
AWS Identity and Access Management

User Guide



AWS Identity and Access Management: User Guide

Copyright © 2020 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 Is IAM?	1
Video Introduction to IAM	1
IAM Features	1
Accessing IAM	2
Understanding How IAM Works	3
Terms	4
Principal	5
Request	5
Authentication	5
Authorization	5
Actions or Operations	6
Resources	6
Overview: Users	6
First-Time Access Only: Your Root User Credentials	7
IAM Users	7
Federating Existing Users	9
Overview: Permissions and Policies	10
Policies and Accounts	10
Policies and Users	10
Policies and Groups	11
Federated Users and Roles	12
Identity-based and Resource-based Policies	12
What Is ABAC for AWS?	12
Comparing ABAC to the Traditional RBAC Model	13
Security Features Outside of IAM	14
Quick Links to Common Tasks	15
Getting Set Up	17
Using IAM to Give Users Access to Your AWS Resources	17
Do I Need to Sign Up for IAM?	18
Additional Resources	18
Getting Started	19
Creating an IAM Admin User and Group	20
Creating an Administrator IAM User and Group (Console)	20
Creating an IAM User and Group (AWS CLI)	21
Related Resources	23
Creating a Delegated User	23
Creating a Delegated IAM User and Group (Console)	20
Reducing the Group Permissions	24
How Users Sign In to Your Account	26
Permissions Required for Console Activities	26
Logging Sign-In Details in CloudTrail	26
Tutorials	27
Delegate Access to the Billing Console	27
Prerequisites	27
Step 1: Activate Access to Billing Data on Your AWS Test Account	28
Step 2: Create IAM Policies That Grant Permissions to Billing Data	28
Step 3: Attach Billing Policies to Your Groups	29
Step 4: Test Access to the Billing Console	29
Related Resources	30
Summary	30
Delegate Access Across AWS Accounts Using Roles	30
Prerequisites	31
Step 1: Create a Role	32
Step 2: Grant Access to the Role	34

Step 3: Test Access by Switching Roles	35
Related Resources	39
Summary	39
Create a Customer Managed Policy	39
Prerequisites	40
Step 1: Create the Policy	40
Step 2: Attach the Policy	40
Step 3: Test User Access	41
Related Resources	41
Summary	41
Using Tags for ABAC	41
Tutorial Overview	42
Prerequisites	43
Step 1: Create Test Users	43
Step 2: Create the ABAC Policy	44
Step 3: Create Roles	47
Step 4: Test Creating Secrets	47
Step 5: Test Viewing Secrets	49
Step 6: Test Scalability	51
Step 7: Test Updating and Deleting Secrets	52
Summary	53
Related Resources	53
Using SAML Session Tags for ABAC	53
Enable Users to Configure Their Own Credentials and MFA Settings	56
Prerequisites	56
Step 1: Create a Policy to Enforce MFA Sign-In	57
Step 2: Attach Policies to Your Test Group	57
Step 3: Test Your User's Access	58
Related Resources	59
Best Practices and Use Cases	60
Best Practices	60
Lock Away Your AWS Account Root User Access Keys	60
Create Individual IAM Users	61
Use Groups to Assign Permissions to IAM Users	61
Grant Least Privilege	61
Get Started Using Permissions with AWS Managed Policies	62
Use Customer Managed Policies Instead of Inline Policies	63
Use Access Levels to Review IAM Permissions	63
Configure a Strong Password Policy for Your Users	64
Enable MFA	64
Use Roles for Applications That Run on Amazon EC2 Instances	65
Use Roles to Delegate Permissions	65
Do Not Share Access Keys	65
Rotate Credentials Regularly	65
Remove Unnecessary Credentials	66
Use Policy Conditions for Extra Security	66
Monitor Activity in Your AWS Account	66
Video Presentation About IAM Best Practices	67
Business Use Cases	67
Initial Setup of Example Corp	68
Use Case for IAM with Amazon EC2	68
Use Case for IAM with Amazon S3	69
IAM Console and Sign-in Page	71
Main Sign-In Page	72
AWS Account Root User Sign-In Page	72
IAM User Sign-In Page	73
Controlling User Access to the AWS Management Console	75

Your AWS Account ID and Its Alias	76
Finding Your AWS Account ID	76
About Account Aliases	77
Creating, Deleting, and Listing an AWS Account Alias	77
Using MFA Devices With Your IAM Sign-in Page	78
Signing in with a Virtual MFA Device	78
Signing in with a U2F Security Key	79
Signing in with a Hardware MFA Device	79
IAM Console Search	79
Using IAM Console Search	80
Icons in the IAM Console Search Results	80
Sample Search Phrases	80
Identities	82
The AWS Account Root User	82
IAM Users	82
IAM Groups	82
IAM Roles	83
Temporary Credentials	83
When to Create an IAM User (Instead of a Role)	83
When to Create an IAM Role (Instead of a User)	83
Users	84
How AWS identifies an IAM user	84
Users and credentials	84
Users and permissions	85
Users and accounts	86
Users as service accounts	86
Adding a User	86
How IAM Users Sign In to AWS	90
Managing Users	92
Changing Permissions for a User	95
Passwords	100
Access Keys	110
Retrieving Lost Passwords or Access Keys	118
Multi-Factor Authentication (MFA)	119
Finding Unused Credentials	154
Getting Credential Reports	156
Using IAM with CodeCommit	161
Using IAM with MCS	162
Working with Server Certificates	163
Groups	168
Creating Groups	169
Managing Groups	170
Roles	175
Terms and Concepts	176
Common Scenarios	178
Identity Providers and Federation	184
Service-Linked Roles	221
Creating Roles	228
Using Roles	253
Managing Roles	278
Roles vs. Resource-based Policies	292
Tagging Users and Roles	295
Choose an AWS Tag Naming Convention	295
Rules for Tagging in IAM And AWS STS	295
Permissions Required for Tagging IAM Entities	296
Managing Tags on IAM Entities (Console)	298
Managing Tags on IAM Entities (AWS CLI or AWS API)	298

Session Tags	299
Temporary Security Credentials	307
AWS STS and AWS Regions	307
Common Scenarios for Temporary Credentials	308
Requesting Temporary Security Credentials	309
Using Temporary Credentials With AWS Resources	319
Controlling Permissions for Temporary Security Credentials	322
Managing AWS STS in an AWS Region	333
Using AWS STS Interface VPC Endpoints	336
Sample Applications That Use Temporary Credentials	337
Additional Resources for Temporary Credentials	338
The Root User	338
Enable MFA on the AWS Account Root User	339
Creating Access Keys for the Root User	339
Deleting Access Keys from the Root User	340
Changing the Root User's Password	341
Log Events with CloudTrail	341
IAM and AWS STS Information in CloudTrail	341
Logging IAM and AWS STS API Requests	342
Logging API Requests to Other AWS Services	342
Logging Regional Sign-in Events	342
Logging User Sign-in Events	344
Logging Sign-in Events for Temporary Credentials	345
Example IAM API Events in CloudTrail Log	345
Example AWS STS API Events in CloudTrail Log	346
Example Sign-In Events in CloudTrail Log	352
Access Management	355
Access Management Resources	356
Policies & Permissions	356
Policy Types	356
Policies and the Root User	361
Overview of JSON Policies	361
Managed Policies and Inline Policies	364
Permissions Boundaries	370
Identity vs Resource	379
Control Access Using Policies	381
Control Access Using IAM Tags	389
Control Access Using AWS Resource Tags	392
Example Policies	394
Managing IAM Policies	442
Creating IAM Policies	443
Validating JSON Policies	448
Testing IAM Policies	448
Add or Remove Identity Permissions	457
Versioning IAM Policies	465
Editing IAM Policies	468
Deleting IAM Policies	472
Reduce Permissions Using Access Data	475
Understanding Policies	491
Policy Summary (List of Services)	492
Service Summary (List of Actions)	502
Action Summary (List of Resources)	507
Example Policy Summaries	510
Permissions Required	517
Permissions for Administering IAM Identities	518
Permissions for Working in the AWS Management Console	519
Granting Permissions Across AWS Accounts	519

Permissions for One Service to Access Another	520
Required Actions	520
Example Policies for IAM	521
Access Analyzer	524
Supported Resource Types	525
S3 Buckets	525
IAM Roles	525
KMS Keys	525
Lambda Functions	526
SQS Queues	527
How Access Analyzer Works	527
Getting Started	527
Permissions Required to Use Access Analyzer	527
Enabling Access Analyzer	529
Access Analyzer Quotas	529
Using Service-Linked Roles	530
Access Analyzer Findings	532
Working with Findings	532
Review Findings	533
Filtering Findings	534
Archiving Findings	536
Resolving Findings	536
Archive Rules	537
Monitoring with EventBridge	538
Findings Events	538
Event Notification Frequency	538
Example Event	538
Creating an Event Rule with a Target	539
Logging with CloudTrail	541
Access Analyzer Information in CloudTrail	541
Understanding Access Analyzer Log File Entries	542
Troubleshooting IAM	543
Troubleshooting General Issues	543
I Lost My Access Keys	543
I Need to Access an Old Account	543
I Can't Sign in to My Account	544
I Get "Access Denied" When I Make a Request to an AWS Service	544
I Get "Access Denied" When I Make a Request with Temporary Security Credentials	545
Policy Variables Aren't Working	546
Changes That I Make Are Not Always Immediately Visible	546
I Am Not Authorized to Perform: <code>iam>DeleteVirtualMFADevice</code>	547
Troubleshoot Policies	547
Troubleshoot Using the Visual Editor	548
Troubleshoot Using Policy Summaries	551
Troubleshoot Policy Management	557
Troubleshoot JSON Policy Documents	558
Troubleshooting U2F Security Keys	561
I Can't Enable My U2F Security Key	562
I Can't Sign in Using My U2F Security Key	562
I Lost or Broke My U2F Key	562
Other Issues	563
Troubleshooting IAM Roles	563
I Can't Assume a Role	563
A New Role Appeared in My AWS Account	564
I Can't Edit or Delete a Role in My AWS Account	564
I'm Not Authorized to Perform: <code>iam:PassRole</code>	565
Why Can't I Assume a Role with a 12-Hour Session? (AWS CLI, AWS API)	565

My Role Has a Policy That Allows Me to Perform an Action, But I Get "Access Denied"	565
Troubleshooting Amazon EC2 and IAM	566
When Attempting to Launch an Instance, I Don't See the Role I Expected to See in the Amazon EC2 Console IAM Role List	566
The Credentials on My Instance Are for the Wrong Role	566
When I Attempt to Call the <code>AddRoleToInstanceProfile</code> , I Get an <code>AccessDenied</code> Error	567
Amazon EC2: When I Attempt to Launch an Instance with a Role, I Get an <code>AccessDenied</code> Error	567
I Can't Access the Temporary Security Credentials on My EC2 Instance	567
What Do the Errors from the <code>info</code> Document in the IAM Subtree Mean?	568
Troubleshooting Amazon S3 and IAM	569
How Do I Grant Anonymous Access to an Amazon S3 Bucket?	569
I'm Signed in as an AWS Account Root User; Why Can't I Access an Amazon S3 Bucket Under My Account?	569
Troubleshooting SAML 2.0 Federation with AWS	569
Invalid SAML response	570
RoleSessionName is required	570
Not authorized for <code>AssumeRoleWithSAML</code>	570
Invalid RoleSessionName characters	571
Invalid response signature	571
Failed to assume role	571
Could not parse metadata	571
Could not parse metadata	572
DurationSeconds exceeds MaxSessionDuration	572
How to View a SAML Response in Your Browser for Troubleshooting	572
Reference	575
IAM Identifiers	575
Friendly Names and Paths	575
IAM ARNs	576
Unique Identifiers	579
Limits	581
IAM Name Limits	581
IAM Object Limits	582
IAM and STS Character Limits	583
Services That Work with IAM	586
Compute	587
Storage	587
Database	588
Developer Tools	589
Security, Identity, & Compliance	589
Machine Learning	590
Management Tools	591
Migration & Transfer	592
Mobile	592
Networking & Content Delivery	593
Media	593
Analytics	594
Application Integration	594
Business Applications	595
Satellite	595
Internet of Things	595
Robotics	596
Blockchain	596
Game Development	596
AR & VR	596
Customer Engagement	596
End User Computing	597
Additional Resources	597

Policy Reference	597
JSON Element Reference	598
Policy Evaluation Logic	636
Policy Grammar	651
AWS Managed Policies for Job Functions	656
Global Condition Keys	665
IAM Condition Keys	676
Actions, Resources, and Condition Keys	686
Resources	1777
Users and Groups	1777
Credentials (Passwords, Access Keys, and MFA devices)	1777
Permissions and Policies	1777
Federation and Delegation	1778
IAM and Other AWS Products	1778
Using IAM with Amazon EC2	1778
Using IAM with Amazon S3	1778
Using IAM with Amazon RDS	1779
Using IAM with Amazon DynamoDB	1779
General Security Practices	1779
General Resources	1779
Making Query Requests	1781
Endpoints	1781
HTTPS Required	1782
Signing IAM API Requests	1782
Document History	1783
AWS Glossary	1788

What Is IAM?

