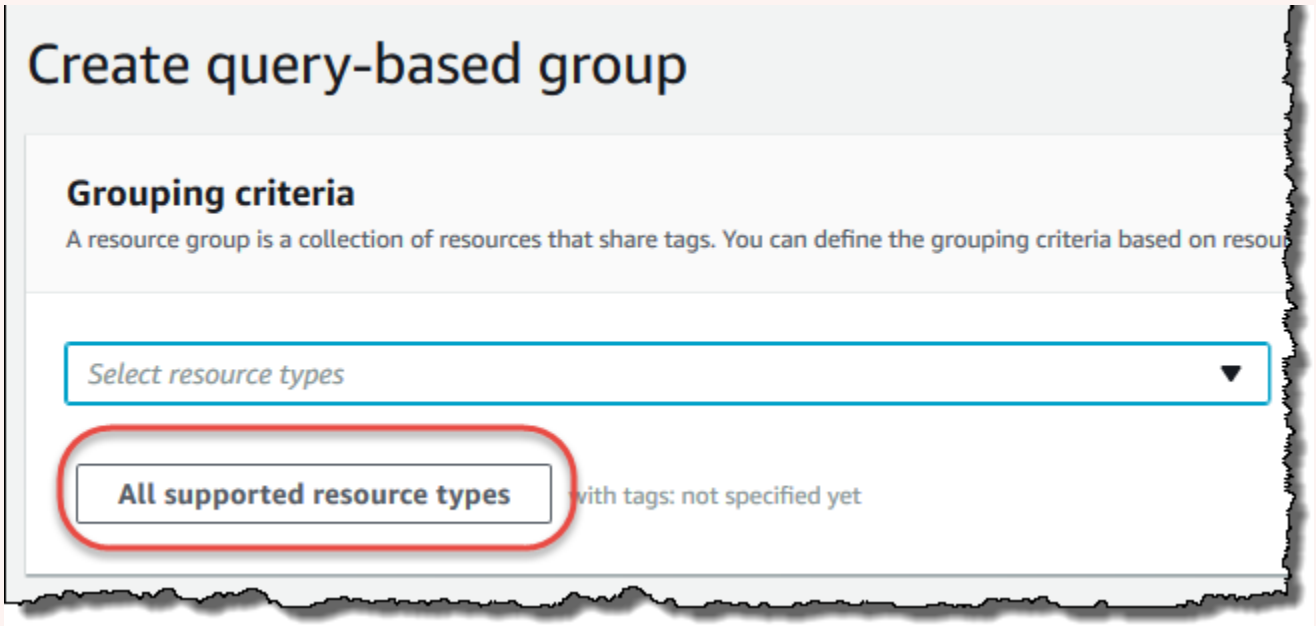
2. When you are prompted to confirm the deletion, type `yes`, and then press **Enter**.

Resource types 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 resource types that you can include in resource groups by using AWS Resource Groups, and the resource types 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. 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 at the end of this topic.

> **Note**
>
> Resource Groups and Tag Editor support the resource types in the following table, but some resource types may not be available in your AWS Region.

AWS DeepComposer

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
`AWS::DeepComposer::Composition`	✗ No	✓ Yes	✗ No
`AWS::DeepComposer::Model`	✗ No	✓ Yes	✗ No

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	✓ Yes	✓ Yes
AWS::ApiGateway::ClientCertificate	✗ No	✓ Yes	✗ No
AWS::ApiGateway::DomainName	✗ No	✗ No	✓ Yes
AWS::ApiGateway::RestApi	✗ No	✓ Yes	✓ Yes
AWS::ApiGateway::Stage	✗ No	✓ Yes	✗ No
AWS::ApiGateway::UsagePlan	✗ No	✓ Yes	✓ Yes

Amazon API Gateway V2

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::ApiGatewayV2::Api	✗ No	✓ Yes	✗ No

IAM Access Analyzer

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::AccessAnalyzer::Analyzer	✗ No	✓ Yes	✗ No

AWS Amplify

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Amplify::App	✗ No	✓ Yes	✗ No

AWS App Runner

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::AppRunner::AutoScalingConfiguration	✗ No	✓ Yes	✗ No
AWS::AppRunner::Connection	✗ No	✓ Yes	✗ No

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::AppRunner::ObservabilityConfiguration	✗ No	✓ Yes	✗ No
AWS::AppRunner::Service	✗ No	✓ Yes	✗ No
AWS::AppRunner::VpcConnector	✗ No	✓ Yes	✗ No
AWS::AppRunner::VpcIngressConnection	✗ No	✓ Yes	✗ No

AWS AppConfig

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::AppConfig::Application	✗ No	✓ Yes	✗ No
AWS::AppConfig::ConfigurationProfile	✗ No	✓ Yes	✗ No
AWS::AppConfig::Deployment	✗ No	✓ Yes	✗ No
AWS::AppConfig::DeploymentStrategy	✗ No	✓ Yes	✗ No
AWS::AppConfig::Extension	✗ No	✓ Yes	✗ No
AWS::AppConfig::ExtensionAssociation	✗ No	✓ Yes	✗ No

AWS AppFabric

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::AppFabric::AppAuthorization	✗ No	✓ Yes	✗ No
AWS::AppFabric::AppBundle	✗ No	✓ Yes	✗ No
AWS::AppFabric::Ingestion	✗ No	✓ Yes	✗ No

Amazon AppFlow

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::AppFlow::Connector	✗ No	✓ Yes	✗ No
AWS::AppFlow::Flow	✗ No	✓ Yes	✗ No

AppIntegrations

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

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::AppIntegrations::DataIntegration	✗ No	✓ Yes	✗ No
AWS::AppIntegrations::EventIntegration	✗ No	✓ Yes	✗ No

AWS App Mesh

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::AppMesh::GatewayRoute	✗ No	✓ Yes	✗ No
AWS::AppMesh::Mesh	✗ No	✓ Yes	✗ No
AWS::AppMesh::Route	✗ No	✓ Yes	✗ No
AWS::AppMesh::VirtualGateway	✗ No	✓ Yes	✗ No
AWS::AppMesh::VirtualNode	✗ No	✓ Yes	✗ No
AWS::AppMesh::VirtualRouter	✗ No	✓ Yes	✗ No
AWS::AppMesh::VirtualService	✗ No	✓ Yes	✗ No

Amazon AppStream

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::AppStream::AppBlock	✗ No	✓ Yes	✗ No
AWS::AppStream::AppBlockBuilder	✗ No	✓ Yes	✗ No
AWS::AppStream::Application	✗ No	✓ Yes	✗ No
AWS::AppStream::Fleet	✓ Yes	✓ Yes	✓ Yes
AWS::AppStream::Image	✗ No	✓ Yes	✗ No
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::Api	✗ No	✓ Yes	✗ No
AWS::AppSync::DataSource	✗ No	✗ No	✓ Yes
AWS::AppSync::DomainName	✗ No	✓ Yes	✗ No
AWS::AppSync::GraphQLApi	✗ No	✗ No	✓ Yes

Application Auto Scaling

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::ApplicationAutoScaling::ScalableTarget	✗ No	✓ Yes	✗ No

AWS Application Migration Service

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::MGN::Application	✗ No	✓ Yes	✗ No
AWS::MGN::Connector	✗ No	✓ Yes	✗ No
AWS::MGN::Job	✗ No	✓ Yes	✗ No
AWS::MGN::LaunchConfigurationTemplate	✗ No	✓ Yes	✗ No
AWS::MGN::ReplicationConfigurationTemplate	✗ No	✓ Yes	✗ No
AWS::MGN::SourceServer	✗ No	✓ Yes	✗ No
AWS::MGN::VcenterClient	✗ No	✓ Yes	✗ No
AWS::MGN::Wave	✗ No	✓ Yes	✗ No

Artificial intelligence operations (AIOps)

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::AIOps::InvestigationGroup	✗ No	✓ Yes	✗ No

Amazon Athena

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Athena::CapacityReservation	✗ No	✓ Yes	✗ No
AWS::Athena::DataCatalog	✗ No	✓ Yes	✗ No
AWS::Athena::WorkGroup	✗ No	✓ Yes	✗ No

AWS Audit Manager

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::AuditManager::Assessment	✗ No	✓ Yes	✗ No

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::AuditManager::AssessmentFramework	✗ No	✓ Yes	✗ No
AWS::AuditManager::Control	✗ No	✓ Yes	✗ No

AWS B2B Data Interchange

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::B2BI::Capability	✗ No	✓ Yes	✗ No
AWS::B2BI::Partnership	✗ No	✓ Yes	✗ No
AWS::B2BI::Profile	✗ No	✓ Yes	✗ No
AWS::B2BI::Transformer	✗ No	✓ Yes	✗ No

AWS Backup

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Backup::BackupPlan	✗ No	✓ Yes	✗ No
AWS::Backup::BackupVault	✗ No	✓ Yes	✗ No
AWS::Backup::Framework	✗ No	✓ Yes	✗ No
AWS::Backup::LegalHold	✗ No	✓ Yes	✗ No
AWS::Backup::ReportPlan	✗ No	✓ Yes	✗ No
AWS::Backup::RestoreTestingPlan	✗ No	✓ Yes	✗ No

AWS Backup gateway

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::BackupGateway::VirtualMachine	✗ No	✓ Yes	✗ No

AWS Backup search

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::BackupSearch::SearchExportJob	✗ No	✓ Yes	✗ No
AWS::BackupSearch::SearchJob	✗ No	✓ Yes	✗ No

AWS Batch

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Batch::ComputeEnvironment	✗ No	✓ Yes	✗ No
AWS::Batch::ConsumableResource	✗ No	✓ Yes	✗ No
AWS::Batch::Job	✗ No	✓ Yes	✗ No
AWS::Batch::JobDefinition	✗ No	✓ Yes	✗ No
AWS::Batch::JobQueue	✗ No	✓ Yes	✗ No
AWS::Batch::SchedulingPolicy	✗ No	✓ Yes	✗ No

Amazon Bedrock

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Bedrock::Agent	✗ No	✓ Yes	✗ No
AWS::Bedrock::AgentAlias	✗ No	✓ Yes	✗ No
AWS::Bedrock::ApplicationInferenceProfile	✗ No	✓ Yes	✗ No
AWS::Bedrock::AsyncInvoke	✗ No	✓ Yes	✗ No
AWS::Bedrock::CustomModel	✗ No	✓ Yes	✗ No
AWS::Bedrock::EvaluationJob	✗ No	✓ Yes	✗ No
AWS::Bedrock::Flow	✗ No	✓ Yes	✗ No
AWS::Bedrock::FlowAlias	✗ No	✓ Yes	✗ No
AWS::Bedrock::Guardrail	✗ No	✓ Yes	✗ No
AWS::Bedrock::KnowledgeBase	✗ No	✓ Yes	✗ No
AWS::Bedrock::ModelCustomizationJob	✗ No	✓ Yes	✗ No
AWS::Bedrock::ModelEvaluationJob	✗ No	✓ Yes	✗ No
AWS::Bedrock::ModelImportJob	✗ No	✓ Yes	✗ No
AWS::Bedrock::ModelInvocationJob	✗ No	✓ Yes	✗ No
AWS::Bedrock::PromptVersion	✗ No	✓ Yes	✗ No

AWS Billing Conductor

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::BillingConductor::BillingGroup	✗ No	✓ Yes	✓ Yes
AWS::BillingConductor::CustomLineItem	✗ No	✓ Yes	✓ Yes
AWS::BillingConductor::PricingPlan	✗ No	✓ Yes	✓ Yes
AWS::BillingConductor::PricingRule	✗ No	✓ Yes	✓ Yes

AWS Billing and Cost Management

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Billing::BillingView	✗ No	✓ Yes	✗ No

Amazon Braket

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

AWS Budgets

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Budgets::Budget	✗ No	✓ Yes	✗ No
AWS::Budgets::BudgetsAction	✗ No	✓ Yes	✗ No

AWS BugBust

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::BugBust::Event	✗ No	✓ 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

Amazon Q Developer in chat applications

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Chatbot::ChatbotConfiguration	✗ No	✓ Yes	✗ No
AWS::Chatbot::CustomAction	✗ No	✓ Yes	✗ No

Amazon Chime

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Chime::AppInstance	✗ No	✓ Yes	✗ No
AWS::Chime::AppInstanceBot	✗ No	✓ Yes	✗ No
AWS::Chime::AppInstanceUser	✗ No	✓ Yes	✗ No
AWS::Chime::Channel	✗ No	✓ Yes	✗ No
AWS::Chime::MediaInsightsPipelineConfiguration	✗ No	✓ Yes	✗ No
AWS::Chime::MediaPipeline	✗ No	✓ Yes	✗ No
AWS::Chime::MediaPipelineKinesisVideoStreamPool	✗ No	✓ Yes	✗ No
AWS::Chime::SipMediaApplication	✗ No	✓ Yes	✗ No
AWS::Chime::VoiceConnector	✗ No	✓ Yes	✗ No
AWS::Chime::VoiceProfileDomain	✗ No	✓ Yes	✗ No

AWS Clean Rooms

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::CleanRooms::AnalysisTemplate	✗ No	✓ Yes	✗ No
AWS::CleanRooms::Collaboration	✗ No	✓ Yes	✗ No
AWS::CleanRooms::ConfiguredAudienceModelAssociation	✗ No	✓ Yes	✗ No
AWS::CleanRooms::ConfiguredTable	✗ No	✓ Yes	✗ No
AWS::CleanRooms::ConfiguredTableAssociation	✗ No	✓ Yes	✗ No
AWS::CleanRooms::Membership	✗ No	✓ Yes	✗ No
AWS::CleanRooms::PrivacyBudgetTemplate	✗ No	✓ Yes	✗ No

AWS Clean Rooms ML

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::CleanRoomsML::AudienceGenerationJob	✗ No	✓ Yes	✗ No
AWS::CleanRoomsML::AudienceModel	✗ No	✓ Yes	✗ No

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::CleanRoomsML::ConfiguredAudienceModel	✗ No	✓ Yes	✗ No
AWS::CleanRoomsML::ConfiguredModelAlgorithm	✗ No	✓ Yes	✗ No
AWS::CleanRoomsML::TrainingDataset	✗ No	✓ Yes	✗ No

Amazon Cloud Directory

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::CloudDirectory::Directory	✗ 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
AWS::CloudFormation::StackSet	✗ No	✓ Yes	✗ No

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]
AWS::CloudFront::VpcOrigin	✗ No	✓ 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.