 Follow us on Twitter

AWS Identity and Access Management (IAM) is a web service that helps you securely control access to AWS resources. You use IAM to control who is authenticated (signed in) and authorized (has permissions) to use resources.

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

Topics

- [Video Introduction to IAM \(p. 1\)](#)
- [IAM Features \(p. 1\)](#)
- [Accessing IAM \(p. 2\)](#)
- [Understanding How IAM Works \(p. 3\)](#)
- [Overview of Identity Management: Users \(p. 6\)](#)
- [Overview of Access Management: Permissions and Policies \(p. 10\)](#)
- [What Is ABAC for AWS? \(p. 12\)](#)
- [Security Features Outside of IAM \(p. 14\)](#)
- [Quick Links to Common Tasks \(p. 15\)](#)

Video Introduction to IAM

AWS Training and Certification provides a 10-minute video introduction to IAM:

[Introduction to AWS Identity and Access Management](#)

IAM Features

IAM gives you the following features:

Shared access to your AWS account

You can grant other people permission to administer and use resources in your AWS account without having to share your password or access key.

Granular permissions

You can grant different permissions to different people for different resources. For example, you might allow some users complete access to Amazon Elastic Compute Cloud (Amazon EC2), Amazon Simple Storage Service (Amazon S3), Amazon DynamoDB, Amazon Redshift, and other AWS services. For other users, you can allow read-only access to just some S3 buckets, or permission to administer just some EC2 instances, or to access your billing information but nothing else.

Secure access to AWS resources for applications that run on Amazon EC2

You can use IAM features to securely provide credentials for applications that run on EC2 instances. These credentials provide permissions for your application to access other AWS resources. Examples include S3 buckets and DynamoDB tables.

Multi-factor authentication (MFA)

You can add two-factor authentication to your account and to individual users for extra security. With MFA you or your users must provide not only a password or access key to work with your account, but also a code from a specially configured device.

Identity federation

You can allow users who already have passwords elsewhere—for example, in your corporate network or with an internet identity provider—to get temporary access to your AWS account.

Identity information for assurance

If you use [AWS CloudTrail](#), you receive log records that include information about those who made requests for resources in your account. That information is based on IAM identities.

PCI DSS Compliance

IAM supports the processing, storage, and transmission of credit card data by a merchant or service provider, and has been validated as being compliant with Payment Card Industry (PCI) Data Security Standard (DSS). For more information about PCI DSS, including how to request a copy of the AWS PCI Compliance Package, see [PCI DSS Level 1](#).

Integrated with many AWS services

For a list of AWS services that work with IAM, see [AWS Services That Work with IAM \(p. 586\)](#).

Eventually Consistent

IAM, like many other AWS services, is [eventually consistent](#). IAM achieves high availability by replicating data across multiple servers within Amazon's data centers around the world. If a request to change some data is successful, the change is committed and safely stored. However, the change must be replicated across IAM, which can take some time. Such changes include creating or updating users, groups, roles, or policies. We recommend that you do not include such IAM changes in the critical, high-availability code paths of your application. Instead, make IAM changes in a separate initialization or setup routine that you run less frequently. Also, be sure to verify that the changes have been propagated before production workflows depend on them. For more information, see [Changes That I Make Are Not Always Immediately Visible \(p. 546\)](#).

Free to use

AWS Identity and Access Management (IAM) and AWS Security Token Service (AWS STS) are features of your AWS account offered at no additional charge. You are charged only when you access other AWS services using your IAM users or AWS STS temporary security credentials. For information about the pricing of other AWS products, see the [Amazon Web Services pricing page](#).

Accessing IAM

You can work with AWS Identity and Access Management in any of the following ways.

AWS Management Console

The console is a browser-based interface to manage IAM and AWS resources. For more information about accessing IAM through the console, see [The IAM Console and Sign-In Page \(p. 71\)](#). For a tutorial that guides you through using the console, see [Creating Your First IAM Admin User and Group \(p. 20\)](#).

AWS Command Line Tools

You can use the AWS command line tools to issue commands at your system's command line to perform IAM and AWS tasks. Using the command line can be faster and more convenient than the console. The command line tools are also useful if you want to build scripts that perform AWS tasks.

AWS provides two sets of command line tools: the [AWS Command Line Interface](#) (AWS CLI) and the [AWS Tools for Windows PowerShell](#). For information about installing and using the AWS CLI, see the [AWS Command Line Interface User Guide](#). For information about installing and using the Tools for Windows PowerShell, see the [AWS Tools for Windows PowerShell User Guide](#).

AWS SDKs

AWS provides SDKs (software development kits) that consist of libraries and sample code for various programming languages and platforms (Java, Python, Ruby, .NET, iOS, Android, etc.). The SDKs provide a convenient way to create programmatic access to IAM and AWS. For example, the SDKs take care of tasks such as cryptographically signing requests, managing errors, and retrying requests automatically. For information about the AWS SDKs, including how to download and install them, see the [Tools for Amazon Web Services](#) page.

IAM HTTPS API

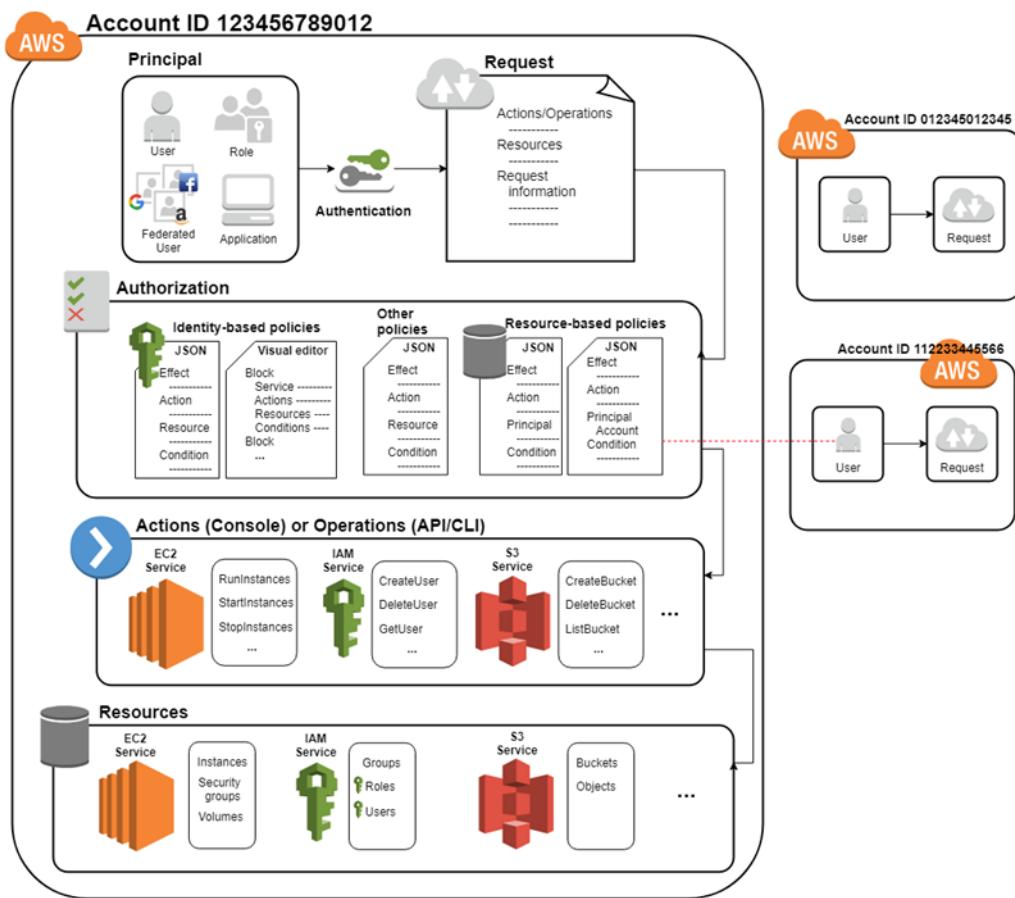
You can access IAM and AWS programmatically by using the IAM HTTPS API, which lets you issue HTTPS requests directly to the service. When you use the HTTPS API, you must include code to digitally sign requests using your credentials. For more information, see [Calling the API by Making HTTP Query Requests \(p. 1781\)](#) and the [IAM API Reference](#).

Understanding How IAM Works

Before you create users, you should understand how IAM works. IAM provides the infrastructure necessary to control authentication and authorization for your account. The IAM infrastructure includes the following elements:

Topics

- [Terms \(p. 4\)](#)
- [Principal \(p. 5\)](#)
- [Request \(p. 5\)](#)
- [Authentication \(p. 5\)](#)
- [Authorization \(p. 5\)](#)
- [Actions or Operations \(p. 6\)](#)
- [Resources \(p. 6\)](#)



Terms

Learn more about IAM terms.

Resources

The user, group, role, policy, and identity provider objects that are stored in IAM. As with other AWS services, you can add, edit, and remove resources from IAM.

Identities

The IAM resource objects that are used to identify and group. You can attach a policy to an IAM identity. These include users, groups, and roles.

Entities

The IAM resource objects that AWS uses for authentication. These include users and roles. Roles can be assumed by IAM users and roles in your or another account. They can also be assumed by users federated through a web identity or SAML.

Principals

A person or application that uses the AWS account root user, an IAM user, or an IAM role to sign in and make requests to AWS.

Principal

A *principal* is a person or application that can make a request for an action or operation on an AWS resource. The principal is authenticated as the AWS account root user or an IAM entity to make requests to AWS. As a best practice, do not use your root user credentials for your daily work. Instead, create IAM entities (users and roles). You can also support federated users or programmatic access to allow an application to access your AWS account.

Request

When a principal tries to use the AWS Management Console, the AWS API, or the AWS CLI, that principal sends a *request* to AWS. The request includes the following information:

- **Actions or operations** – The actions or operations that the principal wants to perform. This can be an action in the AWS Management Console, or an operation in the AWS CLI or AWS API.
- **Resources** – The AWS resource object upon which the actions or operations are performed.
- **Principal** – The person or application that used an entity (user or role) to send the request. Information about the principal includes the policies that are associated with the entity that the principal used to sign in.
- **Environment data** – Information about the IP address, user agent, SSL enabled status, or the time of day.
- **Resource data** – Data related to the resource that is being requested. This can include information such as a DynamoDB table name or a tag on an Amazon EC2 instance.

AWS gathers the request information into a *request context*, which is used to evaluate and authorize the request.

Authentication

A principal must be authenticated (signed in to AWS) using their credentials to send a request to AWS. Some services, such as Amazon S3 and AWS STS, allow a few requests from anonymous users. However, they are the exception to the rule.

To authenticate from the console as a root user, you must sign in with your email address and password. As an IAM user, provide your account ID or alias, and then your user name and password. To authenticate from the API or AWS CLI, you must provide your access key and secret key. You might also be required to provide additional security information. For example, AWS recommends that you use multi-factor authentication (MFA) to increase the security of your account. To learn more about the IAM entities that AWS can authenticate, see [IAM Users \(p. 84\)](#) and [IAM Roles \(p. 175\)](#).

Authorization

You must also be authorized (allowed) to complete your request. During authorization, AWS uses values from the request context to check for policies that apply to the request. It then uses the policies to determine whether to allow or deny the request. Most policies are stored in AWS as [JSON documents \(p. 361\)](#) and specify the permissions for principal entities. There are [several types of policies \(p. 356\)](#) that can affect whether a request is authorized. To provide your users with permissions to access the AWS resources in their own account, you need only identity-based policies. Resource-based policies are popular for granting [cross-account access \(p. 519\)](#). The other policy types are advanced features and should be used carefully.

AWS checks each policy that applies to the context of your request. If a single permissions policy includes a denied action, AWS denies the entire request and stops evaluating. This is called an *explicit deny*.

Because requests are *denied by default*, AWS authorizes your request only if every part of your request is allowed by the applicable permissions policies. The evaluation logic for a request within a single account follows these general rules:

- By default, all requests are denied. (In general, requests made using the AWS account root user credentials for resources in the account are always allowed.)
- An explicit allow in any permissions policy (identity-based or resource-based) overrides this default.
- The existence of an Organizations SCP, IAM permissions boundary, or a session policy overrides the allow. If one or more of these policy types exists, they must all allow the request. Otherwise, it is implicitly denied.
- An explicit deny in any policy overrides any allows.

To learn more about how all types of policies are evaluated, see [Policy Evaluation Logic \(p. 636\)](#). If you need to make a request in a different account, a policy in the other account must allow you to access the resource *and* the IAM entity that you use to make the request must have an identity-based policy that allows the request.

Actions or Operations

After your request has been authenticated and authorized, AWS approves the actions or operations in your request. Operations are defined by a service, and include things that you can do to a resource, such as viewing, creating, editing, and deleting that resource. For example, IAM supports approximately 40 actions for a user resource, including the following actions:

- `CreateUser`
- `DeleteUser`
- `GetUser`
- `UpdateUser`

To allow a principal to perform an operation, you must include the necessary actions in a policy that applies to the principal or the affected resource. To see a list of actions, resource types, and condition keys supported by each service, see [Actions, Resources, and Condition Keys for AWS Services \(p. 686\)](#).

Resources

After AWS approves the operations in your request, they can be performed on the related resources within your account. A resource is an object that exists within a service. Examples include an Amazon EC2 instance, an IAM user, and an Amazon S3 bucket. The service defines a set of actions that can be performed on each resource. If you create a request to perform an unrelated action on a resource, that request is denied. For example, if you request to delete an IAM role but provide an IAM group resource, the request fails. To see AWS service tables that identify which resources are affected by an action, see [Actions, Resources, and Condition Keys for AWS Services \(p. 686\)](#).

Overview of Identity Management: Users

For greater security and organization, you can give access to your AWS account to specific users—identities that you create with custom permissions. You can further simplify access for those users by federating existing identities into AWS.

Topics

- [First-Time Access Only: Your Root User Credentials \(p. 7\)](#)
- [IAM Users \(p. 7\)](#)
- [Federating Existing Users \(p. 9\)](#)

First-Time Access Only: Your Root User Credentials

When you create an AWS account, you create an AWS account root user identity, which you use to sign in to AWS. You can sign in to the AWS Management Console using this root user identity—that is, the email address and password that you provided when creating the account. This combination of your email address and password is also called your *root user credentials*.

When you use your root user credentials, you have complete, unrestricted access to all resources in your AWS account, including access to your billing information and the ability to change your password. This level of access is necessary when you first set up your account. However, we recommend that you **don't** use root user credentials for everyday access. We especially recommend that you do not share your root user credentials with anyone, because doing so gives them unrestricted access to your account. It is not possible to restrict the permissions that are granted to the root user.

The following sections explain how you can use IAM to create and manage user identity and permissions to provide secure, limited access to your AWS resources, both for yourself and for others who need to work with your AWS resources.

IAM Users

The "identity" aspect of AWS Identity and Access Management (IAM) helps you with the question "Who is that user?", often referred to as *authentication*. Instead of sharing your root user credentials with others, you can create individual IAM users within your account that correspond to users in your organization. IAM users are not separate accounts; they are users within your account. Each user can have its own password for access to the AWS Management Console. You can also create an individual access key for each user so that the user can make programmatic requests to work with resources in your account. In the following figure, the users Li, Mateo, DevApp1, DevApp2, TestApp1, and TestApp2 have been added to a single AWS account. Each user has its own credentials.



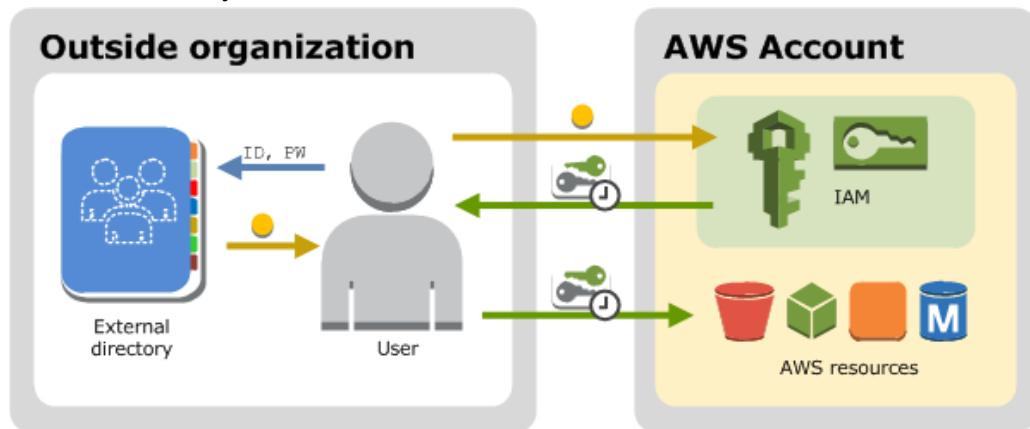
Notice that some of the users are actually applications (for example, DevApp1). An IAM user doesn't have to represent an actual person; you can create an IAM user in order to generate an access key for an application that runs in your corporate network and needs AWS access.

We recommend that you create an IAM user for yourself and then assign yourself administrative permissions for your account. You can then sign in as that user to add more users as needed.

Federating Existing Users

If the users in your organization already have a way to be authenticated, such as by signing in to your corporate network, you don't have to create separate IAM users for them. Instead, you can *federate* those user identities into AWS.

The following diagram shows how a user can use IAM to get temporary AWS security credentials to access resources in your AWS account.



Federation is particularly useful in these cases:

- **Your users already have identities in a corporate directory.**

If your corporate directory is compatible with Security Assertion Markup Language 2.0 (SAML 2.0), you can configure your corporate directory to provide single-sign on (SSO) access to the AWS Management Console for your users. For more information, see [Common Scenarios for Temporary Credentials \(p. 308\)](#).

If your corporate directory is not compatible with SAML 2.0, you can create an identity broker application to provide single-sign on (SSO) access to the AWS Management Console for your users. For more information, see [Enabling Custom Identity Broker Access to the AWS Console \(p. 213\)](#).

If your corporate directory is Microsoft Active Directory, you can use [AWS Directory Service](#) to establish trust between your corporate directory and your AWS account.

- **Your users already have Internet identities.**

If you are creating a mobile app or web-based app that can let users identify themselves through an Internet identity provider like Login with Amazon, Facebook, Google, or any OpenID Connect (OIDC) compatible identity provider, the app can use federation to access AWS. For more information, see [About Web Identity Federation \(p. 185\)](#).

Tip

To use identity federation with Internet identity providers, we recommend you use [Amazon Cognito](#).

Overview of Access Management: Permissions and Policies

The access management portion of AWS Identity and Access Management (IAM) helps you define what a principal entity is allowed to do in an account. A principal entity is a person or application that is authenticated using an IAM entity (user or role). Access management is often referred to as *authorization*. You manage access in AWS by creating policies and attaching them to IAM identities (users, groups of users, or roles) or AWS 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 uses an IAM entity (user or role) to make 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 policy types and uses, see [Policies and Permissions \(p. 356\)](#).

Policies and Accounts

If you manage a single account in AWS, then you define the permissions within that account using policies. If you manage permissions across multiple accounts, it is more difficult to manage permissions for your users. You can use IAM roles, resource-based policies, or access control lists (ACLs) for cross-account permissions. However, if you own multiple accounts, we instead recommend using the AWS Organizations service to help you manage those permissions. For more information, see [What is AWS Organizations?](#) in the *Organizations User Guide*.