AWS CloudHSM

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

AWS Cloud Map

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
`AWS::ServiceDiscovery::Namespace`	✗ No	✓ Yes	✗ No
`AWS::ServiceDiscovery::Service`	✗ No	✓ Yes	✗ No

Amazon CloudSearch

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::CloudSearch::Domain	✗ No	✓ Yes	✗ No

AWS CloudTrail

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::CloudTrail::Channel	✗ No	✓ Yes	✗ No
AWS::CloudTrail::Dashboard	✗ No	✓ Yes	✗ No
AWS::CloudTrail::EventDataStore	✗ No	✓ Yes	✗ No
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
AWS::CloudWatch::InsightRule	✗ No	✓ Yes	✗ No
AWS::CloudWatch::MetricStream	✗ No	✓ Yes	✗ No
AWS::CloudWatch::ServiceLevelObjective	✗ No	✓ Yes	✗ No

Amazon CloudWatch Application Insights

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

CloudWatch Application Signals

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::ApplicationSignals::ServiceLevelObjective	✗ No	✓ Yes	✗ No

CloudWatch Evidently

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Evidently::Feature	✗ No	✓ Yes	✗ No
AWS::Evidently::Launch	✗ No	✓ Yes	✗ No
AWS::Evidently::Project	✗ No	✓ Yes	✗ No
AWS::Evidently::Segment	✗ No	✓ Yes	✗ No

Amazon CloudWatch Logs

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Logs::AnomalyDetector	✗ No	✓ Yes	✗ No
AWS::Logs::Delivery	✗ No	✓ Yes	✗ No
AWS::Logs::DeliveryDestination	✗ No	✓ Yes	✗ No
AWS::Logs::DeliverySource	✗ No	✓ Yes	✗ No
AWS::Logs::Destination	✗ No	✓ Yes	✗ No
AWS::Logs::LogGroup	✗ No	✓ Yes	✓ Yes

Amazon CloudWatch Observability Manager

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Oam::Link	✗ No	✓ Yes	✗ No
AWS::Oam::Sink	✗ No	✓ Yes	✗ No

Amazon CloudWatch RUM

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::RUM::AppMonitor	✗ No	✓ Yes	✗ No

Amazon CloudWatch Synthetics

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Synthetics::Canary	✗ No	✓ Yes	✓ Yes
AWS::Synthetics::Group	✗ No	✓ Yes	✗ No

AWS CodeArtifact

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

AWS CodeBuild

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

Amazon CodeCatalyst

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::CodeCatalyst::Connection	✗ No	✓ Yes	✗ No
AWS::CodeCatalyst::IdentityCenterApplication	✗ No	✓ Yes	✗ No
AWS::CodeCatalyst::Space	✗ No	✓ Yes	✗ No

AWS CodeCommit

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

AWS CodeConnections

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::CodeConnections::Host	✗ No	✓ Yes	✗ No
AWS::CodeConnections::RepositoryLink	✗ No	✓ Yes	✗ No

AWS CodeDeploy

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

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

Amazon CodeGuru Reviewer

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

Amazon CodeGuru Profiler

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::CodeGuruProfiler::ProfilingGroup	✗ No	✓ Yes	✗ No

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

AWS CodeStar Notifications

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::CodeStarNotifications::NotificationRule	✗ No	✓ Yes	✗ No

AWS CodeConnections

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::CodeStarConnections::Connection	✗ No	✓ Yes	✗ No

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::CodeStarConnections::Host	✗ No	✓ Yes	✗ No
AWS::CodeStarConnections::RepositoryLink	✗ No	✓ Yes	✗ No

Amazon CodeWhisperer

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::CodeWhisperer::Customization	✗ No	✓ Yes	✗ No
AWS::CodeWhisperer::Profile	✗ No	✓ Yes	✗ No

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::DocumentClassificationJob	✗ No	✓ Yes	✗ No
AWS::Comprehend::DocumentClassifier	✓ Yes	✓ Yes	✗ No
AWS::Comprehend::DocumentClassifierEndpoint	✗ No	✓ Yes	✗ No
AWS::Comprehend::DominantLanguageDetectionJob	✗ No	✓ Yes	✗ No
AWS::Comprehend::EntitiesDetectionJob	✗ No	✓ Yes	✗ No
AWS::Comprehend::EntityRecognizer	✓ Yes	✓ Yes	✗ No
AWS::Comprehend::EntityRecognizerEndpoint	✗ No	✓ Yes	✗ No
AWS::Comprehend::EventsDetectionJob	✗ No	✓ Yes	✗ No
AWS::Comprehend::Flywheel	✗ No	✓ Yes	✗ No
AWS::Comprehend::KeyPhrasesDetectionJob	✗ No	✓ Yes	✗ No
AWS::Comprehend::PIIEntitiesDetectionJob	✗ No	✓ Yes	✗ No
AWS::Comprehend::SentimentDetectionJob	✗ No	✓ Yes	✗ No

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

AWS Config

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Config::AggregationAuthorization	✗ No	✓ Yes	✗ No
AWS::Config::ConfigRule	✓ Yes	✓ Yes	✗ No
AWS::Config::ConfigurationAggregator	✗ No	✓ Yes	✗ No
AWS::Config::ConfigurationRecorder	✗ No	✓ Yes	✗ No
AWS::Config::ConformancePack	✗ No	✓ Yes	✗ No
AWS::Config::OrganizationConfigRule	✗ No	✓ Yes	✗ No
AWS::Config::OrganizationConformancePack	✗ No	✓ Yes	✗ No
AWS::Config::StoredQuery	✗ No	✓ Yes	✗ No

Amazon Connect

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Connect::AgentStatus	✗ No	✓ Yes	✗ No
AWS::Connect::Contact	✗ No	✓ Yes	✗ No
AWS::Connect::ContactEvaluation	✗ No	✓ Yes	✗ No
AWS::Connect::ContactFlow	✗ No	✓ Yes	✗ No
AWS::Connect::ContactFlowModule	✗ No	✓ Yes	✗ No
AWS::Connect::EvaluationForm	✗ No	✓ Yes	✗ No
AWS::Connect::HoursOfOperation	✗ No	✓ Yes	✗ No
AWS::Connect::Instance	✗ No	✓ Yes	✗ No
AWS::Connect::IntegrationAssociation	✗ No	✓ Yes	✗ No
AWS::Connect::PhoneNumber	✗ No	✓ Yes	✗ No
AWS::Connect::Prompt	✗ No	✓ Yes	✗ No
AWS::Connect::Queue	✗ No	✓ Yes	✗ No
AWS::Connect::QuickConnect	✗ No	✓ Yes	✗ No
AWS::Connect::RoutingProfile	✗ No	✓ Yes	✗ No
AWS::Connect::Rule	✗ No	✓ Yes	✗ No
AWS::Connect::SecurityProfile	✗ No	✓ Yes	✗ No

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Connect::TaskTemplate	✗ No	✓ Yes	✗ No
AWS::Connect::TrafficDistributionGroup	✗ No	✓ Yes	✗ No
AWS::Connect::UseCase	✗ No	✓ Yes	✗ No
AWS::Connect::User	✗ No	✓ Yes	✗ No
AWS::Connect::UserHierarchyGroup	✗ No	✓ Yes	✗ No
AWS::Connect::Vocabulary	✗ No	✓ Yes	✗ No

Amazon Connect Cases

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Cases::Case	✗ No	✓ Yes	✗ No
AWS::Cases::Domain	✗ No	✓ Yes	✗ No
AWS::Cases::RelatedItem	✗ No	✓ Yes	✗ No

Amazon Connect Customer Profiles

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::CustomerProfiles::Domain	✗ No	✓ Yes	✗ No
AWS::CustomerProfiles::Integration	✗ No	✓ Yes	✗ No
AWS::CustomerProfiles::ObjectType	✗ No	✓ Yes	✗ No

Amazon Connect Outbound Campaigns

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::ConnectCampaigns::Campaign	✗ No	✓ Yes	✗ No

Amazon Connect Voice ID

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::VoiceID::Domain	✗ No	✓ Yes	✗ No

Amazon Connect Wisdom

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Wisdom::AIAgent	✗ No	✓ Yes	✗ No
AWS::Wisdom::AIGuardrail	✗ No	✓ Yes	✗ No
AWS::Wisdom::AIPrompt	✗ No	✓ Yes	✗ No
AWS::Wisdom::Assistant	✗ No	✓ Yes	✓ Yes
AWS::Wisdom::AssistantAssociation	✗ No	✓ Yes	✓ Yes
AWS::Wisdom::Content	✗ No	✓ Yes	✗ No
AWS::Wisdom::ContentAssociation	✗ No	✓ Yes	✗ No
AWS::Wisdom::KnowledgeBase	✗ No	✓ Yes	✓ Yes
AWS::Wisdom::MessageTemplate	✗ No	✓ Yes	✗ No
AWS::Wisdom::QuickResponse	✗ No	✓ Yes	✗ No
AWS::Wisdom::Session	✗ No	✓ Yes	✗ No

AWS Control Tower

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::ControlTower::EnabledBaseline	✗ No	✓ Yes	✗ No
AWS::ControlTower::EnabledControl	✗ No	✓ Yes	✗ No
AWS::ControlTower::LandingZone	✗ No	✓ Yes	✗ No

AWS Cost Explorer

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::CE::AnomalyMonitor	✗ No	✓ Yes	✗ No
AWS::CE::AnomalySubscription	✗ No	✓ Yes	✗ No
AWS::CE::CostCategory	✗ No	✓ Yes	✗ No

AWS Cost and Usage Report

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::CUR::ReportDefinition	✗ No	✓ Yes	✗ No

AWS Data Exchange

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

AWS Data Exports

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::BCMDataExports::Export	✗ No	✓ Yes	✗ No

Amazon Data Lifecycle Manager

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::DLM::LifecyclePolicy	✗ 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 DataSync

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::DataSync::Agent	✗ No	✓ Yes	✗ No
AWS::DataSync::DiscoveryJob	✗ No	✓ Yes	✗ No
AWS::DataSync::Location	✗ No	✓ Yes	✗ No
AWS::DataSync::StorageSystem	✗ No	✓ Yes	✗ No

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::DataSync::Task	✗ No	✓ Yes	✗ No
AWS::DataSync::TaskExecution	✗ No	✓ Yes	✗ No

Amazon DataZone

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

AWS Database Migration Service

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::DMS::Certificate	✓ Yes	✓ Yes	✗ No
AWS::DMS::DataMigration	✗ No	✓ Yes	✗ No
AWS::DMS::DataProvider	✗ No	✓ Yes	✗ No

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::DMS::Endpoint	✓ Yes	✓ Yes	✓ Yes
AWS::DMS::EventSubscription	✓ Yes	✓ Yes	× No
AWS::DMS::InstanceProfile	× No	✓ Yes	× No
AWS::DMS::MigrationProject	× No	✓ Yes	× No
AWS::DMS::ReplicationConfig	× No	✓ Yes	× No
AWS::DMS::ReplicationInstance	✓ Yes	✓ Yes	✓ Yes
AWS::DMS::ReplicationSubnetGroup	✓ Yes	✓ Yes	× No
AWS::DMS::ReplicationTask	✓ Yes	✓ Yes	× No
AWS::DMS::ReplicationTaskAssessmentRun	× No	✓ Yes	× No

AWS Deadline Cloud

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Deadline::Farm	× No	✓ Yes	× No
AWS::Deadline::LicenseEndpoint	× No	✓ Yes	× No

Amazon Detective

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Detective::Graph	✗ No	✓ Yes	✗ No

AWS Device Farm

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::DeviceFarm::Device	✗ No	✓ Yes	✗ No
AWS::DeviceFarm::DeviceInstance	✗ No	✓ Yes	✗ No
AWS::DeviceFarm::InstanceProfile	✗ No	✓ Yes	✗ No
AWS::DeviceFarm::Project	✗ No	✓ Yes	✗ No
AWS::DeviceFarm::TestGridProject	✗ No	✓ Yes	✗ No
AWS::DeviceFarm::VPCEConfiguration	✗ No	✓ Yes	✗ No

AWS Diode Messaging

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::DiodeMessaging::AccountMapping	✗ No	✓ Yes	✗ No
AWS::DiodeMessaging::RequestingFlow	✗ No	✓ Yes	✗ No
AWS::DiodeMessaging::RespondingFlow	✗ No	✓ Yes	✗ No

AWS Diode Object Transfer

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Diode::AccountMapping	✗ No	✓ Yes	✗ No
AWS::Diode::Transfer	✗ No	✓ Yes	✗ No

AWS Direct Connect

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::DirectConnect::Connection	✗ No	✓ Yes	✗ No

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::DirectConnect::Gateway	✗ No	✓ Yes	✗ No
AWS::DirectConnect::Lag	✗ No	✓ Yes	✗ No
AWS::DirectConnect::VirtualInterface	✗ No	✓ Yes	✗ No

AWS Directory Service

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::DirectoryService::Directory	✗ No	✓ Yes	✗ No

Amazon DocumentDB Elastic Clusters

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::DocDBElastic::ClusterSnapshot	✗ No	✓ Yes	✗ No

Amazon DynamoDB

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

DynamoDB Accelerator

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::DAX::Cluster	× No	✓ Yes	× No

Amazon EMR

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::EMR::Cluster	✓ Yes	✓ Yes	✓ Yes
AWS::EMR::Editor	× No	✓ Yes	× No
AWS::EMR::NotebookExecution	× No	✓ Yes	× No
AWS::EMR::Studio	× No	✓ Yes	× No

Amazon EMR Containers

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

Amazon EMR Serverless

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

Amazon ElastiCache

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::ElastiCache::CacheCluster	✓ Yes	✓ Yes	✓ Yes
AWS::ElastiCache::ParameterGroup	× No	✓ Yes	× No
AWS::ElastiCache::ReplicationGroup	× No	✓ Yes	× No
AWS::ElastiCache::ReservedInstance	× No	✓ Yes	× No
AWS::ElastiCache::SecurityGroup	× No	✓ Yes	× No
AWS::ElastiCache::ServerlessCache	× No	✓ Yes	× No
AWS::ElastiCache::ServerlessCacheSnapshot	× No	✓ Yes	× No
AWS::ElastiCache::Snapshot	✓ Yes	✓ Yes	× No
AWS::ElastiCache::SubnetGroup	× No	✓ Yes	× No
AWS::ElastiCache::User	× No	✓ Yes	× No
AWS::ElastiCache::UserGroup	× No	✓ 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::CapacityReservationFleet	✗ No	✓ Yes	✗ No
AWS::EC2::CarrierGateway	✗ No	✓ Yes	✗ No
AWS::EC2::ClientVpnEndpoint	✗ No	✓ Yes	✗ No
AWS::EC2::CoipPool	✗ No	✓ Yes	✗ No
AWS::EC2::CustomerGateway	✓ Yes	✓ Yes	✓ Yes

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::EC2::DHCPOptions	✓ Yes	✓ Yes	✓ Yes
AWS::EC2::EC2Fleet	× No	✓ Yes	× No
AWS::EC2::EgressOnlyInternetGateway	× No	✓ Yes	× No
AWS::EC2::EIP	✓ Yes	✓ Yes	× No
AWS::EC2::ElasticGpu	× No	✓ Yes	× No
AWS::EC2::ExportImageTask	× No	✓ Yes	× No
AWS::EC2::ExportInstanceTask	× No	✓ Yes	× No
AWS::EC2::FlowLog	× No	✓ Yes	× No
AWS::EC2::FpgaImage	× No	✓ Yes	× No
AWS::EC2::Host	× No	✓ Yes	× No
AWS::EC2::HostReservation	× No	✓ Yes	× No
AWS::EC2::Image	✓ Yes	✓ Yes	× No
AWS::EC2::ImportImageTask	× No	✓ Yes	× No
AWS::EC2::ImportSnapshotTask	× No	✓ Yes	× No
AWS::EC2::Instance	✓ Yes	✓ Yes	✓ Yes
AWS::EC2::InstanceConnectEndpoint	× No	✓ Yes	× No
AWS::EC2::InstanceEventWindow	× No	✓ Yes	× No
AWS::EC2::InternetGateway	✓ Yes	✓ Yes	✓ Yes

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::EC2::IPv4Pool	✗ No	✓ Yes	✗ No
AWS::EC2::IPv6Pool	✗ No	✓ Yes	✗ No
AWS::EC2::KeyPair	✗ No	✓ Yes	✗ No
AWS::EC2::LaunchTemplate	✗ No	✓ Yes	✓ Yes
AWS::EC2::LocalGateway	✗ No	✓ Yes	✗ No
AWS::EC2::LocalGatewayRouteTable	✗ No	✓ Yes	✗ No
AWS::EC2::LocalGatewayRouteTableVirtualInterfaceGroupAssociation	✗ No	✓ Yes	✗ No
AWS::EC2::LocalGatewayRouteTableVPCAssociation	✗ No	✓ Yes	✗ No
AWS::EC2::LocalGatewayVirtualInterface	✗ No	✓ Yes	✗ No
AWS::EC2::LocalGatewayVirtualInterfaceGroup	✗ No	✓ Yes	✗ No
AWS::EC2::NatGateway	✓ Yes	✓ Yes	✓ Yes
AWS::EC2::NetworkAcl	✓ Yes	✓ Yes	✓ Yes
AWS::EC2::NetworkInsightsAccessScope	✗ No	✓ Yes	✗ No
AWS::EC2::NetworkInsightsAccessScopeAnalysis	✗ No	✓ Yes	✗ No
AWS::EC2::NetworkInsightsAnalysis	✗ No	✓ Yes	✗ No

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::EC2::NetworkInsightsPath	✗ No	✓ Yes	✗ No
AWS::EC2::NetworkInterface	✓ Yes	✓ Yes	✓ Yes
AWS::EC2::PlacementGroup	✗ No	✓ Yes	✓ Yes
AWS::EC2::PrefixList	✗ No	✓ Yes	✗ No
AWS::EC2::ReplaceRootVolumeTask	✗ No	✓ Yes	✗ No
AWS::EC2::ReservedInstance	✓ Yes	✓ Yes	✗ No
AWS::EC2::RouteTable	✓ Yes	✓ Yes	✓ Yes
AWS::EC2::SecurityGroup	✓ Yes	✓ Yes	✓ Yes
AWS::EC2::SecurityGroupRule	✗ No	✓ Yes	✗ No
AWS::EC2::Snapshot	✓ Yes	✓ Yes	✗ No
AWS::EC2::SpotFleet	✗ No	✓ Yes	✗ No
AWS::EC2::SpotInstanceRequest	✓ Yes	✓ Yes	✗ No
AWS::EC2::Subnet	✓ Yes	✓ Yes	✓ Yes
AWS::EC2::SubnetCidrReservation	✗ No	✓ Yes	✗ No
AWS::EC2::TrafficMirrorFilter	✗ No	✓ Yes	✗ No
AWS::EC2::TrafficMirrorFilterRule	✗ No	✓ Yes	✗ No
AWS::EC2::TrafficMirrorSession	✗ No	✓ Yes	✗ No
AWS::EC2::TrafficMirrorTarget	✗ No	✓ Yes	✗ No

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::EC2::TransitGateway	✗ No	✓ Yes	✗ No
AWS::EC2::TransitGatewayAttachment	✗ No	✓ Yes	✗ No
AWS::EC2::TransitGatewayConnectPeer	✗ No	✓ Yes	✗ No
AWS::EC2::TransitGatewayMulticastDomain	✗ No	✓ Yes	✗ No
AWS::EC2::TransitGatewayPolicyTable	✗ No	✓ Yes	✗ No
AWS::EC2::TransitGatewayRouteTable	✗ No	✓ Yes	✗ No
AWS::EC2::TransitGatewayRouteTableAnnouncement	✗ No	✓ Yes	✗ No
AWS::EC2::VerifiedAccessEndpoint	✗ No	✓ Yes	✗ No
AWS::EC2::VerifiedAccessGroup	✗ No	✓ Yes	✗ No
AWS::EC2::VerifiedAccessInstance	✗ No	✓ Yes	✗ No
AWS::EC2::VerifiedAccessTrustProvider	✗ No	✓ Yes	✗ No
AWS::EC2::Volume	✓ Yes	✓ Yes	✓ Yes
AWS::EC2::VPC	✓ Yes	✓ Yes	✓ Yes
AWS::EC2::VPCBlockPublicAccessExclusion	✗ No	✓ Yes	✗ No
AWS::EC2::VPCEndpoint	✗ No	✓ Yes	✗ No

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::EC2::VPCEndpointConnection	✗ No	✓ Yes	✗ No
AWS::EC2::VPCEndpointService	✗ No	✓ Yes	✗ No
AWS::EC2::VPCEndpointServicePermissions	✗ No	✓ Yes	✗ No
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::CapacityProvider	✗ No	✓ Yes	✗ No
AWS::ECS::Cluster	✓ Yes	✓ Yes	✗ No
AWS::ECS::ContainerInstance	✗ No	✓ Yes	✗ No
AWS::ECS::Service	✗ No	✓ Yes	✗ No
AWS::ECS::Task	✗ No	✓ Yes	✗ No
AWS::ECS::TaskDefinition	✓ Yes	✓ Yes	✗ No
AWS::ECS::TaskSet	✗ No	✓ Yes	✗ No

AWS Elastic Disaster Recovery

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::DRS::Job	✗ No	✓ Yes	✗ No
AWS::DRS::RecoveryInstance	✗ No	✓ Yes	✗ No
AWS::DRS::ReplicationConfigurationTemplate	✗ No	✓ Yes	✗ No

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::DRS::SourceNetwork	✗ No	✓ Yes	✗ No
AWS::DRS::SourceServer	✗ No	✓ Yes	✗ No

Amazon Elastic File System

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

Amazon Elastic Kubernetes Service (Amazon EKS)

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

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::EKS::FargateProfile	✗ No	✓ Yes	✗ No
AWS::EKS::IdentityProviderConfig	✗ No	✓ Yes	✗ No
AWS::EKS::Nodegroup	✗ No	✓ Yes	✗ No
AWS::EKS::PodIdentityAssociation	✗ No	✓ Yes	✗ No

Elastic Load Balancing

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::ElasticLoadBalancing::LoadBalancer	✓ Yes	✓ Yes	✓ Yes
AWS::ElasticLoadBalancingV2::Listener	✗ No	✓ Yes	✓ Yes
AWS::ElasticLoadBalancingV2::ListenerRule	✗ No	✓ Yes	✓ Yes
AWS::ElasticLoadBalancingV2::LoadBalancer	✓ Yes	✓ Yes	✓ Yes
AWS::ElasticLoadBalancingV2::TargetGroup	✓ Yes	✓ Yes	✓ Yes

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::ElasticLoadBalancingV2::TrustStore	✗ No	✓ Yes	✗ No

Amazon OpenSearch Service

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

AWS Elemental MediaLive

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::MediaLive::Channel	✗ No	✓ Yes	✗ No
AWS::MediaLive::ChannelPlacementGroup	✗ No	✓ Yes	✗ No
AWS::MediaLive::CloudWatchAlarmTemplate	✗ No	✓ Yes	✗ No

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::MediaLive::CloudWatchAlarmTemplateGroup	✗ No	✓ Yes	✗ No
AWS::MediaLive::EventBridgeRuleTemplate	✗ No	✓ Yes	✗ No
AWS::MediaLive::EventBridgeRuleTemplateGroup	✗ No	✓ Yes	✗ No
AWS::MediaLive::Input	✗ No	✓ Yes	✗ No
AWS::MediaLive::InputDevice	✗ No	✓ Yes	✗ No
AWS::MediaLive::InputSecurityGroup	✗ No	✓ Yes	✗ No
AWS::MediaLive::Multiplex	✗ No	✓ Yes	✗ No
AWS::MediaLive::Network	✗ No	✓ Yes	✗ No
AWS::MediaLive::Node	✗ No	✓ Yes	✗ No
AWS::MediaLive::Reservation	✗ No	✓ Yes	✗ No
AWS::MediaLive::SignalMap	✗ No	✓ Yes	✗ No

AWS Elemental MediaConvert

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::MediaConvert::Job	✗ No	✓ Yes	✗ No
AWS::MediaConvert::JobTemplate	✗ No	✓ Yes	✗ No
AWS::MediaConvert::Preset	✗ No	✓ Yes	✗ No
AWS::MediaConvert::Queue	✗ No	✓ Yes	✗ No

AWS Elemental MediaPackage V2

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::MediaPackageV2::Channel	✗ No	✓ Yes	✗ No
AWS::MediaPackageV2::ChannelGroup	✗ No	✓ Yes	✗ No
AWS::MediaPackageV2::OriginEndpoint	✗ No	✓ Yes	✗ No

AWS Elemental MediaStore

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::MediaStore::Container	✗ No	✓ Yes	✗ No

MediaTailor

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::MediaTailor::Channel	✗ No	✓ Yes	✗ No
AWS::MediaTailor::LiveSource	✗ No	✓ Yes	✗ No
AWS::MediaTailor::PlaybackConfiguration	✗ No	✓ Yes	✗ No
AWS::MediaTailor::SourceLocation	✗ No	✓ Yes	✗ No
AWS::MediaTailor::VodSource	✗ No	✓ Yes	✗ No

AWS Elemental Support Cases

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::ElementalSupportCases::Case	✗ No	✓ Yes	✗ No

AWS End User Messaging Social

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::SocialMessaging::WhatsAppBusinessAccount	✗ No	✓ Yes	✗ No

AWS Entity Resolution

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::EntityResolution::IdMappingWorkflow	✗ No	✓ Yes	✗ No
AWS::EntityResolution::IdNamespace	✗ No	✓ Yes	✗ No

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::EntityResolution::MatchingWorkflow	✗ No	✓ Yes	✗ No
AWS::EntityResolution::SchemaMapping	✗ No	✓ Yes	✗ No

Amazon CloudWatch Events

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Events::EventBus	✗ No	✓ Yes	✗ No
AWS::Events::Rule	✓ Yes	✓ Yes	✓ Yes

Note

Rules in custom event buses aren't supported in Tag Editor.

Amazon EventBridge Pipes

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Pipes::Pipe	✗ No	✓ Yes	✗ No

Amazon EventBridge Scheduler

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Scheduler::ScheduleGroup	✗ No	✓ Yes	✗ No

Amazon EventBridge Schemas

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::EventSchemas::Discoverer	✗ No	✓ Yes	✗ No
AWS::EventSchemas::Registry	✗ No	✓ Yes	✗ No
AWS::EventSchemas::Schema	✗ No	✓ Yes	✗ No

Amazon FSx

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::FSx::Backup	✗ No	✓ Yes	✗ No
AWS::FSx::DataRepositoryTask	✗ No	✓ Yes	✗ No
AWS::FSx::FileCache	✗ No	✓ Yes	✗ No
AWS::FSx::FileSystem	✓ Yes	✓ Yes	✗ No
AWS::FSx::Snapshot	✗ No	✓ Yes	✗ No
AWS::FSx::StorageVirtualMachine	✗ No	✓ Yes	✗ No
AWS::FSx::Volume	✗ No	✓ Yes	✗ No

AWS Fault Injection Service

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::FIS::Experiment	✗ No	✓ Yes	✗ No
AWS::FIS::ExperimentTemplate	✗ No	✓ Yes	✗ No

Amazon FinSpace schemas

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::FinSpace::Environment	✗ No	✓ Yes	✗ No
AWS::FinSpace::KxCluster	✗ No	✓ Yes	✗ No
AWS::FinSpace::KxDatabase	✗ No	✓ Yes	✗ No
AWS::FinSpace::KxDataview	✗ No	✓ Yes	✗ No
AWS::FinSpace::KxEnvironment	✗ No	✓ Yes	✗ No
AWS::FinSpace::KxScalingGroup	✗ No	✓ Yes	✗ No
AWS::FinSpace::KxUser	✗ No	✓ Yes	✗ No
AWS::FinSpace::KxVolume	✗ No	✓ Yes	✗ No

AWS Firewall Manager

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::FMS::Applicationslist	✗ No	✓ Yes	✗ No
AWS::FMS::Policy	✗ No	✓ Yes	✗ No
AWS::FMS::ProtocolsList	✗ No	✓ Yes	✗ No

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::FMS::ResourceSet	✗ No	✓ Yes	✗ No

AWS IoT Fleet Hub

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

Amazon Forecast

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

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Forecast::Forecast	✓ Yes	✓ Yes	× No
AWS::Forecast::ForecastEndpoint	× No	✓ Yes	× No
AWS::Forecast::ForecastExportJob	✓ Yes	✓ Yes	× No
AWS::Forecast::Predictor	✓ Yes	✓ Yes	× No
AWS::Forecast::PredictorBacktestExportJob	✓ Yes	✓ Yes	× No
AWS::Forecast::WhatIfAnalysis	× No	✓ Yes	× No

Amazon Fraud Detector

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::FraudDetector::BatchImport	× No	✓ Yes	× No
AWS::FraudDetector::BatchPrediction	× No	✓ Yes	× No
AWS::FraudDetector::Detector	✓ Yes	✓ Yes	× No
AWS::FraudDetector::DetectorVersion	× No	✓ Yes	× No
AWS::FraudDetector::EntityType	✓ Yes	✓ Yes	× No
AWS::FraudDetector::EventType	✓ Yes	✓ Yes	× No

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::FraudDetector::ExternalModel	✓ Yes	✓ Yes	✗ No
AWS::FraudDetector::Label	✓ Yes	✓ Yes	✗ No
AWS::FraudDetector::List	✗ No	✓ 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

FreeRTOS

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::FreeRTOS::Subscription	✗ No	✓ Yes	✗ No

Amazon GameLift Servers

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::GameLift::Alias	✗ No	✓ Yes	✗ No
AWS::GameLift::ContainerFleet	✗ No	✓ Yes	✗ No
AWS::GameLift::ContainerGroupDefinition	✗ No	✓ Yes	✗ No
AWS::GameLift::Fleet	✗ No	✓ Yes	✗ No
AWS::GameLift::GameServerGroup	✗ No	✓ Yes	✗ No
AWS::GameLift::GameSessionQueue	✗ No	✓ Yes	✗ No
AWS::GameLift::Location	✗ No	✓ Yes	✗ No
AWS::GameLift::MatchmakingConfiguration	✗ No	✓ Yes	✗ No
AWS::GameLift::MatchmakingRuleSet	✗ No	✓ Yes	✗ No
AWS::GameLift::Script	✗ No	✓ Yes	✗ No

AWS Global Accelerator

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::GlobalAccelerator::Accelerator	✗ No	✓ Yes	✗ No
AWS::GlobalAccelerator::CrossAccountAttachment	✗ No	✓ Yes	✗ No

AWS Glue

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Glue::Blueprint	✗ No	✓ Yes	✗ No
AWS::Glue::Catalog	✗ No	✓ Yes	✗ No
AWS::Glue::Completion	✗ No	✓ Yes	✗ No
AWS::Glue::Connection	✗ No	✓ Yes	✗ No
AWS::Glue::Crawler	✓ Yes	✓ Yes	✗ No
AWS::Glue::CustomEntityType	✗ No	✓ Yes	✗ No
AWS::Glue::Database	✗ No	✓ Yes	✓ Yes
AWS::Glue::DataQualityRuleset	✗ No	✓ Yes	✗ No

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Glue::DevEndpoint	✗ No	✓ Yes	✗ No
AWS::Glue::Job	✓ Yes	✓ Yes	✗ No
AWS::Glue::MLTransform	✗ No	✓ Yes	✗ No
AWS::Glue::Registry	✗ No	✓ Yes	✗ No
AWS::Glue::Schema	✗ No	✓ Yes	✗ No
AWS::Glue::Session	✗ No	✓ Yes	✗ No
AWS::Glue::Trigger	✓ Yes	✓ Yes	✗ No
AWS::Glue::UsageProfile	✗ No	✓ Yes	✗ No
AWS::Glue::Workflow	✗ No	✓ 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