Policies and Users

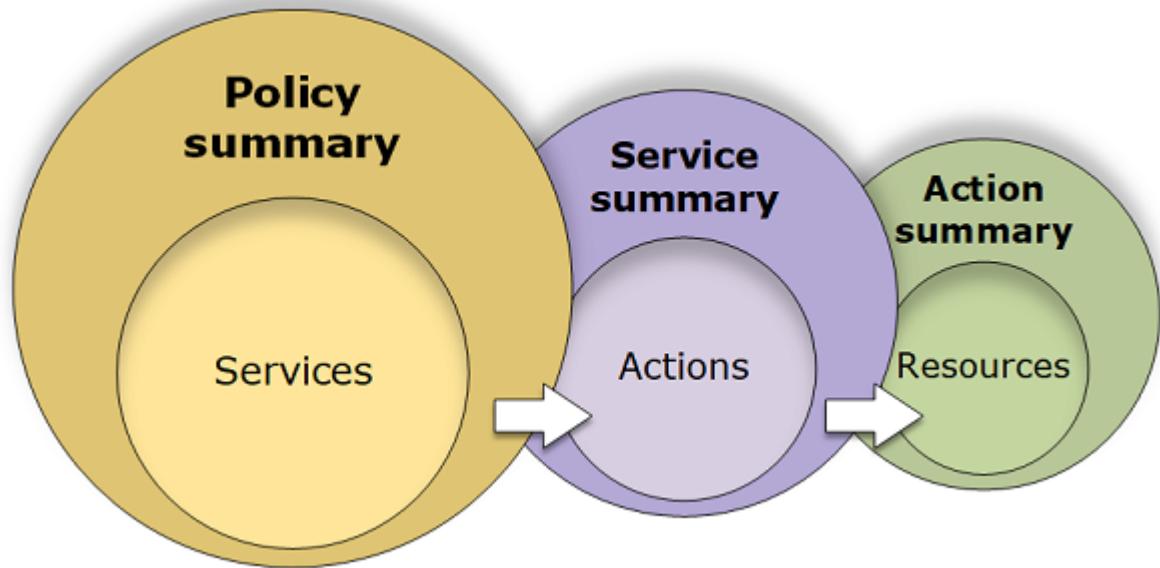
IAM users are identities in the service. When you create an IAM user, they can't access anything in your account until you give them permission. You give permissions to a user by creating an identity-based policy, which is a policy that is attached to the user or a group to which the user belongs. The following example shows a JSON policy that allows the user to perform all Amazon DynamoDB actions (`dynamodb : *`) on the `Books` table in the `123456789012` account within the `us-east-2` Region.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": "dynamodb:*",  
            "Resource": "arn:aws:dynamodb:us-east-2:123456789012:table/Books"  
        }  
    ]  
}
```

After you attach this policy to your IAM user, the user only has those DynamoDB permissions. Most users have multiple policies that together represent the permissions for that user.

Actions or resources that are not explicitly allowed are denied by default. For example, if the preceding policy is the only policy that is attached to a user, then that user is allowed to only perform DynamoDB actions on the `Books` table. Actions on all other tables are prohibited. Similarly, the user is not allowed to perform any actions in Amazon EC2, Amazon S3, or in any other AWS service. The reason is that permissions to work with those services are not included in the policy.

The IAM console includes *policy summary* tables that describe the access level, resources, and conditions that are allowed or denied for each service in a policy. Policies are summarized in three tables: the [policy summary \(p. 492\)](#), the [service summary \(p. 502\)](#), and the [action summary \(p. 507\)](#). The *policy summary* table includes a list of services. Choose a service there to see the *service summary*. This summary table includes a list of the actions and associated permissions for the chosen service. You can choose an action from that table to view the *action summary*. This table includes a list of resources and conditions for the chosen action.



You can view policy summaries on the **Users** page for all policies (managed and inline) that are attached to that user. View summaries on the **Policies** page for all managed policies.

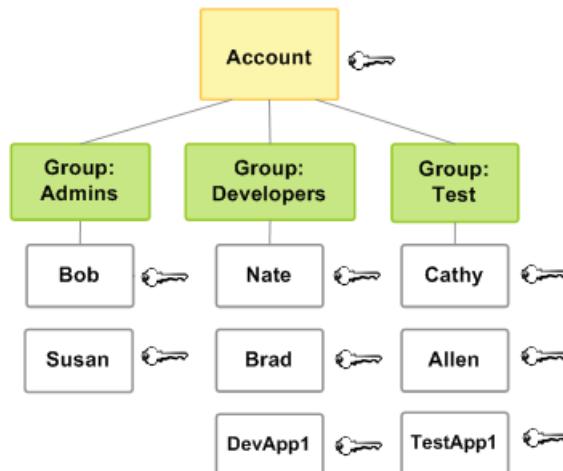
For example, the previous policy is summarized in the AWS Management Console as follows:

Service	Access level	Resource	Request condition
Allow (1 of 102 services) Show remaining 101			
DynamoDB	Full access	TableName = Books	None

You can also view the JSON document for the policy. For information about viewing the summary or JSON document, see [Understanding Permissions Granted by a Policy \(p. 491\)](#).

Policies and Groups

You can organize IAM users into *IAM groups* and attach a policy to a group. In that case, individual users still have their own credentials, but all the users in a group have the permissions that are attached to the group. Use groups for easier permissions management, and to follow our [IAM Best Practices \(p. 60\)](#).



Users or groups can have multiple policies attached to them that grant different permissions. In that case, the users' permissions are calculated based on the combination of policies. But the basic principle still applies: If the user has not been granted an explicit permission for an action and a resource, the user does not have those permissions.

Federated Users and Roles

Federated users don't have permanent identities in your AWS account the way that IAM users do. To assign permissions to federated users, you can create an entity referred to as a *role* and define permissions for the role. When a federated user signs in to AWS, the user is associated with the role and is granted the permissions that are defined in the role. For more information, see [Creating a Role for a Third-Party Identity Provider \(Federation\) \(p. 242\)](#).

Identity-based and Resource-based Policies

Identity-based policies are permissions policies that you attach to an IAM identity, such as an IAM user, group, or role. Resource-based policies are permissions policies that you attach to a resource such as an Amazon S3 bucket or an IAM role trust policy.

Identity-based policies control what actions the identity can perform, on which resources, and under what conditions. Identity-based policies can be further categorized:

- **Managed policies** – Standalone identity-based policies that you can attach to multiple users, groups, and roles in your AWS account. You can use two types of managed policies:
 - **AWS managed policies** – Managed policies that are created and managed by AWS. If you are new to using policies, we recommend that you start by using AWS managed policies.
 - **Customer managed policies** – Managed policies that you create and manage in your AWS account. Customer managed policies provide more precise control over your policies than AWS managed policies. You can create and edit an IAM policy in the visual editor or by creating the JSON policy document directly. For more information, see [Creating IAM Policies \(p. 443\)](#) and [Editing IAM Policies \(p. 468\)](#).
- **Inline policies** – Policies that you create and manage and that are embedded directly into a single user, group, or role. In most cases, we don't recommend using inline policies.

Resource-based policies control what actions a specified principal can perform on that resource and under what conditions. Resource-based policies are inline policies, and there are no managed resource-based policies. To enable cross-account access, you can specify an entire account or IAM entities in another account as the principal in a resource-based policy.

The IAM service supports only one type of resource-based policy called a role *trust policy*, which is attached to an IAM role. Because an IAM role is both an identity and a resource that supports resource-based policies, you must attach both a trust policy and an identity-based policy to an IAM role. Trust policies define which principal entities (accounts, users, roles, and federated users) can assume the role. To learn how IAM roles are different from other resource-based policies, see [How IAM Roles Differ from Resource-based Policies \(p. 292\)](#).

To see which services support resource-based policies, see [AWS Services That Work with IAM \(p. 586\)](#). To learn more about resource-based policies, see [Identity-Based Policies and Resource-Based Policies \(p. 379\)](#).

What Is ABAC for AWS?

Attribute-based access control (ABAC) is an authorization strategy that defines permissions based on attributes. In AWS, these attributes are called *tags*. Tags can be attached to IAM principals (users or roles)

and to AWS resources. You can create a single ABAC policy or small set of policies for your IAM principals. These ABAC policies can be designed to allow operations when the principal's tag matches the resource tag. ABAC is helpful in environments that are growing rapidly and helps with situations where policy management becomes cumbersome.

For example, you can create three roles with the `access-project` tag key. Set the tag value of the first role to `Heart`, the second to `Sun`, and the third to `Lightning`. You can then use a single policy that allows access when the role and the resource are tagged with the same value for `access-project`. For a detailed tutorial that demonstrates how to use ABAC in AWS, see [Tutorial: Using Tags for Attribute-Based Access Control in AWS \(p. 41\)](#).

Comparing ABAC to the Traditional RBAC Model

The traditional authorization model used in IAM is called role-based access control (RBAC). RBAC defines permissions based on a person's job function, known outside of AWS as a *role*. Within AWS a role usually refers to an IAM role, which is an identity in IAM that you can assume. IAM does include [managed policies for job functions \(p. 656\)](#) that align permissions to a job function in an RBAC model.

In IAM, you implement RBAC by creating different policies for different job functions. You then attach the policies to identities (IAM users, groups of users, or IAM roles). As a best practice, you grant the minimum permissions necessary for the job function. This is known as [granting least privilege \(p. 61\)](#). Do this by listing the specific resources that the job function can access. The disadvantage to using the traditional RBAC model is that when employees add new resources, you must update policies to allow access to those resources.

For example, assume that you have three projects, named `Heart`, `Sun`, and `Lightning`, on which your employees work. You create an IAM role for each project. You then attach policies to each IAM role to define the resources that anyone allowed to assume the role can access. If an employee changes jobs within your company, you assign them to a different IAM role. People or programs can be assigned to more than one role. However, the `Sun` project might require additional resources, such as a new Amazon S3 bucket. In that case, you must update the policy attached to the `Sun` role to specify the new bucket resource. Otherwise, `Sun` project members are not allowed to access the new bucket.

ABAC provides the following advantages over the traditional RBAC model:

- **ABAC permissions scale with innovation.** It's no longer necessary for an administrator to update existing policies to allow access to new resources. For example, assume that you designed your ABAC strategy with the `access-project` tag. A developer uses the role with the `access-project = Heart` tag. When people on the `Heart` project need additional Amazon EC2 resources, the developer can create new Amazon EC2 instances with the `access-project = Heart` tag. Then anyone on the `Heart` project can start and stop those instances because their tag values match.
- **ABAC requires fewer policies.** Because you don't have to create different policies for different job functions, you create fewer policies. Those policies are easier to manage.
- **Using ABAC, teams can change and grow quickly.** This is because permissions for new resources are automatically granted based on attributes. For example, if your company already supports the `Heart` and `Sun` projects using ABAC, it's easy to add a new `Lightning` project. An IAM administrator creates a new role with the `access-project = Lightning` tag. It's not necessary to change the policy to support a new project. Anyone that has permissions to assume the role can create and view instances tagged with `access-project = Lightning`. Additionally, a team member might move from the `Heart` project to the `Lightning` project. The IAM administrator assigns the user to a different IAM role. It's not necessary to change the permissions policies.
- **Granular permissions are possible using ABAC.** When you create policies, it's a best practice to [grant least privilege \(p. 61\)](#). Using traditional RBAC, you must write a policy that allows access to only specific resources. However, when you use ABAC, you can allow actions on all resources, but only if the resource tag matches the principal's tag.
- **Use employee attributes from your corporate directory with ABAC.** You can configure your SAML-based or web identity provider to pass session tags to AWS. When your employees federate into AWS,

their attributes are applied to their resulting principal in AWS. You can then use ABAC to allow or deny permissions based on those attributes.

For a detailed tutorial that demonstrates how to use ABAC in AWS, see [Tutorial: Using Tags for Attribute-Based Access Control in AWS \(p. 41\)](#).

Security Features Outside of IAM

You use IAM to control access to tasks that are performed using the AWS Management Console, the [AWS Command Line Tools](#), or service API operations using the [AWS SDKs](#). Some AWS products have other ways to secure their resources as well. The following list provides some examples, though it is not exhaustive.

Amazon EC2

In Amazon Elastic Compute Cloud you log into an instance with a key pair (for Linux instances) or using a user name and password (for Microsoft Windows instances).

For more information, see the following documentation:

- [Getting Started with Amazon EC2 Linux Instances](#) in the *Amazon EC2 User Guide for Linux Instances*
- [Getting Started with Amazon EC2 Windows Instances](#) in the *Amazon EC2 User Guide for Windows Instances*

Amazon RDS

In Amazon Relational Database Service you log into the database engine with a user name and password that are tied to that database.

For more information, see [Getting Started with Amazon RDS](#) in the *Amazon RDS User Guide*.

Amazon EC2 and Amazon RDS

In Amazon EC2 and Amazon RDS you use security groups to control traffic to an instance or database.

For more information, see the following documentation:

- [Amazon EC2 Security Groups for Linux Instances](#) in the *Amazon EC2 User Guide for Linux Instances*
- [Amazon EC2 Security Groups for Windows Instances](#) in the *Amazon EC2 User Guide for Windows Instances*
- [Amazon RDS Security Groups](#) in the *Amazon RDS User Guide*

Amazon WorkSpaces

In Amazon WorkSpaces, users sign in to a desktop with a user name and password.

For more information, see [Getting Started with Amazon WorkSpaces](#) in the *Amazon WorkSpaces Administration Guide*.

Amazon WorkDocs

In Amazon WorkDocs, users get access to shared documents by signing in with a user name and password.

For more information, see [Getting Started with Amazon WorkDocs](#) in the *Amazon WorkDocs Administration Guide*.

These access control methods are not part of IAM. IAM lets you control how these AWS products are administered—creating or terminating an Amazon EC2 instance, setting up new Amazon WorkSpaces

desktops, and so on. That is, IAM helps you control the tasks that are performed by making requests to Amazon Web Services, and it helps you control access to the AWS Management Console. However, IAM does not help you manage security for tasks like signing in to an operating system (Amazon EC2), database (Amazon RDS), desktop (Amazon WorkSpaces), or collaboration site (Amazon WorkDocs).

When you work with a specific AWS product, be sure to read the documentation to learn the security options for all the resources that belong to that product.

Quick Links to Common Tasks

Use the following links to get help with common tasks associated with IAM.

Sign in as an IAM user