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::DataBrew::Recipe	✓ Yes	✓ Yes	✓ Yes
AWS::DataBrew::Ruleset	✗ No	✓ Yes	✗ No
AWS::DataBrew::Schedule	✓ Yes	✓ Yes	✓ Yes

AWS Ground Station

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::GroundStation::Config	✗ No	✓ Yes	✗ No
AWS::GroundStation::Contact	✗ No	✓ Yes	✗ No
AWS::GroundStation::DataflowEndpointGroup	✗ No	✓ Yes	✗ No
AWS::GroundStation::Ephemeris	✗ No	✓ Yes	✗ No
AWS::GroundStation::MissionProfile	✗ No	✓ Yes	✗ No
AWS::GroundStation::Satellite	✗ No	✓ Yes	✗ No

Amazon GuardDuty

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::GuardDuty::Detector	✗ No	✓ Yes	✓ Yes
AWS::GuardDuty::Filter	✗ No	✓ Yes	✗ No
AWS::GuardDuty::IPSet	✗ No	✓ Yes	✗ No
AWS::GuardDuty::MalwareProtectionPlan	✗ No	✓ Yes	✗ No
AWS::GuardDuty::ThreatIntelSet	✗ No	✓ Yes	✗ No

AWS HealthImaging

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::HealthImaging::Datastore	✗ No	✓ Yes	✗ No
AWS::HealthImaging::ImageSet	✗ No	✓ Yes	✗ No

AWS HealthLake

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::HealthLake::FHIRDatastore	✗ No	✓ Yes	✗ No

AWS HealthOmics

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Omics::AnnotationStore	✗ No	✓ Yes	✗ No
AWS::Omics::AnnotationStoreVersion	✗ No	✓ Yes	✗ No
AWS::Omics::ReadSet	✗ No	✓ Yes	✗ No
AWS::Omics::Reference	✗ No	✓ Yes	✗ No
AWS::Omics::ReferenceStore	✗ No	✓ Yes	✗ No
AWS::Omics::Run	✗ No	✓ Yes	✗ No
AWS::Omics::RunCache	✗ No	✓ Yes	✗ No
AWS::Omics::RunGroup	✗ No	✓ Yes	✗ No
AWS::Omics::SequenceStore	✗ No	✓ Yes	✗ No
AWS::Omics::VariantStore	✗ No	✓ Yes	✗ No

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Omics::Workflow	✗ No	✓ Yes	✗ No

Amazon Interactive Video Service

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::IVS::Channel	✗ No	✓ Yes	✗ No
AWS::IVS::Composition	✗ No	✓ Yes	✗ No
AWS::IVS::EncoderConfiguration	✗ No	✓ Yes	✗ No
AWS::IVS::IngestConfiguration	✗ No	✓ Yes	✗ No
AWS::IVS::PlaybackKeyPair	✗ No	✓ Yes	✗ No
AWS::IVS::PlaybackRestrictionPolicy	✗ No	✓ Yes	✗ No
AWS::IVS::PublicKey	✗ No	✓ Yes	✗ No
AWS::IVS::RecordingConfiguration	✗ No	✓ Yes	✗ No
AWS::IVS::Stage	✗ No	✓ Yes	✗ No
AWS::IVS::StorageConfiguration	✗ No	✓ Yes	✗ No
AWS::IVS::StreamKey	✗ No	✓ Yes	✗ No

IAM

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::SSO::Application	✗ No	✓ Yes	✗ No
AWS::SSO::Instance	✗ No	✓ Yes	✗ No
AWS::SSO::PermissionSet	✗ No	✓ Yes	✗ No
AWS::SSO::TrustedTokenIssuer	✗ No	✓ Yes	✗ No

AWS Identity and Access Management

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::IAM::InstanceProfile	✓ Yes[1]	✓ Yes[2]	✗ No
AWS::IAM::ManagedPolicy	✓ Yes[1]	✓ Yes[2]	✗ No
AWS::IAM::OpenIDConnectProvider	✓ Yes[1]	✓ Yes[2]	✗ No
AWS::IAM::Role	✗ No	✗ No	✓ Yes[2]
AWS::IAM::SAMLProvider	✓ Yes[1]	✓ Yes[2]	✗ No
AWS::IAM::ServerCertificate	✓ Yes[1]	✓ Yes[2]	✗ No
AWS::IAM::VirtualMFADevice	✓ 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.

EC2 Image Builder

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
`AWS::ImageBuilder::Component`	✗ No	✓ Yes	✗ No
`AWS::ImageBuilder::ContainerRecipe`	✗ No	✓ Yes	✗ No
`AWS::ImageBuilder::DistributionConfiguration`	✗ No	✓ Yes	✗ No
`AWS::ImageBuilder::Image`	✗ No	✓ Yes	✗ No
`AWS::ImageBuilder::ImagePipeline`	✗ No	✓ Yes	✗ No
`AWS::ImageBuilder::ImageRecipe`	✗ No	✓ Yes	✗ No
`AWS::ImageBuilder::InfrastructureConfiguration`	✗ No	✓ Yes	✗ No
`AWS::ImageBuilder::LifecyclePolicy`	✗ No	✓ Yes	✗ No
`AWS::ImageBuilder::Workflow`	✗ No	✓ Yes	✗ No

Amazon Inspector

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Inspector::AssessmentTemplate	✗ No	✓ Yes	✓ Yes
AWS::InspectorV2::CisScanConfiguration	✗ No	✓ Yes	✗ No
AWS::InspectorV2::Filter	✗ No	✓ Yes	✗ No

Internet Monitor

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::InternetMonitor::Monitor	✗ No	✓ Yes	✗ No

AWS IoT

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::IoT::Authorizer	✗ No	✓ Yes	✗ No

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::IoT::BillingGroup	✗ No	✓ Yes	✗ No
AWS::IoT::CACertificate	✗ No	✓ Yes	✗ No
AWS::IoT::CertificateProvider	✗ No	✓ Yes	✗ No
AWS::IoT::Command	✗ No	✓ Yes	✗ No
AWS::IoT::CustomMetric	✗ No	✓ Yes	✗ No
AWS::IoT::Dimension	✗ No	✓ Yes	✗ No
AWS::IoT::DomainConfiguration	✗ No	✓ Yes	✗ No
AWS::IoT::FleetMetric	✗ No	✓ Yes	✗ No
AWS::IoT::Job	✗ No	✓ Yes	✗ No
AWS::IoT::JobTemplate	✗ No	✓ Yes	✗ No
AWS::IoT::MitigationAction	✗ No	✓ Yes	✗ No
AWS::IoT::OTAUpdate	✗ No	✓ Yes	✗ No
AWS::IoT::Policy	✗ No	✓ Yes	✗ No
AWS::IoT::ProvisioningTemplate	✗ No	✓ Yes	✗ No
AWS::IoT::RoleAlias	✗ No	✓ Yes	✗ No
AWS::IoT::ScheduledAudit	✗ No	✓ Yes	✗ No
AWS::IoT::SecurityProfile	✗ No	✓ Yes	✗ No
AWS::IoT::SoftwarePackage	✗ No	✓ Yes	✗ No

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::IoT::Stream	✗ No	✓ Yes	✗ No
AWS::IoT::ThingGroup	✗ No	✓ Yes	✗ No
AWS::IoT::ThingType	✗ No	✓ Yes	✗ No
AWS::IoT::TopicRule	✗ No	✓ Yes	✓ Yes
AWS::IoT::Tunnel	✗ No	✓ Yes	✗ No

AWS IoT Analytics

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

AWS IoT Core Device Advisor

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::IoTCoreDeviceAdvisor::SuiteDefinition	✗ No	✓ Yes	✗ No
AWS::IoTCoreDeviceAdvisor::SuiteRun	✗ No	✓ Yes	✗ No

AWS IoT Events

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::IoTEvents::AlarmModel	✗ No	✓ Yes	✗ No
AWS::IoTEvents::DetectorModel	✓ Yes	✓ Yes	✓ Yes
AWS::IoTEvents::Input	✓ Yes	✓ Yes	✓ Yes

AWS IoT FleetWise

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::IoTFleetWise::Campaign	✗ No	✓ Yes	✓ Yes
AWS::IoTFleetWise::DecoderManifest	✗ No	✓ Yes	✓ Yes
AWS::IoTFleetWise::Fleet	✗ No	✓ Yes	✓ Yes
AWS::IoTFleetWise::ModelManifest	✗ No	✓ Yes	✓ Yes
AWS::IoTFleetWise::SignalCatalog	✗ No	✓ Yes	✓ Yes
AWS::IoTFleetWise::StateTemplate	✗ No	✓ Yes	✗ No
AWS::IoTFleetWise::Vehicle	✗ No	✓ Yes	✓ Yes

AWS IoT Greengrass

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Greengrass::BulkDeployment	✗ No	✓ Yes	✗ No
AWS::Greengrass::ConnectorDefinition	✓ Yes	✓ Yes	✗ No
AWS::Greengrass::CoreDefinition	✓ Yes	✓ Yes	✗ No
AWS::Greengrass::DeviceDefinition	✓ Yes	✓ Yes	✗ No

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
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 IoT Greengrass Version 2

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::GreengrassV2::ComponentVersion	× No	✓ Yes	× No
AWS::GreengrassV2::CoreDevice	× No	✓ Yes	× No

AWS IoT SiteWise console

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::IoTSiteWise::AccessPolicy	✗ No	✓ Yes	✗ No
AWS::IoTSiteWise::Asset	✗ No	✓ Yes	✗ No
AWS::IoTSiteWise::AssetModel	✗ No	✓ Yes	✗ No
AWS::IoTSiteWise::Dashboard	✗ No	✓ Yes	✗ No
AWS::IoTSiteWise::Dataset	✗ No	✓ Yes	✗ No
AWS::IoTSiteWise::Gateway	✗ No	✓ Yes	✗ No
AWS::IoTSiteWise::Portal	✗ No	✓ Yes	✗ No
AWS::IoTSiteWise::Project	✗ No	✓ Yes	✗ No
AWS::IoTSiteWise::TimeSeries	✗ No	✓ Yes	✗ No

AWS IoT Wireless

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::IoTWireless::Destination	✗ No	✓ Yes	✗ No
AWS::IoTWireless::DeviceProfile	✗ No	✓ Yes	✗ No

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::IoTWireless::FuotaTask	✗ No	✓ Yes	✗ No
AWS::IoTWireless::ImportTask	✗ No	✓ Yes	✗ No
AWS::IoTWireless::MulticastGroup	✗ No	✓ Yes	✗ No
AWS::IoTWireless::NetworkAnalyzerConfiguration	✗ No	✓ Yes	✗ No
AWS::IoTWireless::PartnerAccount	✗ No	✓ Yes	✗ No
AWS::IoTWireless::ServiceProfile	✗ No	✓ Yes	✗ No
AWS::IoTWireless::TaskDefinition	✗ No	✓ Yes	✗ No
AWS::IoTWireless::WirelessDevice	✗ No	✓ Yes	✗ No
AWS::IoTWireless::WirelessGateway	✗ No	✓ Yes	✗ No

Amazon Kendra

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Kendra::DataSource	✗ No	✓ Yes	✗ No
AWS::Kendra::FeaturedResultsSet	✗ No	✓ Yes	✗ No
AWS::Kendra::Index	✗ No	✓ Yes	✗ No

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Kendra::QuerySuggestionsBlockList	✗ No	✓ Yes	✗ No
AWS::Kendra::Thesaurus	✗ No	✓ Yes	✗ No

Amazon Kendra Intelligent Ranking

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::KendraRanking::ExecutionPlan	✗ No	✓ 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 Keyspaces (for Apache Cassandra)

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Cassandra::Keyspace	✗ No	✓ Yes	✓ Yes
AWS::Cassandra::Table	✗ No	✓ Yes	✗ No

Amazon Kinesis

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

Amazon Managed Service for Apache Flink

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 Data Firehose

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

Amazon Kinesis Video Streams

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

AWS Lambda

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

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

AWS Launch Wizard

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::LaunchWizard::Deployment	× No	✓ Yes	× No

Amazon Lex

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Lex::Bot	× No	✓ Yes	× No
AWS::Lex::BotAlias	× No	✓ Yes	× No
AWS::LexV2::TestSet	× No	✓ Yes	× No

AWS License Manager

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::LicenseManager::License	✗ No	✓ Yes	✗ No
AWS::LicenseManager::LicenseConfiguration	✗ No	✓ Yes	✗ No
AWS::LicenseManager::ReportGenerator	✗ No	✓ Yes	✗ No

Amazon Lightsail

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Lightsail::Bucket	✗ No	✓ Yes	✗ No
AWS::Lightsail::Certificate	✗ No	✓ Yes	✗ No
AWS::Lightsail::Container	✗ No	✓ Yes	✗ No
AWS::Lightsail::Database	✗ No	✓ Yes	✗ No
AWS::Lightsail::Disk	✗ No	✓ Yes	✗ No
AWS::Lightsail::DiskSnapshot	✗ No	✓ Yes	✗ No
AWS::Lightsail::Distribution	✗ No	✓ Yes	✗ No

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Lightsail::Domain	✗ No	✓ Yes	✗ No
AWS::Lightsail::Instance	✗ No	✓ Yes	✗ No
AWS::Lightsail::InstanceSnapshot	✗ No	✓ Yes	✗ No
AWS::Lightsail::KeyPair	✗ No	✓ Yes	✗ No
AWS::Lightsail::LoadBalancer	✗ No	✓ Yes	✗ No
AWS::Lightsail::RelationalDatabaseSnapshot	✗ No	✓ Yes	✗ No
AWS::Lightsail::StaticIp	✗ No	✓ Yes	✗ No

Linux subscriptions in AWS License Manager

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::LicenseManagerLinuxSubscriptions::SubscriptionProvider	✗ No	✓ Yes	✗ No

Amazon Location Service

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Location::GeofenceCollection	✗ No	✓ Yes	✗ No
AWS::Location::Map	✗ No	✓ Yes	✗ No
AWS::Location::PlaceIndex	✗ No	✓ Yes	✗ No
AWS::Location::RouteCalculator	✗ No	✓ Yes	✗ No
AWS::Location::Tracker	✗ No	✓ Yes	✗ No

Lookout for Equipment

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::LookoutEquipment::Dataset	✗ No	✓ Yes	✗ No
AWS::LookoutEquipment::InferenceScheduler	✗ No	✓ Yes	✗ No
AWS::LookoutEquipment::LabelGroup	✗ No	✓ Yes	✗ No
AWS::LookoutEquipment::Model	✗ No	✓ Yes	✗ No

Amazon Lookout for Metrics

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::LookoutMetrics::Alert	✗ No	✓ Yes	✗ No
AWS::LookoutMetrics::AnomalyDetector	✗ No	✓ Yes	✗ No
AWS::LookoutMetrics::MetricSet	✗ No	✓ Yes	✗ No

Lookout for Vision

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::LookoutVision::Model	✗ No	✓ Yes	✗ No

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 Machine Learning

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::MachineLearning::BatchPrediction	✗ No	✓ Yes	✗ No
AWS::MachineLearning::DataSource	✗ No	✓ Yes	✗ No
AWS::MachineLearning::Evaluation	✗ No	✓ Yes	✗ No
AWS::MachineLearning::MLModel	✗ No	✓ 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

AWS Mainframe Modernization

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::M2::Application	✗ No	✓ Yes	✗ No
AWS::M2::Environment	✗ No	✓ Yes	✗ No

AWS Mainframe Modernization Application Testing

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::AppTest::TestCase	✗ No	✓ Yes	✗ No
AWS::AppTest::TestConfiguration	✗ No	✓ Yes	✗ No
AWS::AppTest::TestRun	✗ No	✓ Yes	✗ No
AWS::AppTest::TestSuite	✗ No	✓ Yes	✗ No

Amazon Managed Blockchain

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::ManagedBlockchain::Accessor	✗ No	✓ Yes	✗ No
AWS::ManagedBlockchain::Invitation	✗ No	✓ Yes	✗ No
AWS::ManagedBlockchain::Member	✗ No	✓ Yes	✗ No
AWS::ManagedBlockchain::Network	✗ No	✓ Yes	✗ No
AWS::ManagedBlockchain::Node	✗ No	✓ Yes	✗ No
AWS::ManagedBlockchain::Proposal	✗ No	✓ Yes	✗ No

Amazon Managed Grafana

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Grafana::Workspace	✗ No	✓ Yes	✗ No

Amazon Managed Service for Prometheus

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::APS::RuleGroupsNamespace	✗ No	✓ Yes	✗ No
AWS::APS::Scraper	✗ No	✓ Yes	✗ No
AWS::APS::Workspace	✗ No	✓ Yes	✗ No

Amazon Managed Streaming for Apache Kafka

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

Amazon Managed Streaming for Apache Kafka Connect

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::KafkaConnect::Connector	✗ No	✓ Yes	✗ No
AWS::KafkaConnect::CustomPlugin	✗ No	✓ Yes	✗ No
AWS::KafkaConnect::WorkerConfiguration	✗ No	✓ Yes	✗ No

Amazon Managed Workflows for Apache Airflow

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

AWS Marketplace Catalog API

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::MarketplaceCatalog::ChangeSet	✗ No	✓ Yes	✗ No

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::MarketplaceCatalog::Entity	✗ No	✓ Yes	✗ No

AWS Elemental MediaConnect

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::MediaConnect::Flow	✗ No	✓ Yes	✗ No
AWS::MediaConnect::FlowEntitlement	✗ No	✓ Yes	✗ No
AWS::MediaConnect::FlowOutput	✗ No	✓ Yes	✗ No
AWS::MediaConnect::FlowSource	✗ No	✓ Yes	✗ No

AWS Elemental MediaPackage

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::MediaPackage::Asset	✗ No	✓ Yes	✗ No
AWS::MediaPackage::Channel	✗ No	✓ Yes	✗ No

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::MediaPackage::OriginEndpoint	✗ No	✓ Yes	✗ No
AWS::MediaPackage::PackagingConfiguration	✗ No	✓ Yes	✗ No
AWS::MediaPackage::PackagingGroup	✗ No	✓ Yes	✗ No

Amazon MemoryDB

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::MemoryDB::ACL	✗ No	✓ Yes	✗ No
AWS::MemoryDB::Cluster	✗ No	✓ Yes	✗ No
AWS::MemoryDB::MultiRegionCluster	✗ No	✓ Yes	✗ No
AWS::MemoryDB::ParameterGroup	✗ No	✓ Yes	✗ No
AWS::MemoryDB::Snapshot	✗ No	✓ Yes	✗ No
AWS::MemoryDB::SubnetGroup	✗ No	✓ Yes	✗ No
AWS::MemoryDB::User	✗ No	✓ Yes	✗ No

AWS Migration Hub Orchestrator

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::MigrationHubOrchestrator::Template	✗ No	✓ Yes	✗ No
AWS::MigrationHubOrchestrator::Workflow	✗ No	✓ Yes	✗ No

AWS Migration Hub Refactor Spaces

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::RefactorSpaces::Application	✗ No	✓ Yes	✗ No
AWS::RefactorSpaces::Environment	✗ No	✓ Yes	✗ No
AWS::RefactorSpaces::Route	✗ No	✓ Yes	✗ No
AWS::RefactorSpaces::Service	✗ No	✓ Yes	✗ No

Amazon Neptune

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::NeptuneGraph::Graph	✗ No	✓ Yes	✗ No
AWS::NeptuneGraph::GraphSnapshot	✗ No	✓ Yes	✗ No

AWS Network Firewall

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::NetworkFirewall::Firewall	✗ No	✓ Yes	✗ No
AWS::NetworkFirewall::FirewallPolicy	✗ No	✓ Yes	✗ No
AWS::NetworkFirewall::RuleGroup	✗ No	✓ Yes	✗ No

Network Synthetic Monitor

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::NetworkMonitor::Monitor	✗ No	✓ Yes	✗ No

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::NetworkMonitor::Probe	✗ No	✓ Yes	✗ No

AWS Network Manager

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::NetworkManager::Connection	✗ No	✓ Yes	✗ No
AWS::NetworkManager::ConnectPeer	✗ No	✓ Yes	✗ No
AWS::NetworkManager::CoreNetwork	✗ No	✓ Yes	✗ No
AWS::NetworkManager::Device	✗ No	✓ Yes	✗ No
AWS::NetworkManager::GlobalNetwork	✗ No	✓ Yes	✗ No
AWS::NetworkManager::Link	✗ No	✓ Yes	✗ No
AWS::NetworkManager::Site	✗ No	✓ Yes	✗ No
AWS::NetworkManager::TransitGatewayPeering	✗ No	✓ Yes	✗ No
AWS::NetworkManager::VpcAttachment	✗ No	✓ Yes	✗ No

Amazon One

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::One::DeviceConfigurationTemplate	✗ No	✓ Yes	✗ No
AWS::One::DeviceInstance	✗ No	✓ Yes	✗ No
AWS::One::Site	✗ No	✓ Yes	✗ No

Amazon OpenSearch Service OpenSearch

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

OpenSearch Serverless

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::OpenSearchServerless::Collection	✗ No	✓ Yes	✗ No

Amazon OpenSearch Service

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

Amazon OpenSearch Service Ingestion

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

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::ResourcePolicy	✗ No	✓ Yes	✗ No
AWS::Organizations::Root	✓ Yes	✓ Yes	✗ No

AWS Outposts

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Outposts::Outpost	✗ No	✓ Yes	✗ No
AWS::Outposts::Site	✗ No	✓ Yes	✗ No

AWS Panorama

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Panorama::ApplicationInstance	✗ No	✓ Yes	✗ No
AWS::Panorama::Device	✗ No	✓ Yes	✗ No
AWS::Panorama::Package	✗ No	✓ Yes	✗ No

AWS Parallel Computing Service

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

AWS Payment Cryptography

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

Amazon Payments

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Payments::PaymentInstrument	✗ No	✓ Yes	✗ No

Amazon Relational Database Service Performance Insights

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Pi::PerformanceAnalysisReport	✗ No	✓ Yes	✗ No

Amazon Personalize

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Personalize::BatchInferenceJob	✗ No	✓ Yes	✗ No
AWS::Personalize::BatchSegmentJob	✗ No	✓ Yes	✗ No
AWS::Personalize::Campaign	✗ No	✓ Yes	✗ No
AWS::Personalize::Dataset	✗ No	✓ Yes	✗ No
AWS::Personalize::DatasetExportJob	✗ No	✓ Yes	✗ No
AWS::Personalize::DatasetGroup	✗ No	✓ Yes	✗ No
AWS::Personalize::DatasetImportJob	✗ No	✓ Yes	✗ No
AWS::Personalize::EventTracker	✗ No	✓ Yes	✗ No
AWS::Personalize::Filter	✗ No	✓ Yes	✗ No
AWS::Personalize::Recommender	✗ No	✓ Yes	✗ No
AWS::Personalize::Solution	✗ No	✓ 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 Pinpoint SMS and Voice API

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::PinpointSMSVoiceV2::ConfigurationSet	✗ No	✓ Yes	✗ No
AWS::PinpointSMSVoiceV2::OptOutList	✗ No	✓ Yes	✗ No
AWS::PinpointSMSVoiceV2::PhoneNumber	✗ No	✓ Yes	✗ No
AWS::PinpointSMSVoiceV2::Pool	✗ No	✓ Yes	✗ No

AWS Pricing Calculator

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::BCMPricingCalculator::BillEstimate	✗ No	✓ Yes	✗ No
AWS::BCMPricingCalculator::BillScenario	✗ No	✓ Yes	✗ No
AWS::BCMPricingCalculator::WorkloadEstimate	✗ No	✓ Yes	✗ No

AWS Private CA Connector for Active Directory

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::PCAConnectorAD::Connector	✗ No	✓ Yes	✗ No

AWS Private CA Connector for SCEP

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::PCAConnectorScep::Connector	✗ No	✓ Yes	✗ No

AWS Proton

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Proton::Component	✗ No	✓ Yes	✗ No
AWS::Proton::Deployment	✗ No	✓ Yes	✗ No
AWS::Proton::Environment	✗ No	✓ Yes	✗ No
AWS::Proton::EnvironmentAccountConnection	✗ No	✓ Yes	✗ No
AWS::Proton::EnvironmentTemplate	✗ No	✓ Yes	✗ No
AWS::Proton::Repository	✗ No	✓ Yes	✗ No
AWS::Proton::Service	✗ No	✓ Yes	✗ No
AWS::Proton::ServiceInstance	✗ No	✓ Yes	✗ No
AWS::Proton::ServiceTemplate	✗ No	✓ Yes	✗ No

Amazon Q Business Apps

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::QApps::QApp	✗ No	✓ Yes	✗ No
AWS::QApps::QAppSession	✗ No	✓ Yes	✗ No

Amazon Q Business

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::QBusiness::Application	✗ No	✓ Yes	✗ No
AWS::QBusiness::DataSource	✗ No	✓ Yes	✗ No
AWS::QBusiness::Index	✗ No	✓ Yes	✗ No
AWS::QBusiness::Plugin	✗ No	✓ Yes	✗ No
AWS::QBusiness::Retriever	✗ No	✓ Yes	✗ No
AWS::QBusiness::WebExperience	✗ 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
AWS::QLDB::Table	× No	✓ Yes	× No

Amazon QuickSight

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::QuickSight::Analysis	× No	✓ Yes	× No
AWS::QuickSight::Brand	× No	✓ Yes	× No
AWS::QuickSight::CustomPermissions	× No	✓ Yes	× No
AWS::QuickSight::Dashboard	× No	✓ Yes	× No
AWS::QuickSight::DataSet	× No	✓ Yes	× No
AWS::QuickSight::DataSource	× No	✓ Yes	× No
AWS::QuickSight::Folder	× No	✓ Yes	× No
AWS::QuickSight::Namespace	× No	✓ Yes	× No

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::QuickSight::Template	✗ No	✓ Yes	✗ No
AWS::QuickSight::Theme	✗ No	✓ Yes	✗ No
AWS::QuickSight::Topic	✗ No	✓ Yes	✗ No
AWS::QuickSight::User	✗ No	✓ Yes	✗ No
AWS::QuickSight::VPCConnection	✗ No	✓ Yes	✗ No

AWS DeepRacer

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::DeepRacer::Car	✗ No	✓ Yes	✗ No
AWS::DeepRacer::EvaluationJob	✗ No	✓ Yes	✗ No
AWS::DeepRacer::Leaderboard	✗ No	✓ Yes	✗ No
AWS::DeepRacer::LeaderboardEvaluationJob	✗ No	✓ Yes	✗ No
AWS::DeepRacer::ReinforcementLearningModel	✗ No	✓ Yes	✗ No
AWS::DeepRacer::TrainingJob	✗ No	✓ Yes	✗ No

Recycle Bin

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::RBin::Rule	✗ No	✓ Yes	✗ No

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::EventSubscription	✗ No	✓ Yes	✗ No
AWS::Redshift::HSMClientCertificate	✓ Yes	✓ Yes	✗ No
AWS::Redshift::HSMConfiguration	✗ No	✓ Yes	✗ No
AWS::Redshift::Integration	✗ No	✓ Yes	✗ No
AWS::Redshift::Namespace	✗ No	✓ Yes	✗ No
AWS::Redshift::Snapshot	✗ No	✓ Yes	✗ No

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Redshift::SnapshotCopyGrant	✗ No	✓ Yes	✗ No
AWS::Redshift::SnapshotSchedule	✗ No	✓ Yes	✗ No
AWS::Redshift::UsageLimit	✗ No	✓ Yes	✗ No

Amazon Redshift Serverless

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::RedshiftServerless::Namespace	✗ No	✓ Yes	✗ No
AWS::RedshiftServerless::RecoveryPoint	✗ No	✓ Yes	✗ No
AWS::RedshiftServerless::Snapshot	✗ No	✓ Yes	✗ No
AWS::RedshiftServerless::Workgroup	✗ No	✓ Yes	✗ No

Amazon Rekognition

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Rekognition::Collection	✗ No	✓ Yes	✗ No
AWS::Rekognition::StreamProcessor	✗ No	✓ Yes	✗ No

Amazon Relational Database Service (Amazon RDS)

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::RDS::CustomDBEngineVersion	✗ No	✓ Yes	✗ No
AWS::RDS::DBCluster	✓ Yes	✓ Yes	✓ Yes
AWS::RDS::DBClusterEndpoint	✗ No	✓ Yes	✗ No
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::DBProxy	✗ No	✓ Yes	✗ No
AWS::RDS::DBProxyEndpoint	✗ No	✓ Yes	✗ No

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::RDS::DBProxyTargetGroup	✗ No	✓ Yes	✗ No
AWS::RDS::DBSecurityGroup	✓ Yes	✓ Yes	✓ Yes
AWS::RDS::DBSnapshot	✓ Yes	✓ Yes	✗ No
AWS::RDS::DBSubnetGroup	✓ Yes	✓ Yes	✓ Yes
AWS::RDS::Deployment	✗ No	✓ Yes	✗ No
AWS::RDS::EventSubscription	✓ Yes	✓ Yes	✗ No
AWS::RDS::GlobalCluster	✗ No	✓ Yes	✗ No
AWS::RDS::Integration	✗ No	✓ Yes	✗ No
AWS::RDS::OptionGroup	✓ Yes	✓ Yes	✗ No
AWS::RDS::ReservedDBInstance	✓ Yes	✓ Yes	✗ No
AWS::RDS::SnapshotTenantDatabase	✗ No	✓ Yes	✗ No
AWS::RDS::TenantDatabase	✗ No	✓ Yes	✗ No

AWS Resilience Hub

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::ResilienceHub::App	✗ No	✓ Yes	✗ No
AWS::ResilienceHub::AppAssessment	✗ No	✓ Yes	✗ No
AWS::ResilienceHub::RecommendationTemplate	✗ No	✓ Yes	✗ No
AWS::ResilienceHub::ResiliencyPolicy	✗ No	✓ 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
AWS::RoboMaker::SimulationJobBatch	× No	✓ Yes	× No
AWS::RoboMaker::World	× No	✓ Yes	× No
AWS::RoboMaker::WorldExportJob	× No	✓ Yes	× No

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::RoboMaker::WorldGenerationJob	✗ No	✓ Yes	✗ No
AWS::RoboMaker::WorldTemplate	✗ No	✓ 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

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Route53RecoveryControl::Cluster	✗ No	✓ Yes	✗ No
AWS::Route53RecoveryControl::ControlPanel	✗ No	✓ Yes	✗ No
AWS::Route53RecoveryControl::SafetyRule	✗ No	✓ Yes	✗ No

Amazon Route 53 Profiles

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Route53Profiles::Profile	✗ No	✓ Yes	✗ No
AWS::Route53Profiles::ProfileAssociation	✗ No	✓ Yes	✗ No