See [How IAM Users Sign In to AWS \(p. 90\)](#).

Manage passwords for IAM users

You need a password in order to access the AWS Management Console, including access to billing information.

For your AWS account root user, see [Changing the AWS Account Root User Password \(p. 100\)](#).

For an IAM user, see [Managing Passwords for IAM Users \(p. 104\)](#).

Manage permissions for IAM users

You use policies to grant permissions to the IAM users in your AWS account. IAM users have no permissions when they are created, so you must add permissions to allow them to use AWS resources.

For more information, see [Managing IAM Policies \(p. 442\)](#).

List the users in your AWS account and get information about their credentials

See [Getting Credential Reports for Your AWS Account \(p. 156\)](#).

Add multi-factor authentication (MFA)

To add a virtual MFA device, see one of the following:

- [Enable a Virtual MFA Device for Your AWS Account Root User \(Console\) \(p. 123\)](#)
- [Enable a Virtual MFA Device for an IAM User \(Console\) \(p. 122\)](#)

To add a U2F security key, see one of the following:

- [Enable a U2F Security Key for the AWS Account Root User \(Console\) \(p. 128\)](#)
- [Enable a U2F Security Key for Another IAM User \(Console\) \(p. 127\)](#)

To add a hardware MFA device, see one of the following:

- [Enable a Hardware MFA Device for the AWS Account Root User \(Console\) \(p. 133\)](#)
- [Enable a Hardware MFA Device for Another IAM User \(Console\) \(p. 132\)](#)

Get an access key

You need an access key if you want to make AWS requests using the AWS SDKs, the [AWS Command Line Tools](#), or the API operations.

Important

You can view and download your secret access key *only* when you create the access key. You cannot view or recover a secret access key later. However, if you lose your secret access key, you can create a new access key.

For your AWS account, see [Managing Access Keys for your AWS Account](#).

For an IAM user, see [Managing Access Keys for IAM Users \(p. 110\)](#).

Tag a user or role

You can tag an IAM user or role using the IAM console, the AWS CLI, or the API through one of the AWS SDKs.

To learn about tags in IAM, see [Tagging IAM Users and Roles \(p. 295\)](#).

For details about how to manage tags in IAM, see [Managing Tags on IAM Entities \(Console\) \(p. 298\)](#).

To learn about using IAM tags to control access to AWS, see [Controlling Access to and for IAM Users and Roles Using IAM Resource Tags \(p. 389\)](#).

Get started with all of AWS

This set of documentation deals primarily with the IAM service. To learn about getting started with AWS and using multiple services to solve a problem such as building and launching your first project, see the [Getting Started Resource Center](#).

Getting Set Up

AWS Identity and Access Management (IAM) helps you securely control access to Amazon Web Services (AWS) and your account resources. IAM can also keep your account credentials private. With IAM, you can create multiple IAM users under the umbrella of your AWS account or enable temporary access through identity federation with your corporate directory. In some cases, you can also enable access to resources across AWS accounts.

Without IAM, however, you must either create multiple AWS accounts—each with its own billing and subscriptions to AWS products—or your employees must share the security credentials of a single AWS account. In addition, without IAM, you cannot control the tasks a particular user or system can do and what AWS resources they might use.

This guide provides a conceptual overview of IAM, describes business use cases, and explains AWS permissions and policies.

Topics

- [Using IAM to Give Users Access to Your AWS Resources \(p. 17\)](#)
- [Do I Need to Sign Up for IAM? \(p. 18\)](#)
- [Additional Resources \(p. 18\)](#)

Using IAM to Give Users Access to Your AWS Resources

Here are the ways you can use IAM to control access to your AWS resources.

Type of access	Why would I use it?	Where can I get more information?
Access for users in your AWS account	You want to add users under the umbrella of your AWS account, and you want to use IAM to create users and manage their permissions.	To learn how to use the AWS Management Console to create users and to manage their permissions in your AWS account, see Getting Started (p. 19) . To learn about using the IAM API or AWS Command Line Interface to create users in your AWS account, see Creating Your First IAM Admin User and Group (p. 20) . For more information about working with IAM users, see Identities (Users, Groups, and Roles) (p. 82) .
Non-AWS user access via identity federation between your authorization system and AWS	You have non-AWS users in your identity and authorization system, and they need access to your AWS resources.	To learn how to use security tokens to give your users access to your AWS account resources through federation with your corporate directory, go to Temporary Security Credentials (p. 307) . For information about the AWS Security Token Service API, go to the AWS Security Token Service API Reference .

Type of access	Why would I use it?	Where can I get more information?
Cross-account access between AWS accounts	You want to share access to certain AWS resources with users under other AWS accounts.	To learn how to use IAM to grant permissions to other AWS accounts, see Roles Terms and Concepts (p. 176) .

Do I Need to Sign Up for IAM?

If you don't already have an AWS account, you need to create one to use IAM. You don't need to specifically sign up to use IAM. There is no charge to use IAM.

Note

IAM works only with AWS products that are integrated with IAM. For a list of services that support IAM, see [AWS Services That Work with IAM \(p. 586\)](#).

To sign up for AWS

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

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

Additional Resources

Here are some resources to help you get things done with IAM.

- Manage your AWS account credentials: [AWS Security Credentials](#) in the [AWS General Reference](#)
- Get started with and learn more about [What Is IAM? \(p. 1\)](#)
- Set up a command line interface (CLI) to use with IAM. For the cross-platform AWS CLI, see the [AWS Command Line Interface Documentation](#) and [IAM CLI reference](#). You can also manage IAM with Windows PowerShell; see the [AWS Tools for Windows PowerShell Documentation](#) and [IAM Windows PowerShell reference](#).
- Download an AWS SDK for convenient programmatic access to IAM: [Tools for Amazon Web Services](#)
- Get the FAQ: [AWS Identity and Access Management FAQ](#)
- Get technical support: [AWS Support Center](#)
- Get premium technical support: [AWS Premium Support Center](#)
- Find definitions of AWS terms: [Amazon Web Services Glossary](#)
- Get community support: [IAM Discussion Forums](#)
- Contact AWS: [Contact Us](#)

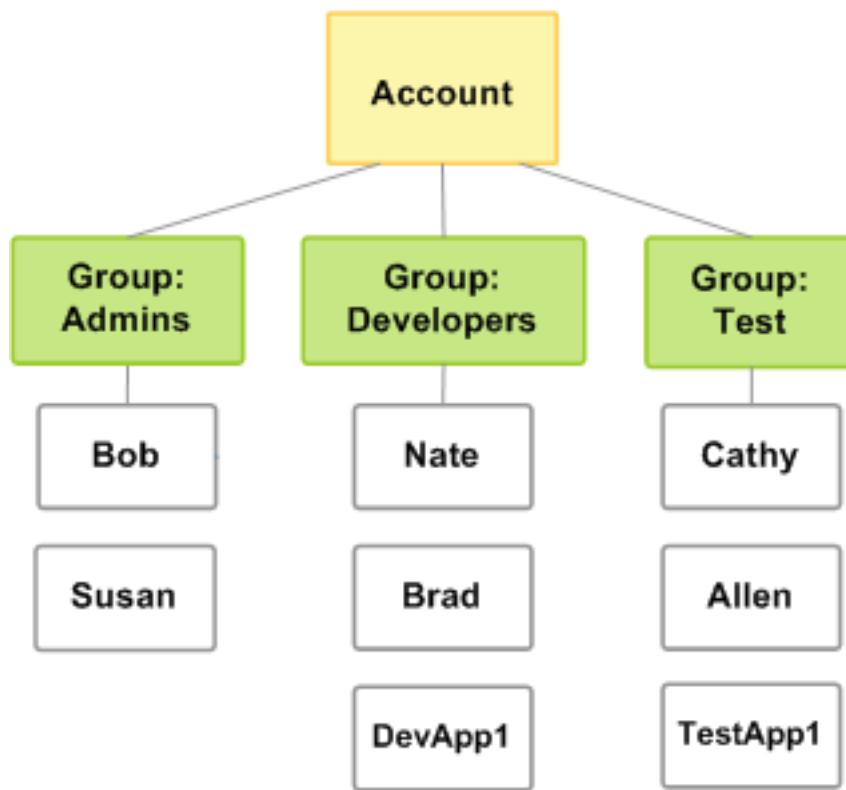
Getting Started

This topic shows you how to give access to your AWS resources by creating AWS Identity and Access Management (IAM) users in your AWS account. First, you'll learn about IAM concepts you should understand before you create groups and users, and then you'll walk through how to perform the necessary tasks using the AWS Management Console. The first task is to set up an administrators group for your AWS account. Having an administrators group for your AWS account isn't required, but we strongly recommend it.

Note

This set of documentation deals primarily with the IAM service. To learn about getting started with AWS and using multiple services to solve a problem such as building and launching your first project, see the [Getting Started Resource Center](#).

The following figure shows a simple example of an AWS account with three groups. A group is a collection of users who have similar responsibilities. In this example, one group is for administrators (it's called *Admins*). There's also a *Developers* group and a *Test* group. Each group has multiple users. Each user can be in more than one group, although the figure doesn't illustrate that. You can't put groups inside other groups. You use policies to grant permissions to groups.



In the procedure that follows, you will perform the following tasks:

- Create an Administrators group and give the group permission to access all of your AWS account's resources.
- Create a user for yourself and add that user to the Administrators group.
- Create a password for your user so you can sign in to the AWS Management Console.

You will grant the Administrators group permission to access all your available AWS account resources. Available resources are any AWS products you use, or that you are signed up for. Users in the Administrators group can also access your AWS account information, *except* for your AWS account's security credentials.

Topics

- [Creating Your First IAM Admin User and Group \(p. 20\)](#)
- [Creating Your First IAM Delegated User and Group \(p. 23\)](#)
- [How Users Sign In to Your Account \(p. 26\)](#)

Creating Your First IAM Admin User and Group

Important

If you arrived at this page trying to enable Amazon Advertising for your application or web site, see [Becoming a Product Advertising API Developer](#).

As a [best practice \(p. 60\)](#), do not use the AWS account root user for any task where it's not required. Instead, create a new IAM user for each person that requires administrator access. Then make those users administrators by placing the users into an "Administrators" group to which you attach the AdministratorAccess managed policy.

Thereafter, the users in the administrators group should set up the groups, users, and so on, for the AWS account. All future interaction should be through the AWS account's users and their own keys instead of the root user. However, to perform some account and service management tasks, you must log in using the root user credentials. To view the tasks that require you to sign in as the root user, see [AWS Tasks that Require Account Root User](#).

Creating an Administrator IAM User and Group (Console)

This procedure describes how to use the AWS Management Console to create an IAM user for yourself and add that user to a group that has administrative permissions from an attached managed policy.

To create an administrator user for yourself and add the user to an administrators group (console)

1. Use your AWS account email address and password to sign in as the [AWS account root user](#) to the IAM console at <https://console.aws.amazon.com/iam/>.

Note

We strongly recommend that you adhere to the best practice of using the **Administrator** IAM user below and securely lock away the root user credentials. Sign in as the root user only to perform a few [account and service management tasks](#).

2. Enable access to billing data for the IAM admin user that you will create.
 - a. On the navigation bar, choose your account name, and then choose **My Account**.
 - b. Next to **IAM User and Role Access to Billing Information**, choose **Edit**.
 - c. Select the check box to **Activate IAM Access** and choose **Update**.
 - d. On the navigation bar, choose **Services** and then **IAM** to return to the IAM dashboard.
3. In the navigation pane, choose **Users** and then choose **Add user**.
4. For **User name**, type **Administrator**.
5. Select the check box next to **AWS Management Console access**, select **Custom password**, and then type your new password in the text box. By default, AWS forces the new user to create a new

password when first signing in. You can optionally clear the check box next to **User must create a new password at next sign-in** to allow the new user to reset their password after they sign in.

6. Choose **Next: Permissions**.
7. On the **Set permissions** page, choose **Add user to group**.
8. Choose **Create group**.
9. In the **Create group** dialog box, for **Group name** type **Administrators**.
10. Choose **Filter policies**, and then choose **AWS managed - job function** to filter the table contents.
11. In the policy list, select the check box for **AdministratorAccess**. Then choose **Create group**.
12. Back in the list of groups, select the check box for your new group. Choose **Refresh** if necessary to see the group in the list.
13. Choose **Next: Tags**.
14. (Optional) Add metadata to the user by attaching tags as key-value pairs. For more information about using tags in IAM, see [Tagging IAM Users and Roles \(p. 295\)](#).
15. Choose **Next: Review** to see the list of group memberships to be added to the new user. When you are ready to proceed, choose **Create user**.

You can use this same process to create more groups and users and to give your users access to your AWS account resources. To learn about using policies that restrict user permissions to specific AWS resources, see [Access Management \(p. 355\)](#) and [Example IAM Identity-Based Policies \(p. 394\)](#). To add additional users to the group after it's created, see [Adding and Removing Users in an IAM Group \(p. 171\)](#).

Creating an IAM User and Group (AWS CLI)

If you followed the steps in the previous section, you used the AWS Management Console to set up an administrators group while creating the IAM user in your AWS account. This procedure shows an alternative way to create a group.

Overview: Setting Up an Administrators Group

1. Create a group and give it a name (for example, Admins). For more information, see [Creating a Group \(AWS CLI\) \(p. 21\)](#).
2. Attach a policy that gives the group administrative permissions—access to all AWS actions and resources. For more information, see [Attaching a Policy to the Group \(AWS CLI\) \(p. 22\)](#).
3. Add at least one user to the group. For more information, see [Creating an IAM User in Your AWS Account \(p. 86\)](#).

Creating a Group (AWS CLI)

This section shows how to create a group in the IAM system.

To create an administrators group (AWS CLI)

1. Type the `aws iam create-group` command with the name you've chosen for the group. Optionally, you can include a path as part of the group name. For more information about paths, see [Friendly Names and Paths \(p. 575\)](#). The name can consist of letters, digits, and the following characters: plus (+), equal (=), comma (,), period (.), at (@), underscore (_), and hyphen (-). The name is not case sensitive and can be a maximum of 128 characters in length.

In this example, you create a group named Admins.