Amazon Route 53 Recovery Readiness in Application Recovery Controller (ARC)

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Route53RecoveryReadiness::Cell	✗ No	✓ Yes	✗ No
AWS::Route53RecoveryReadiness::ReadinessCheck	✗ No	✓ Yes	✗ No
AWS::Route53RecoveryReadiness::RecoveryGroup	✗ No	✓ Yes	✗ No
AWS::Route53RecoveryReadiness::ResourceSet	✗ No	✓ Yes	✗ No

Amazon Route 53 Resolver

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Route53Resolver::FirewallDomainList	✗ No	✓ Yes[2]	✗ No
AWS::Route53Resolver::FirewallRuleGroup	✗ No	✓ Yes[2]	✗ No
AWS::Route53Resolver::FirewallRuleGroupAssociation	✗ No	✓ Yes[2]	✗ No

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Route53Resolver::OutpostResolver	✗ No	✓ Yes[2]	✗ No
AWS::Route53Resolver::ResolverEndpoint	✓ Yes[1]	✓ Yes[2]	✗ No
AWS::Route53Resolver::ResolverQueryLoggingConfig	✗ No	✓ Yes[2]	✗ No
AWS::Route53Resolver::ResolverRule	✓ 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
AWS::Glacier::Vault	✓ Yes	✓ Yes	✗ No

AWS SQL Workbench

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::SQLWorkbench::Chart	✗ No	✓ Yes	✗ No
AWS::SQLWorkbench::Connection	✗ No	✓ Yes	✗ No
AWS::SQLWorkbench::Notebook	✗ No	✓ Yes	✗ No
AWS::SQLWorkbench::SavedQuery	✗ No	✓ Yes	✗ No

Amazon SageMaker AI

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::SageMaker::Action	✗ No	✓ Yes	✗ No
AWS::SageMaker::Algorithm	✗ No	✓ Yes	✗ No
AWS::SageMaker::App	✗ No	✓ Yes	✗ No
AWS::SageMaker::AppImageConfig	✗ No	✓ Yes	✗ No
AWS::SageMaker::Artifact	✗ No	✓ Yes	✗ No
AWS::SageMaker::AutoMLJob	✗ No	✓ Yes	✗ No
AWS::SageMaker::Cluster	✗ No	✓ Yes	✗ No

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::SageMaker::ClusterSchedulerConfig	✗ No	✓ Yes	✗ No
AWS::SageMaker::CodeRepository	✗ No	✓ Yes	✗ No
AWS::SageMaker::CompilationJob	✗ No	✓ Yes	✗ No
AWS::SageMaker::ComputeQuota	✗ No	✓ Yes	✗ No
AWS::SageMaker::Context	✗ No	✓ Yes	✗ No
AWS::SageMaker::DataQualityJobDefinition	✗ No	✓ Yes	✗ No
AWS::SageMaker::DeviceFleet	✗ No	✓ Yes	✗ No
AWS::SageMaker::Domain	✗ No	✓ Yes	✗ No
AWS::SageMaker::EdgeDeploymentPlan	✗ No	✓ Yes	✗ No
AWS::SageMaker::EdgePackagingJob	✗ No	✓ Yes	✗ No
AWS::SageMaker::Endpoint	✗ No	✓ Yes	✓ Yes
AWS::SageMaker::EndpointConfig	✗ No	✓ Yes	✓ Yes
AWS::SageMaker::Experiment	✗ No	✓ Yes	✗ No
AWS::SageMaker::ExperimentTrial	✗ No	✓ Yes	✗ No
AWS::SageMaker::ExperimentTrialComponent	✗ No	✓ Yes	✗ No
AWS::SageMaker::FeatureGroup	✗ No	✓ Yes	✗ No

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::SageMaker::FlowDefinition	✗ No	✓ Yes	✗ No
AWS::SageMaker::Hub	✗ No	✓ Yes	✗ No
AWS::SageMaker::HubContent	✗ No	✓ Yes	✗ No
AWS::SageMaker::HumanTaskUi	✗ No	✓ Yes	✗ No
AWS::SageMaker::HyperParameterTuningJob	✗ No	✓ Yes	✗ No
AWS::SageMaker::Image	✗ No	✓ Yes	✗ No
AWS::SageMaker::InferenceComponent	✗ No	✓ Yes	✗ No
AWS::SageMaker::InferenceExperiment	✗ No	✓ Yes	✗ No
AWS::SageMaker::InferenceRecommendationsJob	✗ No	✓ Yes	✗ No
AWS::SageMaker::LabelingJob	✗ No	✓ Yes	✗ No
AWS::SageMaker::LineageGroup	✗ No	✓ Yes	✗ No
AWS::SageMaker::MlflowTrackingServer	✗ No	✓ Yes	✗ No
AWS::SageMaker::Model	✗ No	✓ Yes	✓ Yes
AWS::SageMaker::ModelBiasJobDefinition	✗ No	✓ Yes	✗ No
AWS::SageMaker::ModelCard	✗ No	✓ Yes	✗ No

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::SageMaker::ModelExplainabilityJobDefinition	✗ No	✓ Yes	✗ No
AWS::SageMaker::ModelPackage	✗ No	✓ Yes	✗ No
AWS::SageMaker::ModelPackageGroup	✗ No	✓ Yes	✓ Yes
AWS::SageMaker::ModelQualityJobDefinition	✗ No	✓ Yes	✗ No
AWS::SageMaker::MonitoringSchedule	✗ No	✓ Yes	✗ No
AWS::SageMaker::NotebookInstance	✓ Yes	✓ Yes	✓ Yes
AWS::SageMaker::OptimizationJob	✗ No	✓ Yes	✗ No
AWS::SageMaker::Pipeline	✗ No	✓ Yes	✗ No
AWS::SageMaker::ProcessingJob	✗ No	✓ Yes	✗ No
AWS::SageMaker::Project	✗ No	✓ Yes	✓ Yes
AWS::SageMaker::Space	✗ No	✓ Yes	✗ No
AWS::SageMaker::StudioLifecycleConfig	✗ No	✓ Yes	✗ No
AWS::SageMaker::TrainingJob	✗ No	✓ Yes	✗ No
AWS::SageMaker::TransformJob	✗ No	✓ Yes	✗ No
AWS::SageMaker::UserProfile	✗ No	✓ Yes	✗ No
AWS::SageMaker::Workforce	✗ No	✓ Yes	✗ No

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::SageMaker::Workteam	✗ No	✓ Yes	✗ No

Amazon SageMaker AI geospatial

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::SagemakerGeospatial::EarthObservationJob	✗ No	✓ Yes	✗ No
AWS::SagemakerGeospatial::RasterDataCollection	✗ No	✓ Yes	✗ No
AWS::SagemakerGeospatial::VectorEnrichmentJob	✗ No	✓ Yes	✗ No

Savings Plans

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::SavingsPlans::SavingsPlan	✗ 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 Security Hub

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::SecurityHub::AutomationRule	✗ No	✓ Yes	✗ No
AWS::SecurityHub::ConfigurationPolicy	✗ No	✓ Yes	✗ No
AWS::SecurityHub::Hub	✗ No	✓ Yes	✗ No
AWS::SecurityHub::ProductSubscription	✗ No	✓ Yes	✗ No

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

AWS Service Catalog AppRegistry

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

Service Quotas

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

AWS Shield

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Shield::Protection	✗ No	✓ Yes	✗ No
AWS::Shield::ProtectionGroup	✗ No	✓ Yes	✗ No

AWS SimSpace Weaver

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::SimSpaceWeaver::Simulation	✗ 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
AWS::SES::MailManagerArchive	× No	✓ Yes	× No
AWS::SES::MailManagerIngressPoint	× No	✓ Yes	× No
AWS::SES::MailManagerRuleSet	× No	✓ Yes	× No
AWS::SES::MailManagerTrafficPolicy	× No	✓ 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::AccessGrant	× No	✓ Yes	× No
AWS::S3::AccessGrantsLocation	× No	✓ Yes	× No
AWS::S3::Bucket	✓ Yes	✓ Yes	✓ Yes
AWS::S3::Job	× No	✓ Yes	× No
AWS::S3::StorageLens	× No	✓ Yes	× No
AWS::S3::StorageLensGroup	× No	✓ Yes	× No

Amazon Simple Workflow Service

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::SWF::Domain	✗ No	✓ Yes	✗ No

AWS Snowball Edge Device Management

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::SnowDeviceManagement::ManagedDevice	✗ No	✓ Yes	✗ No
AWS::SnowDeviceManagement::Task	✗ No	✓ Yes	✗ No

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::FileShare	✗ No	✓ Yes	✗ No
AWS::StorageGateway::FileSystemAssociation	✗ No	✓ Yes	✗ No
AWS::StorageGateway::Gateway	✓ Yes	✓ Yes	✗ No
AWS::StorageGateway::Tape	✗ No	✓ Yes	✗ No
AWS::StorageGateway::TapePool	✗ No	✓ Yes	✗ No
AWS::StorageGateway::Volume	✗ No	✓ Yes	✗ No

AWS Supply Chain

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

AWS Systems Manager

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

AWS Systems Manager Incident Manager

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::SSMIncidents::IncidentRecord	✗ No	✓ Yes	✗ No

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::SSMIncidents::ReplicationSet	✗ No	✓ Yes	✗ No
AWS::SSMIncidents::ResponsePlan	✗ No	✓ Yes	✗ No

AWS Systems Manager Incident Manager Contacts

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::SSMContacts::Contact	✗ No	✓ Yes	✗ No
AWS::SSMContacts::Rotation	✗ No	✓ Yes	✗ No

AWS Systems Manager Quick Setup

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::SSMQuickSetup::ConfigurationManager	✗ No	✓ Yes	✗ No

AWS Systems Manager for SAP

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

AWS Telco Network Builder

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::TNB::FunctionPackage	✗ No	✓ Yes	✗ No
AWS::TNB::NetworkInstance	✗ No	✓ Yes	✗ No
AWS::TNB::NetworkPackage	✗ No	✓ Yes	✗ No

Amazon Textract

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Textract::Adapter	✗ No	✓ Yes	✗ No

Amazon Timestream

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Timestream::Database	✗ No	✓ Yes	✗ No
AWS::Timestream::ScheduledQuery	✗ No	✓ Yes	✓ Yes
AWS::Timestream::Table	✗ No	✓ Yes	✗ No

Amazon Transcribe

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Transcribe::LanguageModel	✗ No	✓ Yes	✗ No
AWS::Transcribe::MedicalScribeJob	✗ No	✓ Yes	✗ No
AWS::Transcribe::MedicalTranscriptionJob	✗ No	✓ Yes	✗ No
AWS::Transcribe::MedicalVocabulary	✗ No	✓ Yes	✗ No
AWS::Transcribe::TranscriptionJob	✗ No	✓ Yes	✗ No
AWS::Transcribe::Vocabulary	✗ No	✓ Yes	✗ No
AWS::Transcribe::VocabularyFilter	✗ No	✓ Yes	✗ No

AWS Transfer Family

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Transfer::Agreement	✗ No	✓ Yes	✗ No
AWS::Transfer::Certificate	✗ No	✓ Yes	✗ No
AWS::Transfer::Connector	✗ No	✓ Yes	✗ No
AWS::Transfer::HostKey	✗ No	✓ Yes	✗ No
AWS::Transfer::Profile	✗ No	✓ Yes	✗ No
AWS::Transfer::Server	✗ No	✓ Yes	✗ No
AWS::Transfer::User	✗ No	✓ Yes	✗ No
AWS::Transfer::WebApp	✗ No	✓ Yes	✗ No
AWS::Transfer::Workflow	✗ No	✓ Yes	✗ No

Amazon Translate

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Translate::ParallelData	✗ No	✓ Yes	✗ No
AWS::Translate::Terminology	✗ No	✓ Yes	✗ No

AWS User Notifications

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::UserNotifications::NotificationConfiguration	✗ No	✓ Yes	✗ No

User subscriptions in AWS License Manager

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::LicenseManagerUserSubscriptions::AssociateUser	✗ No	✓ Yes	✗ No
AWS::LicenseManagerUserSubscriptions::IdentityProvider	✗ No	✓ Yes	✗ No
AWS::LicenseManagerUserSubscriptions::LicenseServerEndpoint	✗ No	✓ Yes	✗ No
AWS::LicenseManagerUserSubscriptions::ProductSubscription	✗ No	✓ Yes	✗ No

Amazon VPC Lattice

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::VpcLattice::AccessLogSubscription	✗ No	✓ Yes	✗ No
AWS::VpcLattice::Listener	✗ No	✓ Yes	✗ No
AWS::VpcLattice::ResourceConfiguration	✗ No	✓ Yes	✗ No
AWS::VpcLattice::ResourceGateway	✗ No	✓ Yes	✗ No
AWS::VpcLattice::Rule	✗ No	✓ Yes	✗ No
AWS::VpcLattice::Service	✗ No	✓ Yes	✗ No
AWS::VpcLattice::ServiceNetwork	✗ No	✓ Yes	✗ No
AWS::VpcLattice::ServiceNetworkResourceAssociation	✗ No	✓ Yes	✗ No
AWS::VpcLattice::ServiceNetworkServiceAssociation	✗ No	✓ Yes	✗ No
AWS::VpcLattice::ServiceNetworkVpcAssociation	✗ No	✓ Yes	✗ No
AWS::VpcLattice::TargetGroup	✗ No	✓ Yes	✗ No

AWS Marketplace Vendor Insights

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

AWS WAF

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::WAF::RateBasedRule	✗ No	✓ Yes	✗ No
AWS::WAF::Rule	✗ No	✓ Yes	✗ No
AWS::WAF::RuleGroup	✗ No	✓ Yes	✗ No
AWS::WAF::WebACL	✗ No	✓ Yes	✗ No

AWS WAF Classic Regional

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::WAFRegional::RateBasedRule	✗ No	✓ Yes	✗ No
AWS::WAFRegional::Rule	✗ No	✓ Yes	✗ No
AWS::WAFRegional::RuleGroup	✗ No	✓ Yes	✗ No
AWS::WAFRegional::WebACL	✗ No	✓ Yes	✗ No

AWS Well-Architected Tool

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::WellArchitected::Lens	✗ No	✓ Yes	✗ No
AWS::WellArchitected::Profile	✗ No	✓ Yes	✗ No
AWS::WellArchitected::ReviewTemplate	✗ No	✓ Yes	✗ No
AWS::WellArchitected::Workload	✗ No	✓ Yes	✗ No

AWS Wickr

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Wickr::Network	✗ No	✓ Yes	✗ No

Amazon WorkMail

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::Workmail::Organization	✗ No	✓ Yes	✗ No

Amazon WorkSpaces

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::WorkSpaces::ConnectionAlias	✗ No	✓ Yes	✗ No
AWS::WorkSpaces::Directory	✗ No	✓ Yes	✗ No
AWS::WorkSpaces::Workspace	✓ Yes	✓ Yes	✓ Yes
AWS::WorkSpaces::WorkspaceBundle	✗ No	✓ Yes	✗ No

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::WorkSpaces::WorkspaceImage	✗ No	✓ Yes	✗ No
AWS::WorkSpaces::WorkspaceIpGroup	✗ No	✓ Yes	✗ No
AWS::WorkSpaces::WorkspacesPool	✗ No	✓ Yes	✗ No

Amazon WorkSpaces Secure Browser

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::WorkSpacesWeb::BrowserSettings	✗ No	✓ Yes	✗ No
AWS::WorkSpacesWeb::DataProtectionSettings	✗ No	✓ Yes	✗ No
AWS::WorkSpacesWeb::IdentityProvider	✗ No	✓ Yes	✗ No
AWS::WorkSpacesWeb::IpAccessSettings	✗ No	✓ Yes	✗ No
AWS::WorkSpacesWeb::NetworkSettings	✗ No	✓ Yes	✗ No
AWS::WorkSpacesWeb::Portal	✗ No	✓ Yes	✗ No
AWS::WorkSpacesWeb::TrustStore	✗ No	✓ Yes	✗ No
AWS::WorkSpacesWeb::UserAccessLoggingSettings	✗ No	✓ Yes	✗ No

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::WorkSpacesWeb::UserSettings	✗ No	✓ Yes	✗ No

Amazon WorkSpaces Thin Client

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

AWS X-Ray

Resources	Tag Editor Tagging	Tag-based Groups	AWS CloudFormation Stack-based Groups
AWS::XRay::Group	✗ No	✓ Yes	✗ No
AWS::XRay::SamplingRule	✗ No	✓ Yes	✗ No

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::DeploymentJob	No longer supported by Tag Editor.	May 2, 2022

Creating resource groups with AWS CloudFormation

AWS Resource Groups is integrated with AWS CloudFormation, a service that helps you to model and set up your AWS resources so that you can spend less time creating and managing your resources and infrastructure. You create a template that describes all of the AWS resources that you want (such as resource groups), and AWS CloudFormation provisions and configures those resources for you.

When you use AWS CloudFormation, you can reuse your template to set up your resource groups consistently and repeatedly. Describe your resource groups once, and then provision the same resource groups over and over in multiple AWS accounts and Regions.

Resource Groups and AWS CloudFormation templates

To provision and configure resources for Resource Groups and related services, you must understand AWS CloudFormation templates. Templates are formatted text files in JSON or YAML. These templates describe the resources that you want to provision in your AWS CloudFormation stacks. If you're unfamiliar with JSON or YAML, you can use AWS CloudFormation Designer to help you get started with AWS CloudFormation templates. For more information, see What is AWS CloudFormation Designer? in the *AWS CloudFormation User Guide*.

Resource Groups supports creating resource groups in AWS CloudFormation. For more information, including examples of JSON and YAML templates for resource groups, see the AWS Resource Groups resource type reference in the *AWS CloudFormation User Guide*.

Learn more about AWS CloudFormation

To learn more about AWS CloudFormation, see the following resources:

- AWS CloudFormation
- AWS CloudFormation User Guide
- AWS CloudFormation API Reference
- AWS CloudFormation Command Line Interface User Guide

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
- Identity and access management for AWS Resource Groups
- Logging and monitoring in Resource Groups
- Compliance validation for Resource Groups
- Resilience in Resource Groups
- Infrastructure security in Resource Groups
- Access AWS Resource Groups using an interface endpoint (AWS PrivateLink)
- Security best practices for Resource Groups

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. You are also responsible for the security configuration and management tasks for the AWS services that you use. For more information about data privacy, see the Data Privacy FAQ. For information about data protection in Europe, see the AWS Shared Responsibility Model and GDPR blog post on the *AWS Security Blog*.

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

- Use multi-factor authentication (MFA) with each account.
- Use SSL/TLS to communicate with AWS resources. We require TLS 1.2 and recommend TLS 1.3.
- Set up API and user activity logging with AWS CloudTrail. For information about using CloudTrail trails to capture AWS activities, see Working with CloudTrail trails in the *AWS CloudTrail User Guide*.
- Use AWS encryption solutions, along with all default security controls within AWS services.
- Use advanced managed security services such as Amazon Macie, which assists in discovering and securing sensitive data that is stored in Amazon S3.
- If you require FIPS 140-3 validated cryptographic modules when accessing AWS through a command line interface or an API, use a FIPS endpoint. For more information about the available FIPS endpoints, see Federal Information Processing Standard (FIPS) 140-3.

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

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
- Authenticating with identities
- Managing access using policies
- How Resource Groups works with IAM
- AWS managed policies for AWS Resource Groups
- Using service-linked roles for Resource Groups
- AWS Resource Groups identity-based policy examples
- Troubleshooting AWS Resource Groups identity and access

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.

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 service users 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.

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.

Authenticating with identities

Authentication is how you sign in to AWS using your identity credentials. You must be *authenticated* (signed in to AWS) as the AWS account root user, as an IAM user, or by assuming an IAM role.

You can sign in to AWS as a federated identity by using credentials provided through an identity source. AWS IAM Identity Center (IAM Identity Center) users, your company's single sign-on authentication, and your Google or Facebook credentials are examples of federated identities. When you sign in as a federated identity, your administrator previously set up identity federation using IAM roles. When you access AWS by using federation, you are indirectly assuming a role.

Depending on the type of user you are, you can sign in to the AWS Management Console or the AWS access portal. For more information about signing in to AWS, see How to sign in to your AWS account in the *AWS Sign-In User Guide*.

If you access AWS programmatically, AWS provides a software development kit (SDK) and a command line interface (CLI) to cryptographically sign your requests by using your credentials. If you don't use AWS tools, you must sign requests yourself. For more information about using the recommended method to sign requests yourself, see AWS Signature Version 4 for API requests in the *IAM User Guide*.

Regardless of the authentication method that you use, you might 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 Multi-factor authentication in the *AWS IAM Identity Center User Guide* and AWS Multi-factor authentication in IAM in the *IAM User Guide*.

AWS account root user

When you create an AWS account, you begin with one 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 don't use the root user for your everyday tasks. Safeguard your root user credentials and use them to perform the tasks that only the root user can perform. For the complete list of tasks that require you to sign in as the root user, see Tasks that require root user credentials in the *IAM User Guide*.

IAM users and groups

An *IAM user* is an identity within your AWS account that has specific permissions for a single person or application. Where possible, we recommend relying on temporary credentials instead of creating IAM users who have long-term credentials such as passwords and access keys. However, if you have specific use cases that require long-term credentials with IAM users, we recommend that you rotate

access keys. For more information, see Rotate access keys regularly for use cases that require long-term credentials in the *IAM User Guide*.

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 Use cases for IAM users 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. To temporarily assume an IAM role in the AWS Management Console, you can switch from a user to an IAM role (console). 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 Methods to assume a role in the *IAM User Guide*.

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

- **Federated user access** – To assign permissions to a federated identity, you create a role and define permissions for the role. When a federated identity authenticates, the identity is associated with the role and is granted the permissions that are defined by the role. For information about roles for federation, see Create a role for a third-party identity provider (federation) in the *IAM User Guide*. If you use IAM Identity Center, you configure a permission set. To control what your identities can access after they authenticate, IAM Identity Center correlates the permission set to a role in IAM. For information about permissions sets, see Permission sets in the *AWS IAM Identity Center User Guide*.
- **Temporary IAM user permissions** – An IAM user or role can assume an IAM role to temporarily take on different permissions for a specific task.
- **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 Cross account resource access in IAM 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.
  - **Forward access sessions (FAS)** – When you use an IAM user or role to perform actions in AWS, you are considered a principal. When you use some services, you might perform an action that then initiates another action in a different service. FAS uses the permissions of the principal calling an AWS service, combined with the requesting AWS service to make requests to downstream services. FAS requests are only made when a service receives a request that requires interactions with other AWS services or resources to complete. In this case, you must have permissions to perform both actions. For policy details when making FAS requests, see Forward access sessions.
  - **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 Create 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 AWS 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 Use an IAM role to grant permissions to applications running on Amazon EC2 instances in the *IAM User Guide*.

Managing access using policies

You control access in AWS by creating policies and attaching them to AWS identities or resources. A policy is an object in AWS that, when associated with an identity or resource, defines their permissions. AWS evaluates these policies when a principal (user, root user, or role session) makes a request. 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**.

By default, users and roles have no permissions. To grant users permission to perform actions on the resources that they need, an IAM administrator can create IAM policies. The administrator can then add the IAM policies to roles, and users can assume the roles.

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 Define custom IAM permissions with customer managed 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 Choose 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 an 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 Service control policies in the *AWS Organizations User Guide*.
- **Resource control policies (RCPs)** – RCPs are JSON policies that you can use to set the maximum available permissions for resources in your accounts without updating the IAM policies attached to each resource that you own. The RCP limits permissions for resources in member accounts and can impact the effective permissions for identities, including the AWS account root user, regardless of whether they belong to your organization. For more information about

Organizations and RCPs, including a list of AWS services that support RCPs, see Resource control policies (RCPs) 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
- Resource-based policies
- Authorization based on Resource Groups tags
- Resource Groups IAM roles

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-east-1:123456789012:group/my-test-group"
```

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

```
"Resource": "arn:aws:resource-groups:us-east-1:123456789012: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 a calling principal 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.

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 and Updating groups in AWS Resource Groups 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.

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 managed policies for AWS Resource Groups

An AWS managed policy is a standalone policy that is created and administered by AWS. AWS managed policies are designed to provide permissions for many common use cases so that you can start assigning permissions to users, groups, and roles.

Keep in mind that AWS managed policies might not grant least-privilege permissions for your specific use cases because they're available for all AWS customers to use. We recommend that you reduce permissions further by defining customer managed policies that are specific to your use cases.

You cannot change the permissions defined in AWS managed policies. If AWS updates the permissions defined in an AWS managed policy, the update affects all principal identities (users, groups, and roles) that the policy is attached to. AWS is most likely to update an AWS managed policy when a new AWS service is launched or new API operations become available for existing services.

For more information, see AWS managed policies in the *IAM User Guide*.

AWS-managed policies for Resource Groups

- ResourceGroupsServiceRolePolicy
- ResourceGroupsTaggingAPITagUntagSupportedResources
- ResourceGroupsTaggingAPITagUntagSupportedResources

AWS managed policy: ResourceGroupsServiceRolePolicy

You can't attach `ResourceGroupsServiceRolePolicy` to any IAM entities yourself. This policy can be attached only to a service-linked role that allows Resource Groups to perform actions on your behalf. For more information, see Using service-linked roles for Resource Groups.

This policy grants the permissions required for Resource Groups to retrieve information about the resources in your resource groups and any AWS CloudFormation stacks that those resources belong to. This lets Resource Groups generate CloudWatch Events for the group lifecycle events feature.

To see the latest version of this AWS managed policy, see `ResourceGroupsServiceRolePolicy` in the IAM console.

AWS managed policy: ResourceGroupsandTagEditorFullAccess

When you attach a policy to a principal entity, you give the entity permissions that are defined in the policy. AWS managed policies make it easier for you to assign appropriate permissions to users, groups, and roles than if you had to write the policies yourself.

This policy grants the permissions required for full access to Resource Groups and Tag Editor functionality.

To see the latest version of this AWS managed policy, see `ResourceGroupsandTagEditorFullAccess` in the IAM console.

For more information about this policy, see ResourceGroupsandTagEditorFullAccessin the *AWS Managed Policy Reference Guide*.

AWS managed policy: ResourceGroupsandTagEditorReadOnlyAccess

When you attach a policy to a principal entity, you give the entity permissions that are defined in the policy. AWS managed policies make it easier for you to assign appropriate permissions to users, groups, and roles than if you had to write the policies yourself.

This policy grants the permissions required for read only access to Resource Groups and Tag Editor functionality.

To see the latest version of this AWS managed policy, see `ResourceGroupsandTagEditorReadOnlyAccess` in the IAM console.

For more information about this policy, see ResourceGroupsandTagEditorReadOnlyAccess in the *AWS Managed Policy Reference Guide*.

AWS managed policy: ResourceGroupsTaggingAPITagUntagSupportedResources

When you attach a policy to a principal entity, you give the entity permissions that are defined in the policy. AWS managed policies make it easier for you to assign appropriate permissions to users, groups, and roles than if you had to write the policies yourself.

This policy grants the permissions required to tag and untag all of the resource types supported by AWS Resource Groups Tagging API **except** `AWS::ApiGateway`, `AWS::CloudFormation`, `AWS::CodeBuild`, and `AWS::ServiceCatalog`. Tagging and untagging these excluded resource

types requires additional, service-specific permissions which allow actions other than tagging and untagging. The following list describes which permissions are required to tag and untag the resource types excluded from the policy:

- The `AWS::ApiGateway` resource types require the `apigateway:Patch` permission on the API Gateway resource, and the tag child resource requires the `apigateway:Put`, `apigateway:Get`, `apigateway:Delete` permissions.
- The `AWS::CloudFormation` resource types require the `cloudformation:UpdateStack` and `cloudformation:UpdateStackSet` permissions.
- The `AWS::CodeBuild` resource types require the `codebuild:UpdateProject` permission.
- The `AWS::ServiceCatalog` resource types require the `servicecatalog:TagResource`, `servicecatalog:UntagResource`, `servicecatalog:UpdatePortfolio`, and `servicecatalog:UpdateProduct` permissions.

This policy also grants the permissions required to retrieve all tagged, or previously tagged, resources through the Resource Groups Tagging API.

To see the latest version of this AWS managed policy, see `ResourceGroupsTaggingAPITagUntagSupportedResources` in the IAM console.

For more information about this policy, see ResourceGroupsTaggingAPITagUntagSupportedResources in the *AWS Managed Policy Reference Guide*.

Resource Groups updates to AWS managed policies

View details about updates to AWS managed policies for Resource Groups since this service began tracking these changes. For automatic alerts about changes to this page, subscribe to the RSS feed on the Resource Groups Document history page.

Change	Description	Date
Updated policy — ResourceGroupsTaggingAPITagUntagSupportedResources	Resource Groups updated this policy to include permissions for eight new services, including Amazon Application Recovery Controller (ARC)	December 20, 2024

Change	Description	Date
	and Amazon VPC Lattice. The following permissions were added to the policy: • `kinesisvideo:TagResource` • `kinesisvideo:UntagResource` • `redshift-serverless:TagResource` • `redshift-serverless:UntagResource` • `route53-recovery-control-config:TagResource` • `route53-recovery-control-config:UntagResource` • `route53-recovery-readiness:TagResource` • `route53-recovery-readiness:UntagResource` • `ssm-contacts:TagResource` • `ssm-contacts:UntagResource` • `ssm-incidents:TagResource` • `ssm-incidents:UntagResource`	

Change	Description	Date
	• vpc-lattice:TagResource • vpc-lattice:UntagResource • workspaces-web:TagResource • workspaces-web:UntagResource	
New policy – ResourceGroupsTaggingAPITagUntagSupportedResources	Resource Groups added a new policy to provide the required permissions to tag and untag all of the resource types supported by AWS Resource Groups Tagging API.	October 11, 2024
Policy update – ResourceGroupsandTagEditorFullAccess	Resource Groups updated a policy to include additional AWS CloudFormation permissions.	August 10, 2023
Policy update – ResourceGroupsandTagEditorReadOnlyAccess	Resource Groups updated a policy to include additional AWS CloudFormation permissions.	August 10, 2023
New policy – ResourceGroupsServiceRolePolicy	Resource Groups added a new policy to support its service-linked role.	November 17, 2022
Resource Groups started tracking changes	Resource Groups started tracking changes for its AWS managed policies.	November 17, 2022

Using service-linked roles for Resource Groups

AWS Resource Groups uses AWS Identity and Access Management (IAM) service-linked roles. A service-linked role is a unique type of IAM role that is linked directly to Resource Groups. Service-linked roles are predefined by Resource Groups and include all the permissions that the service requires to call other AWS services on your behalf.