```
aws iam create-group --group-name Admins
{
```

```
"Group": {  
    "Path": "/",  
    "CreateDate": "2014-06-05T20:29:53.622Z",  
    "GroupId": "ABCDEFGHABCDEFGHABCDE",  
    "Arn": "arn:aws:iam::123456789012:group/Admins",  
    "GroupName": "Admins"  
}  
}
```

2. Type the `aws iam list-groups` command to list the groups in your AWS account and confirm the group was created.

```
aws iam list-groups  
{  
    "Groups": [  
        {  
            "Path": "/",  
            "CreateDate": "2014-06-05T20:29:53.622Z",  
            "GroupId": "ABCDEFGHABCDEFGHABCDE",  
            "Arn": "arn:aws:iam::123456789012:group/Admins",  
            "GroupName": "Admins"  
        }  
    ]  
}
```

The response includes the Amazon Resource Name (ARN) for your new group. The ARN is a standard format that AWS uses to identify resources. The 12-digit number in the ARN is your AWS account ID. The friendly name you assigned to the group (Admins) appears at the end of the group's ARN.

Attaching a Policy to the Group (AWS CLI)

This section shows how to attach a policy that lets any user in the group perform any action on any resource in the AWS account. You do this by attaching the [AWS managed policy \(p. 364\)](#) called AdministratorAccess to the Admins group. For more information about policies, see [Access Management \(p. 355\)](#).

To add a policy giving full administrator permissions (AWS CLI)

1. Type the `aws iam attach-group-policy` command to attach the policy called AdministratorAccess to your Admins group. The command uses the ARN of the AWS managed policy called AdministratorAccess.

```
aws iam attach-group-policy --group-name Admins --policy-arn arn:aws:iam::aws:policy/AdministratorAccess
```

If the command is successful, there is no response.

2. Type the `aws iam list-attached-group-policies` command to confirm the policy is attached to the Admins group.

```
aws iam list-attached-group-policies --group-name Admins
```

The response lists the names of the policies attached to the Admins group. A response like the following tells you that the policy named AdministratorAccess has been attached to the Admins group:

```
{  
    "AttachedPolicies": [
```

```
{  
    "PolicyName": "AdministratorAccess",  
    "PolicyArn": "arn:aws:iam::aws:policy/AdministratorAccess"  
}  
],  
"IsTruncated": false  
}
```

You can confirm the contents of a particular policy with the `aws iam get-policy` command.

Important

After you have the administrators group set up, you must add at least one user to it. For more information about adding users to a group, see [Creating an IAM User in Your AWS Account \(p. 86\)](#).

Related Resources

For related information found in the *Amazon Web Services General Reference*, see the following resources:

- [AWS Tasks that Require Account Root User](#)

For related information in the *IAM User Guide*, see the following resources:

- [Refining Permissions Using Service Last Accessed Data \(p. 475\)](#)
- [Tutorial: Delegate Access to the Billing Console \(p. 27\)](#)

Creating Your First IAM Delegated User and Group

To support multiple users in your AWS account, you must delegate permission to allow other people to perform only the actions you want to allow. To do this, create an IAM group with the permissions those people need and then add IAM users to the necessary groups as you create them. You can use this process to set up the groups, users, and permissions for your entire AWS account.

This solution is best used by small and medium organizations where an AWS administrator can manually manage the users and groups. For large organizations, you can use [custom IAM roles \(p. 213\)](#), [federation \(p. 184\)](#), or [single sign-on](#).

Creating a Delegated IAM User and Group (Console)

You can use the AWS Management Console to create an IAM group with delegated permissions, and then create an IAM user for someone else and add it to the group.

To create a delegated group and user for someone else (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane on the left, choose **Policies**.

If this is your first time choosing **Policies**, the **Welcome to Managed Policies** page appears. Choose **Get Started**.

3. Choose **Create policy**.
4. Choose the **JSON** tab and on the right side of the window choose **Import managed policy**.
5. In the **Import managed policies** window, type **power** to reduce the list of policies. Then select the button next to the **PowerUserAccess** AWS managed policy.

6. Choose **Import**.
The imported policy is added to your JSON policy.
7. Choose **Review policy**.
8. On the **Review** page, for **Name**, type **PowerUserExampleCorp**. For **Description**, type **Allows full access to all services except those for user management**. Then choose **Create policy** to save your work.
9. In the navigation pane, choose **Groups** and then choose **Create New Group**.
10. In the **Group Name** box, type **PowerUsers**.
11. In the list of policies, select the check box next to **PowerUserExampleCorp**. Then choose **Next Step**.
12. Choose **Create Group**.
13. In the navigation pane, choose **Users** and then choose **Add user**.
14. For **User name**, type **mary.major@examplecorp.com**.
15. Choose **Add another user** and type **diego.ramirez@examplecorp.com** for the second user.
16. Select the check box next to **AWS Management Console access** and select **Autogenerated password**. By default, AWS forces the new user to create a new password when first signing in. Clear the check box next to **User must create a new password at next sign-in** to allow the new user to reset their password after they sign in.
17. Choose **Next: Permissions**.
18. On the **Set permissions** page, choose **Add user to group** and select the check box next to **PowerUsers**.
19. Choose **Next: Tags**.
20. (Optional) Add metadata to the user by attaching tags as key-value pairs. For more information about using tags in IAM, see [Tagging IAM Users and Roles \(p. 295\)](#).
21. Choose **Next: Review** to see the list of group memberships to be added to the new user. When you are ready to proceed, choose **Create users**.
22. Download or copy the passwords for your new users and deliver them to the users securely. Separately, provide your users with a [link to your IAM user console page \(p. 73\)](#) and the user names you just created.

Reducing the Group Permissions

Members of the **PowerUser** group have full access to all services except a few that provide user management actions (like IAM and Organizations). After a predefined period of inactivity (such as 90 days) has passed, you can review the services that your group members have accessed. Then you can reduce the permissions of the **PowerUserExampleCorp** policy to include only the services that your team needs.

For more information about the service last accessed data, see [Refining Permissions Using Service Last Accessed Data \(p. 475\)](#).

Reviewing Service Last Accessed Data

Wait for a predefined period of inactivity (such as 90 days) to pass. Then you can review the service last accessed data for your users or groups to learn when your users last attempted to access the services that your **PowerUserExampleCorp** policy allows.

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Groups** and then choose the **PowerUser** group name.

3. On the group summary page, choose the **Access Advisor** tab.

The table of service last accessed data shows when the group members last attempted to access each service, in chronological order from the most recent attempt. The table includes only the services that the policy allows. In this case, the `PowerUserExampleCorp` policy allows access to all AWS services.

4. Review the table and make a list of the services that your group members have recently accessed.

For example, assume that within the last month, your team has accessed only the Amazon EC2 and Amazon S3 services. But six months ago, they accessed Amazon EC2 Auto Scaling and IAM. You know that they were investigating EC2 Auto Scaling, but decided that it wasn't necessary. You also know that they used IAM to create a role to allow Amazon EC2 to access data in an S3 bucket. So you decide to scale back the user's permissions to allow access to only the Amazon EC2 and Amazon S3 services.

Editing a Policy to Reduce Permissions

After you review your service last accessed data, you can edit your policy to allow access to only the services that your users need.

To use data to allow access to only necessary services

1. In the navigation pane, choose **Policies** and then choose the `PowerUserExampleCorp` policy name.
2. Choose **Edit policy**, and then choose the **JSON** tab.
3. Edit the JSON policy document to allow only the services you want.

For example, edit the first statement that includes the `Allow` effect and the `NotAction` element to allow only Amazon EC2 and Amazon S3 actions. To do this, replace it with the statement with the `FullAccessToSomeServices` ID. Your new policy will look like the following example policy.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "FullAccessToSomeServices",  
            "Effect": "Allow",  
            "Action": [  
                "ec2:*",  
                "s3:*"  
            ],  
            "Resource": "*"  
        },  
        {  
            "Effect": "Allow",  
            "Action": [  
                "iam:CreateServiceLinkedRole",  
                "iam>DeleteServiceLinkedRole",  
                "iam>ListRoles",  
                "organizations:DescribeOrganization"  
            ],  
            "Resource": "*"  
        }  
    ]  
}
```

4. To further reduce your policies' permissions to specific actions and resources, view your events in CloudTrail **Event history**. There you can view detailed information about the specific actions and resources that your user has accessed. For more information, see [Viewing CloudTrail Events in the CloudTrail Console](#) in the *AWS CloudTrail User Guide*.

How Users Sign In to Your Account

After you create IAM users (with passwords), those users can sign in to the AWS Management Console. To sign in, they need your account ID or alias. They can also sign in from a custom URL that includes your account ID.

Note

If your company has an existing identity system, you might want to create a single sign-on (SSO) option. SSO gives users access to the AWS Management Console without requiring them to have an IAM user identity. SSO also eliminates the need for users to sign in to your organization's site and to AWS separately. For more information, see [Enabling Custom Identity Broker Access to the AWS Console \(p. 213\)](#).

Before you create a sign-in URL for your account, you create an account alias so that the URL includes your account name instead of an account ID. For more information, see [Your AWS Account ID and Its Alias \(p. 76\)](#).

You can find the sign-in URL for an account on the IAM console dashboard.

IAM users sign-in link:

<https://my-account.signin.aws.amazon.com/console>

[Customize](#) | [Copy Link](#)

To create a sign-in URL for your IAM users, use the following pattern:

`https://account-ID-or-alias.signin.aws.amazon.com/console`

IAM users can also sign in at the following endpoint and enter the account ID or alias manually, instead of using your custom URL:

`https://signin.aws.amazon.com/console`

Permissions Required for Console Activities

IAM users in your account have access only to the AWS resources that you specify in a policy. That policy must be attached to the user or to an IAM group that the user belongs to. To work in the console, users must have permissions to perform the actions that the console performs, such as listing and creating AWS resources. For more information, see [Access Management \(p. 355\)](#) and [Example IAM Identity-Based Policies \(p. 394\)](#).

If users in your account need programmatic access, you can create an access key pair (an access key ID and a secret access key) for each user. For more information, see [Managing Access Keys \(Console\) \(p. 112\)](#).

Logging Sign-In Details in CloudTrail

If you enable CloudTrail to log sign-in events, you must understand how CloudTrail logs the events. CloudTrail includes global and Regions log entries. Where a sign-in event is logged in CloudTrail depends on how your users sign in. For details, see [Logging IAM Events with CloudTrail](#).

IAM Tutorials

This section contains walkthroughs that present complete end-to-end procedures for common tasks that you can perform in IAM. They are intended for a lab-type environment, with sample company names, user names, and so on. Their purpose is to provide general guidance. They are not intended for direct use in your production environment without careful review and adaptation to the unique aspects of your organization's environment.

Topics

- [Tutorial: Delegate Access to the Billing Console \(p. 27\)](#)
- [Tutorial: Delegate Access Across AWS Accounts Using IAM Roles \(p. 30\)](#)
- [Tutorial: Create and Attach Your First Customer Managed Policy \(p. 39\)](#)
- [Tutorial: Using Tags for Attribute-Based Access Control in AWS \(p. 41\)](#)
- [Tutorial: Enable Your Users to Configure Their Own Credentials and MFA Settings \(p. 56\)](#)

Tutorial: Delegate Access to the Billing Console

AWS account owners can delegate access to specific IAM users who need to view or manage the AWS Billing and Cost Management data for an AWS account. The instructions that follow will help you set up a pretested scenario. This scenario will help you gain hands-on experience configuring billing permissions without concern for affecting your main AWS production account.

This workflow has four basic steps.

Step 1: Activate Access to Billing Data on Your AWS Test Account (p. 28)

If you create a single AWS account, only the AWS account owner ([AWS account root user \(p. 338\)](#)) has access to view and manage billing information. IAM users cannot access billing data until the account owner activates IAM access and attaches policies that provide billing actions to the user or role. To view additional tasks that require you to sign in as the root user, see [AWS Tasks that Require Account Root User](#).

If you [create a member account](#) using AWS Organizations, this feature is enabled by default.

Step 2: Create IAM Policies That Grant Permissions to Billing Data (p. 28)

After enabling billing access on your account, you must still explicitly grant access to billing data to specific IAM users or groups. You grant this access with a customer managed policy.

Step 3: Attach Billing Policies to Your Groups (p. 29)

When you attach a policy to a group, all members of that group receive the complete set of access permissions that are associated with that policy. In this scenario, you attach the new billing policies to groups containing only those users who require the billing access.

Step 4: Test Access to the Billing Console (p. 29)

Once you've completed the core tasks, you're ready to test the policy. Testing ensures that the policy works the way you want it to.

Prerequisites

Create a test AWS account to use with this tutorial. In this account create two test users and two test groups as summarized in the following table. Be sure to assign a password to each user so that you can sign in later in Step 4.

Create user accounts	Create and configure group accounts	
FinanceManager	BillingFullAccessGroup	FinanceManager
FinanceUser	BillingViewAccessGroup	FinanceUser

Step 1: Activate Access to Billing Data on Your AWS Test Account

To learn how to activate billing access for your test users, see [Activating Access to the Billing and Cost Management Console](#) in the *AWS Billing and Cost Management User Guide*.

Note

If you [create a member account](#) using AWS Organizations, this feature is enabled by default.

Step 2: Create IAM Policies That Grant Permissions to Billing Data

Next, create custom policies that grant both view and full access permissions to the pages within the Billing and Cost Management console. For general information about IAM permissions policies, see [Managed Policies and Inline Policies \(p. 364\)](#).

To create IAM policies that grant permissions to billing data

1. Sign in to the AWS Management Console as a user with administrator credentials. To adhere to IAM best practices, don't sign in with your root user credentials. For more information, see [Create individual IAM users \(p. 61\)](#).
2. Open the IAM console at <https://console.aws.amazon.com/iam/>.
3. In the navigation pane, choose **Policies**, and then choose **Create policy**.
4. On the **Visual editor** tab, choose **Choose a service** to get started. Then choose **Billing**.
5. Follow these steps to create two policies:

Full access

- a. Choose **Select actions** and then select the check box next to **All Actions (*)**. You do not need to select a resource or condition for this policy.
- b. Choose **Review policy**.
- c. On the **Review** page, next to **Name**, type **BillingFullAccess**, and then choose **Create policy** to save it.

Read-only access

- a. Repeat steps [3 and 4 \(p. 28\)](#).
- b. Choose **Select actions** and then select the check box next to **Read**. You do not need to select a resource or condition for this policy.
- c. Choose **Review policy**.
- d. On the **Review** page, for **Name**, type **BillingViewAccess**. Then choose **Create policy** to save it.

To review descriptions for each of the permissions available in IAM policies that grant users access to the Billing and Cost Management console, see [Billing Permissions Descriptions](#).

Step 3: Attach Billing Policies to Your Groups

Now that you have custom billing policies available, you can attach them to their corresponding groups that you created earlier. Although you can attach a policy directly to a user or role, we recommend (in accordance with IAM best practices) that you use groups instead. For more information, see [Use groups to assign permissions to IAM users \(p. 61\)](#).

To attach billing policies to your groups

1. In the navigation pane, choose **Policies** to display the full list of policies available to your AWS account. To attach each policy to its appropriate group, follow these steps:

Full access

- a. In the policy search box, type **BillingFullAccess**, and then select the check box next to the policy name.
- b. Choose **Policy actions**, and then choose **Attach**.
- c. In the identity (user, group, and role) search box, type **BillingFullAccessGroup**, select the check box next to the name of the group, and then choose **Attach policy**.

Read-only access

- a. In the policy search box, type **BillingViewAccess**, and then select the check box next to the policy name.
 - b. Choose **Policy actions**, and then choose **Attach**.
 - c. In the identity (user, group, and role) search box, type **BillingViewAccessGroup**, select the check box next to the name of the group, and then choose **Attach policy**.
2. Sign out of the console, and then proceed to [Step 4: Test Access to the Billing Console \(p. 29\)](#).

Step 4: Test Access to the Billing Console

We recommend that you test access by signing in as each of the test users to learn what your users might experience. Use the following steps to sign in using both test accounts to see the difference between access rights.

To test billing access by signing in with both test user accounts

1. Use your AWS account ID or account alias, your IAM user name, and your password to sign in to the [IAM console](#).

Note

For your convenience, the AWS sign-in page uses a browser cookie to remember your IAM user name and account information. If you previously signed in as a different user, choose **Sign in to a different account** near the bottom of the page to return to the main sign-in page. From there, you can type your AWS account ID or account alias to be redirected to the IAM user sign-in page for your account.

2. Sign in with each account using the steps provided below so you can compare the different user experiences.

Full access

- a. Sign in to your AWS account as the user FinanceManager.
- b. On the navigation bar, choose **FinanceManager@<account alias or ID number>**, and then choose **My Billing Dashboard**.

- c. Browse through the pages and choose the various buttons to ensure that you have full modify permissions.

Read-only access

- a. Sign in to your AWS account as the user FinanceUser.
- b. On the navigation bar, choose **FinanceUser@<account alias or ID number>**, and then choose **My Billing Dashboard**.
- c. Browse through the pages. Notice that you can display costs, reports, and billing data with no problems. However, if you choose an option to modify a value, you receive an **Access Denied** message. For example, on the **Preferences** page, choose any of the check boxes on the page, and then choose **Save preferences**. The console message informs you that you need **ModifyBilling** permissions to make changes to that page.

Related Resources

For related information found in the *AWS Billing and Cost Management User Guide*, see the following resources:

- [Activating Access to the Billing and Cost Management Console](#)
- [Example 4: Allow full access to AWS services but deny IAM users access to the Billing and Cost Management console.](#)
- [Billing Permissions Descriptions](#)

For related information in the *IAM User Guide*, see the following resources:

- [Managed Policies and Inline Policies \(p. 364\)](#)
- [Controlling User Access to the AWS Management Console \(p. 75\)](#)
- [Attaching a Policy to an IAM Group \(p. 172\)](#)

Summary

You've now successfully completed all of the steps necessary to delegate user access to the Billing and Cost Management console. As a result, you've seen firsthand what your users billing console experience will be like. You can now proceed to implement this logic in your production environment at your convenience.

Tutorial: Delegate Access Across AWS Accounts Using IAM Roles

This tutorial teaches you how to use a role to delegate access to resources that are in different AWS accounts that you own (Production and Development). You share resources in one account with users in a different account. By setting up cross-account access in this way, you don't need to create individual IAM users in each account. In addition, users don't have to sign out of one account and sign into another in order to access resources in different AWS accounts. After configuring the role, you see how to use the role from the AWS Management Console, the AWS CLI, and the API.

In this tutorial, imagine that the Production account is where live applications are managed. The Development account is a sandbox where developers and testers can freely test applications. In

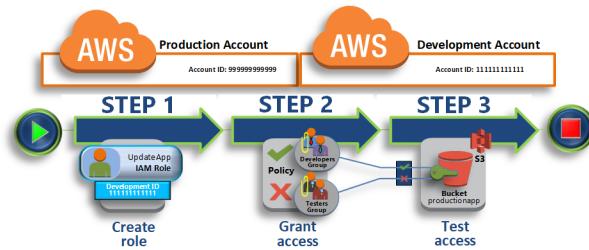
each account, application information is stored in Amazon S3 buckets. You manage IAM users in the Development account, where you have two IAM groups: Developers and Testers. Users in both groups have permissions to work in the Development account and access resources there. From time to time, a developer must update the live applications in the Production account. These applications are stored in an Amazon S3 bucket called `productionapp`.

At the end of this tutorial, you have the following:

- Users in the Development account (the trusted account) that are allowed to assume a specific role in the Production account.
- A role in the Production account (the trusting account) that is allowed to access a specific Amazon S3 bucket.
- The `productionapp` bucket in the Production account.

Developers can use the role in the AWS Management Console to access the `productionapp` bucket in the Production account. They can also access the bucket by using API calls that are authenticated by temporary credentials provided by the role. Similar attempts by a Tester to use the role fail.

This workflow has three basic steps.



Step 1: Create a Role (p. 32)

First, you use the AWS Management Console to establish trust between the Production account (ID number 999999999999) and the Development account (ID number 111111111111). You start by creating an IAM role named `UpdateApp`. When you create the role, you define the Development account as a trusted entity and specify a permissions policy that allows trusted users to update the `productionapp` bucket.

Step 2: Grant Access to the Role (p. 34)

In this step of the tutorial, you modify the IAM group policy so that Testers are denied access to the `UpdateApp` role. Because Testers have PowerUser access in this scenario, we must explicitly deny the ability to use the role.

Step 3: Test Access by Switching Roles (p. 35)

Finally, as a Developer, you use the `UpdateApp` role to update the `productionapp` bucket in the Production account. You see how to access the role through the AWS console, the AWS CLI, and the API.

Prerequisites

This tutorial assumes that you have the following already in place:

- Two separate AWS accounts that you can use, one to represent the Development account, and one to represent the Production account.
- Users and groups in the Development account created and configured as follows:

User	Group	Permissions
David	Developers	Both users are able to sign in and use the AWS Management Console in the Development account.
Theresa	Testers	

- You do not need to have any users or groups created in the Production account.
- An Amazon S3 bucket created in the Production account. We call it `ProductionApp` in this tutorial, but because S3 bucket names must be globally unique, you must use a bucket with a different name.

Step 1: Create a Role

You can allow users from one AWS account to access resources in another AWS account. To do this, create a role that defines who can access it and what permissions it grants to users that switch to it.

In this step of the tutorial, you create the role in the Production account and specify the Development account as a trusted entity. You also limit the role's permissions to only read and write access to the `productionapp` bucket. Anyone who is granted permission to use the role can read and write to the `productionapp` bucket.

Before you can create a role, you need the account ID of the Development AWS account. The account ID is a unique identifier assigned to each AWS account.

To obtain the Development AWS account ID

1. Sign in to the AWS Management Console as an administrator of the Development account, and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation bar, choose **Support**, and then **Support Center**. Your currently signed-in 12-digit account number (ID) appears in the **Support Center** title bar. For this scenario, we pretend the Development account ID is `111111111111`. However, you should use a valid account ID if you are reconstructing the scenario in your test environment.

To create a role in the Production account that can be used by the Development account

1. Sign in to the AWS Management Console as an administrator of the Production account, and open the IAM console.
2. Before creating the role, prepare the managed policy that defines the permissions that the role requires. You attach this policy to the role in a later step.