A service-linked role makes setting up Resource Groups easier because you don't have to manually add the necessary permissions. Resource Groups defines the permissions of its service-linked roles and sets trust policies on each that ensures that only the Resource Groups service can assume its roles. The defined permissions include the trust policy and the permissions policy, and that permissions policy can't be attached to any other IAM entity.

For information about other services that support service-linked roles, see AWS services that work with IAM and look for the services that have **Yes** in the **Service-linked roles** column. Choose a **Yes** with a link to view the service-linked role documentation for that service.

Service-linked role permissions for Resource Groups

Resource Groups uses the following service-linked role to support group lifecycle events. Choose the link on the role name to view the role in the IAM console after you create it.

- `AWSServiceRoleForResourceGroups`

Resource Groups uses the permissions in this role to query the AWS services that own your resources to help resolve group membership and to keep the group up-to-date. It allows Resource Groups to emit service-related events to the Amazon EventBridge service.

The `AWSServiceRoleForResourceGroups` service-linked role trusts ***only*** the following service to assume the role:

- `resourcegroups.amazonaws.com`

The permissions attached to the role come from the following AWS managed policy. Choose the link on the policy name to view the policy in the IAM console.

- `AWS managed policies for AWS Resource Groups`

Creating the service-linked role for Resource Groups

> **Important**
>
> This service-linked role can appear in your account if you complete an action in another service that requires the features supported by this role. For more information, see A new role appeared in my AWS account.

To create the service-linked role, turn on the group lifecycle events feature.

Editing a service-linked role for Resource Groups

Resource Groups doesn't allow you to edit the AWSServiceRoleForResourceGroups service-linked role. After you create a service-linked role, you can't change the name of the role because various entities might reference the role. However, you can edit the description of the role using IAM. For more information, see Editing a service-linked role in the *IAM User Guide*.

Deleting a service-linked role for Resource Groups

You can delete the service-linked role only after you turn off the group lifecycle events feature.

> **Important**
>
> - AWS prevents you from removing the service-linked role until you first turn off the group lifecycle events feature that created it.
> - We recommend that you do not delete the service-linked role as long as you have any resource groups in your AWS account. The Resource Groups service can't interact with other AWS services to manage your groups if you delete this role.

Manually delete the service-linked role

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

Console

To delete the Resource Groups service-linked role

1. Open the IAM console to the Roles page.
2. Find the role named AWSServiceRoleForResourceGroups, and select the check box beside it.
3. Choose **Delete**.
4. Confirm your intent to delete the role by entering the role's name in the box, and then choose **Delete**.

The role disappears from your list of roles in the IAM console.

AWS CLI

To delete the Resource Groups service-linked role

To delete the role, enter the following command with the parameters exactly as shown. Do not replace any of the values.

```
$ aws iam delete-service-linked-role \
    --role-name AWSServiceRoleForResourceGroups
{
    "DeletionTaskId": "task/aws-service-role/resource-groups.amazonaws.com/
AWSServiceRoleForResourceGroups/34e58943-e9a5-4220-9856-fc565EXAMPLE"
}
```

The command returns a task ID. The actual role deletion occurs asynchronously. You can check the status of the role deletion by passing the provided task identifier to the following AWS CLI command.

```
$ aws iam get-service-linked-role-deletion-status \
    --deletion-task-id "task/aws-service-role/resource-groups.amazonaws.com/
AWSServiceRoleForResourceGroups/34e58943-e9a5-4220-9856-fc565EXAMPLE"
{
    "Status": "SUCCEEDED"
}
```

Supported Regions for Resource Groups service-linked roles

Resource Groups supports using service-linked roles in all of the AWS Regions where the service is available. For more information, see AWS Regions and Endpoints.

AWS Resource Groups identity-based policy examples

By default, IAM principals, such as roles and users, 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 the principals permission to perform specific API operations on the specified resources they need. The administrator must then attach those policies to the principals 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
- Using the Resource Groups console and API
- Allow users to view their own permissions
- Viewing groups based on tags

Policy best practices

Identity-based policies 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 with AWS managed policies and move toward least-privilege permissions** – To get started granting permissions to your users and workloads, use the *AWS managed policies* that grant permissions for many common use cases. They are available in your AWS account. We recommend that you reduce permissions further by defining AWS customer managed policies that are specific to your use cases. For more information, see AWS managed policies or AWS managed policies for job functions in the *IAM User Guide*.
- **Apply least-privilege permissions** – When you set permissions with IAM policies, grant only the permissions required to perform a task. You do this by defining the actions that can be taken on specific resources under specific conditions, also known as *least-privilege permissions*. For more

information about using IAM to apply permissions, see Policies and permissions in IAM in the *IAM User Guide*.

- **Use conditions in IAM policies to further restrict access** – You can add a condition to your policies to limit access to actions and resources. For example, you can write a policy condition to specify that all requests must be sent using SSL. You can also use conditions to grant access to service actions if they are used through a specific AWS service, such as AWS CloudFormation. For more information, see IAM JSON policy elements: Condition in the *IAM User Guide*.
- **Use IAM Access Analyzer to validate your IAM policies to ensure secure and functional permissions** – IAM Access Analyzer validates new and existing policies so that the policies adhere to the IAM policy language (JSON) and IAM best practices. IAM Access Analyzer provides more than 100 policy checks and actionable recommendations to help you author secure and functional policies. For more information, see Validate policies with IAM Access Analyzer in the *IAM User Guide*.
- **Require multi-factor authentication (MFA)** – If you have a scenario that requires IAM users or a root user in your AWS account, turn on MFA for additional security. To require MFA when API operations are called, add MFA conditions to your policies. For more information, see Secure API access with MFA in the *IAM User Guide*.

For more information about best practices in IAM, see Security best practices in IAM 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 principals (IAM roles or users) 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*:

JSON

```
{
  "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, see Granting permissions for using AWS Resource Groups and Tag Editor 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:ListGroupsForUser",
                "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 resource, in this example, a resource group. However, permission is granted only if the group tag `project` has the same value as the `project` tag attached to the calling principal.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "resource-groups:GetGroup",
            "Resource": "arn:aws:resource-groups:us-
east-1:111122223333:group/group_name",
            "Condition": {
                "StringEquals": {"aws:ResourceTag/project": "${aws:PrincipalTag/
project}"}
            }
        }
    ]
```

```
}
```

You can attach this policy to the principals in your account. If a principal with the tag key `project` and tag value `alpha` attempts to view a resource group, the group must also be tagged `project=alpha`. Otherwise the user is denied access. The condition tag key `project` matches both `Project` and `project` 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
- I am not authorized to perform iam:PassRole
- I want to allow people outside of my AWS account to access my Resource Groups

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 sign-in credentials.

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

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to
 perform: resource-groups:ListGroups 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:ListGroups` 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, your policies must be updated 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 that are 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's policies must be updated to allow her to perform the `iam:PassRole` action.

If you need help, contact your AWS administrator. Your administrator is the person who provided you with your sign-in credentials.

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.
- 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 Cross account resource access in IAM 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-8081-7zz0386fbcb6",
"eventID":"8z7z18dz-6z90-47bz-87cf-e8346428zzz3",
"eventType":"AwsApiCall",
"recipientAccountId":"831000000000"
}
```

Compliance validation for Resource Groups

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

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

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

- Security Compliance & Governance – These solution implementation guides discuss architectural considerations and provide steps for deploying security and compliance features.
- HIPAA Eligible Services Reference – Lists HIPAA eligible services. Not all AWS services are HIPAA eligible.
- AWS Compliance Resources – This collection of workbooks and guides might apply to your industry and location.
- AWS Customer Compliance Guides – Understand the shared responsibility model through the lens of compliance. The guides summarize the best practices for securing AWS services and map the guidance to security controls across multiple frameworks (including National Institute of Standards and Technology (NIST), Payment Card Industry Security Standards Council (PCI), and International Organization for Standardization (ISO)).
- Evaluating Resources with Rules in the *AWS Config Developer Guide* – The AWS Config service assesses how well your resource configurations comply with internal practices, industry guidelines, and regulations.
- AWS Security Hub – This AWS service provides a comprehensive view of your security state within AWS. Security Hub uses security controls to evaluate your AWS resources and to check your compliance against security industry standards and best practices. For a list of supported services and controls, see Security Hub controls reference.
- Amazon GuardDuty – This AWS service detects potential threats to your AWS accounts, workloads, containers, and data by monitoring your environment for suspicious and malicious activities. GuardDuty can help you address various compliance requirements, like PCI DSS, by meeting intrusion detection requirements mandated by certain compliance frameworks.

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

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 AWS global network security. For information about AWS security services and how AWS protects infrastructure, see AWS Cloud Security. To design your AWS environment using the best practices for infrastructure security, see Infrastructure Protection in *Security Pillar AWS Well-Architected Framework*.

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

- Transport Layer Security (TLS). We require TLS 1.2 and recommend TLS 1.3.

- Cipher suites with perfect forward secrecy (PFS) such as DHE (Ephemeral Diffie-Hellman) or ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Most modern systems such as Java 7 and later support these modes.

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

Resource Groups does not support resource-based policies.

Access AWS Resource Groups using an interface endpoint (AWS PrivateLink)

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

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

For more information, see Access AWS services through AWS PrivateLink in the *AWS PrivateLink Guide*.

Considerations for Resource Groups

Before you set up an interface endpoint for Resource Groups, review Considerations in the *AWS PrivateLink Guide*.

Resource Groups supports making calls to all of its API actions through the interface endpoint.

Create an interface endpoint for Resource Groups

You can create an interface endpoint for Resource Groups using either the Amazon VPC console or the AWS Command Line Interface (AWS CLI). For more information, see Create an interface endpoint in the *AWS PrivateLink Guide*.

Create an interface endpoint for Resource Groups using the following service name:

```
com.amazonaws.region.resource-groups
```

If you enable private DNS for the interface endpoint, you can make API requests to Resource Groups using its default Regional DNS name. For example, `resource-groups.us-east-1.amazonaws.com`.

Create an endpoint policy for your interface endpoint

An endpoint policy is an IAM resource that you can attach to an interface endpoint. The default endpoint policy allows full access to Resource Groups through the interface endpoint. To control the access allowed to Resource Groups from your VPC, attach a custom endpoint policy to the interface endpoint.

An endpoint policy specifies the following information:

- The principals that can perform actions (AWS accounts, IAM users, and IAM roles).
- The actions that can be performed.
- The resources on which the actions can be performed.

For more information, see Control access to services using endpoint policies in the *AWS PrivateLink Guide*.

Example: VPC endpoint policy for Resource Groups actions

The following is an example of a custom endpoint policy. When you attach this policy to your interface endpoint, it grants access to the listed Resource Groups actions for all principals on all resources.

```
{
   "Statement": [
      {
         "Principal": "*",
         "Effect": "Allow",
         "Action": [
            "resource-groups:CreateGroup",
            "resource-groups:GetAccountSettings",
```

```
            "resource-groups:GetGroupQuery"
         ],
         "Resource":"*"
      }
   ]
}
```

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 your IAM principals, such as users and roles, 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.

Service quotas for Resource Groups

The following table describes quotas within AWS Resource Groups (Resource Groups). For adjustable quota, you can request an increase in the Service Quotas console.

Name	Default	Adjustable	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.

AWS Resource Groups document history

Change	Description	Date
Support for new resource types	160 more resource types are now supported by Resource Groups and Tag Editor.	April 16, 2025
AWS PrivateLink	With AWS PrivateLink for AWS Resource Groups, you can connect directly to Resource Groups by using an interface endpoint in your virtual private cloud (VPC).	April 7, 2025
Support for new resource types	172 more resource types are now supported by Resource Groups and Tag Editor.	January 22, 2025
Updated AWS managed policy ResourceGroupsTaggingAPITagUntagSupportedResources	Resource Groups updated this policy to include the following permissions: `kinesisvideo:TagResource` , `kinesisvideo:UntagResource` , `redshift-serverless:TagResource` , `redshift-serverless:UntagResource` , `route53-recovery-control-config:TagResource` , `route53-recovery-control-config:UntagResource` , `route53-recovery-readiness:TagResour`	December 11, 2024

	`ce ,route53-recovery-readiness:UntagResource ,ssm-contacts:TagResource ,ssm-contacts:UntagResource ,ssm-incidents:TagResource ,ssm-incidents:UntagResource ,vpc-lattice:TagResource ,vpc-lattice:UntagResource ,workspaces-web:TagResource ,and workspaces-web:UntagResource .`	
Support for new resource types	405 more resource types are now supported by Resource Groups and Tag Editor.	December 6, 2024
Added new AWS managed policy ResourceGroupsTaggingAPITagUntagSupportedResources	Resource Groups added a new AWS managed policy to grants the permissions required to tag and untag all of the resource types supported by AWS Resource Groups Tagging API (with exceptions). This policy also grants the permissions required to retrieve all tagged, or previously tagged, resources through the Resource Groups Tagging API.	October 11, 2024

Updated content	Updated topic titles and reorganized content to improve readability and discoverability.	August 1, 2024
Support for more resource types	More resource types are now supported by Resource Groups and Tag Editor.	May 30, 2024
Updated AWS managed polices ResourceGroupsandTagEditorFullAccess and ResourceGroupsandTagEditorReadOnlyAccess	Resource Groups updated two AWS managed policies to add additional AWS CloudFormation permissions.	August 10, 2023
Resource Groups service quotas	You can now view Resource Groups quota limits using Service Quotas.	June 29, 2023
IAM best practices update	Updated guide to align with the IAM best practices . For more information, see Security best practices in IAM.	January 3, 2023
Tag Editor information has been moved to its own guide	The documentation for Tag Editor has been removed from this guide and moved to the new Tag Editor User Guide.	December 13, 2022
Resource groups can now include resources of Amazon Keyspaces (for Apache Cassandra)	AWS Resource Groups now supports including resources for Amazon Keyspaces (for Apache Cassandra) in a resource group.	October 20, 2022

Deprecation of resource types	The following resource types are no longer supported by Tag Editor: `AWS::RoboMaker::Robot` , `AWS::RoboMaker::Fleet` , and `AWS::RoboMaker::DeploymentJob` .	May 17, 2022
New AWS managed policy - ResourceGroupsServiceRolePolicy	Resource Groups added a new AWS managed policy in AWS Identity and Access Management (IAM) to support the service's service-linked role.	January 12, 2022
Group lifecycle events	Resource Groups can now generate events in Amazon CloudWatch Events to alert you when changes happen to your resource groups.	January 12, 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	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	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 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 *AWS Resource Groups API Reference*.	September 15, 2021
Added support for resources of Amazon OpenSearch Service	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	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	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	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.	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	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 reservati ons 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 *Amazon EC2 User Guide*.	July 29, 2020
Added support for AWS IoT Greengrass resources.	More resource types are now supported by AWS Resource Groups and Tag Editor.	March 25, 2020

View operations data for AWS Resource Groups	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 informati on 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 *AWS Systems Manager User Guide*.	March 16, 2020
Check for compliance with tag policies	After you create and attach tag policies to accounts using AWS Organizations, you can find noncompliant tags on resources in your organizat ion's accounts.	November 26, 2019

Support for more resource types	More resource types are now supported by AWS Resource Groups and Tag Editor.	October 4, 2019
New resource types supported by AWS Resource Groups	More resource types are now supported by AWS Resource Groups, especially for groups based on an AWS CloudForm ation stack.	August 5, 2019
New resource types supported by AWS Resource Groups	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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