You want to set read and write access to the `productionapp` bucket. Although AWS provides some Amazon S3 managed policies, there isn't one that provides read and write access to a single Amazon S3 bucket. You can create your own policy instead.

In the navigation pane on the left, choose **Policies** and then choose **Create policy**.

3. Choose the **JSON** tab and copy the text from the following JSON policy document. Paste this text into the **JSON** text box, replacing the resource ARN (`arn:aws:s3:::productionapp`) with the real one appropriate to your S3 bucket.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3>ListAllMyBuckets",
      "Resource": "arn:aws:s3:::productionapp"
    }
  ]
}
```

```
        "Resource": "*"
    },
{
    "Effect": "Allow",
    "Action": [
        "s3>ListBucket",
        "s3:GetBucketLocation"
    ],
    "Resource": "arn:aws:s3:::productionapp"
},
{
    "Effect": "Allow",
    "Action": [
        "s3.GetObject",
        "s3:PutObject",
        "s3>DeleteObject"
    ],
    "Resource": "arn:aws:s3:::productionapp/*"
}
]
```

The `ListBucket` permission allows users to view objects in the `productionapp` bucket. The `GetObject`, `PutObject`, `DeleteObject` permissions allows users to view, update, and delete contents in the `productionapp` bucket.

4. When you are finished, choose **Review policy**. The [Policy Validator \(p. 448\)](#) reports any syntax errors.

Note

You can switch between the **Visual editor** and **JSON** tabs anytime. However, if you make changes or choose **Review policy** in the **Visual editor** tab, IAM might restructure your policy to optimize it for the visual editor. For more information, see [Policy Restructuring \(p. 548\)](#).

5. On the **Review** page, type `read-write-app-bucket` for the policy name. Review the policy **Summary** to see the permissions granted by your policy, and then choose **Create policy** to save your work.

The new policy appears in the list of managed policies.

6. In the navigation pane on the left, choose **Roles** and then choose **Create role**.
7. Choose the **Another AWS account** role type.
8. For **Account ID**, type the Development account ID.

This tutorial uses the example account ID `111111111111` for the Development account. You should use a valid account ID. If you use an invalid account ID, such as `111111111111`, IAM does not let you create the new role.

For now you do not need to require an external ID, or require users to have multi-factor authentication (MFA) in order to assume the role. So leave these options unselected. For more information, see [Using Multi-Factor Authentication \(MFA\) in AWS \(p. 119\)](#)

9. Choose **Next: Permissions** to set the permissions that will be associated with the role.
10. Select the box next to the policy that you created previously.

Tip

For **Filter**, choose **Customer managed** to filter the list to include only the policies that you have created. This hides the AWS created policies and makes it much easier to find the one you're looking for.

Then choose **Next: Tags**.

11. (Optional) Add metadata to the user by attaching tags as key-value pairs. For more information about using tags in IAM, see [Tagging IAM Users and Roles \(p. 295\)](#).

12. Choose **Next: Review** and type **UpdateApp** for the role name.
13. (Optional) For **Role description**, type a description for the new role.
14. After reviewing the role, choose **Create role**.

The **UpdateApp** role appears in the list of roles.

Now you must obtain the role's Amazon Resource Name (ARN), which is a unique identifier for the role. When you modify the Developers and Testers group's policy, you will specify the role's ARN to grant or deny permissions.

To obtain the ARN for UpdateApp

1. In the navigation pane of the IAM console, choose **Roles**.
2. In the list of roles, choose the **UpdateApp** role.
3. In the **Summary** section of the details pane, copy the **Role ARN** value.

The Production account has an account ID of 999999999999, so the role ARN is `arn:aws:iam::999999999999:role/UpdateApp`. Ensure that you supply the real AWS account ID for your 'production' account.

At this point, you have established trust between the Production and Development accounts. You did this by creating a role in the Production account that identifies the Development account as a trusted principal. You also defined what users who switch to the **UpdateApp** role can do.

Next, modify the permissions for the groups.

Step 2: Grant Access to the Role

At this point, both Testers and Developers group members have permissions that allow them to freely test applications in the Development account. Here are the steps required to add permissions to allow switching to the role.

To modify the Developers group to allow them to switch to the UpdateApp role

1. Sign in as an administrator in the Development account, and open the IAM console.
2. Choose **Groups**, and then choose **Developers**.
3. Choose the **Permissions** tab, expand the **Inline Policies** section, and then choose **Create Group Policy**. If no inline policy exists yet, then the button does not appear. Instead, choose the link at the end of "To create one, click here."
4. Choose **Custom Policy** and then choose **Select** button.
5. Type a policy name like **allow-assume-S3-role-in-production**.
6. Add the following policy statement to allow the `AssumeRole` action on the `UpdateApp` role in the Production account. Be sure that you change `PRODUCTION-ACCOUNT-ID` in the `Resource` element to the actual AWS account ID of the Production account.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": "sts:AssumeRole",  
            "Resource": "arn:aws:iam::PRODUCTION-ACCOUNT-ID:role/UpdateApp"  
        }  
    ]  
}
```

The **Allow** effect explicitly allows the Developers group access to the `UpdateApp` role in the Production account. Any developer who tries to access the role will succeed.

7. Choose **Apply Policy** to add the policy to the Developer group.

In most environments, the following procedure is likely not needed. If, however, you use Power User permissions, then some groups might already be able to switch roles. The following procedure shows how to add a "Deny" permission to the Testers group to ensure that they cannot assume the role. If this procedure is not needed in your environment, then we recommend that you do not add it. "Deny" permissions make the overall permissions picture more complicated to manage and understand. Use "Deny" permissions only when there is not a better option.

To modify the Testers group to deny permission to assume the `UpdateApp` role

1. Choose **Groups**, and then choose **Testers**.
2. Choose the **Permissions** tab, expand the **Inline Policies** section, and then choose **Create Group Policy**.
3. Choose **Custom Policy** and then choose the **Select** button.
4. Type a policy name like `deny-assume-S3-role-in-production`.
5. Add the following policy statement to deny the `AssumeRole` action on the `UpdateApp` role. Be sure that you change `PRODUCTION-ACCOUNT-ID` in the Resource element to the actual AWS account ID of the Production account.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Deny",  
            "Action": "sts:AssumeRole",  
            "Resource": "arn:aws:iam::PRODUCTION-ACCOUNT-ID:role/UpdateApp"  
        }  
    ]  
}
```

The **Deny** effect explicitly denies the Testers group access to the `UpdateApp` role in the Production account. Any tester who tries to access the role will get an access denied message.

6. Choose **Apply Policy** to add the policy to the Tester group.

The Developers group now has permissions to use the `UpdateApp` role in the Production account. The Testers group is prevented from using the `UpdateApp` role.

Next, you'll learn how David, a developer, can access the `productionapp` bucket in the Production account. David can access the bucket from the AWS Management Console, the AWS CLI, or the AWS API.

Step 3: Test Access by Switching Roles

After completing the first two steps of this tutorial, you have a role that grants access to a resource in the Production account. You also have one group in the Development account whose users are allowed to use that role. The role is now ready to use. This step discusses how to test switching to that role from the AWS Management Console, the AWS CLI, and the AWS API.

Important

You can switch to a role only when you are signed in as an IAM user or a federated user. Additionally, if you launch an Amazon EC2 instance to run an application, the application can assume a role through its instance profile. You cannot switch to a role when you are signed in as the AWS account root user.

Switch Roles (Console)

If David needs to work with in the Production environment in the AWS Management Console, he can do so by using **Switch Role**. He specifies the account ID or alias and the role name, and his permissions immediately switch to those permitted by the role. He can then use the console to work with the `productionapp` bucket, but cannot work with any other resources in Production. While David is using the role, he also cannot make use of his power-user privileges in the Development account. That's because only one set of permissions can be in effect at a time.

Important

Switching roles using the AWS Management Console works only with accounts that do not require an `ExternalId`. For example, assume that you grant access to your account to a third party and require an `ExternalId` in a `Condition` element in your permissions policy. In that case, the third party can access your account only by using the AWS API or a command line tool. The third party cannot use the console because it cannot supply a value for `ExternalId`. For more information about this scenario, see [How to Use an External ID When Granting Access to Your AWS Resources to a Third Party \(p. 233\)](#), and [How to Enable Cross-Account Access to the AWS Management Console](#) in the [AWS Security Blog](#).

There are two ways that David can use to enter the **Switch Role** page:

- David receives a link from his administrator that points to a pre-defined Switch Role configuration. The link is provided to the administrator on the final page of the **Create role** wizard or on the **Role Summary** page for a cross-account role. Choosing this link takes David to the **Switch Role** page with the **Account ID** and **Role name** fields already filled in. All David needs to do is choose **Switch Role** and he's done.
- The administrator does not send the link in email, but instead sends the **Account ID** number and **Role Name** values. David must manually type them to switch roles. This is illustrated in the following procedure.

To assume a role

1. David signs into the AWS Management Console using his normal user that is in the Development group.
2. He chooses the link that his administrator sent to him in email. This takes him to the **Switch Role** page with the account ID or alias and the role name information already filled in.

—or—

He chooses his name (the Identity menu) on the navigation bar, and then chooses **Switch Role**.

If this is the first time that David tries to access the Switch Role page this way, he will first land on a first-run **Switch Role** page. This page provides additional information on how switching roles can enable users to manage resources across AWS accounts. David must choose the **Switch Role** button on this page to complete the rest of this procedure.

3. Next, in order to access the role, David must manually type the Production account ID number (`999999999999`) and the role name (`UpdateApp`).

Also, David wants to monitor which roles (and associated permissions) are currently active. To keep track of this information, he types `PRODUCTION` in the **Display Name** text box, selects the red color option, and then chooses **Switch Role**.

4. David can now use the Amazon S3 console to work with the Amazon S3 bucket, or any other resource to which the `UpdateApp` role has permissions.
5. When he is done with the work he needs to do, David can return to his original permissions. To do that, he chooses the **PRODUCTION** role display name on the navigation bar and then chooses **Back to David @ 111111111111**.

6. The next time that David wants to switch roles and chooses the Identity menu in the navigation bar, he sees the PRODUCTION entry still there from last time. He can simply choose that entry to switch roles immediately without having to reenter the account ID and role name.

Switch Roles (AWS CLI)

If David needs to work in the Production environment at the command line, he can do so by using the [AWS CLI](#). He runs the `aws sts assume-role` command and passes the role ARN to get temporary security credentials for that role. He then configures those credentials in environment variables so subsequent AWS CLI commands work using the role's permissions. While David is using the role, he cannot use his power-user privileges in the Development account. The reason is that only one set of permissions can be in effect at a time.

Note that all access keys and tokens are examples only and cannot be used as shown. Replace with the appropriate values from your live environment.

To assume a role

1. David opens a command prompt window, and confirms that the AWS CLI client is working by running the command:

```
aws help
```

Note

David's default environment uses the David user credentials from his default profile that he created with the `aws configure` command. For more information, see [Configuring the AWS Command Line Interface](#) in the [AWS Command Line Interface User Guide](#).

2. He begins the switch role process by running the following command to switch to the `UpdateApp` role in the Production account. He got the role ARN from the administrator that created the role. The command requires that you provide a session name as well, you can choose any text you like for that.

```
aws sts assume-role --role-arn "arn:aws:iam::999999999999:role/UpdateApp" --role-session-name "David-ProdUpdate"
```

David then sees the following in the output:

```
{  
    "Credentials": {  
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",  
        "SessionToken": "AQoDYXdzEGcaEXAMPLE2gsYUL0  
+Im5ZEXAMLeEYjs1M2FUlgIJx9tQqNMBEXAMPLE  
CvSRyhOFW7jEXAMPLEw+vE/7s1HRpXviG7b+qYf4nD00EXAMPLEmj4wxS04L/  
uZEXAMPLEcihzFB5lTYLto9dyBgSDY  
EXAMPLE9/  
g7QRUhZp4bqbEXAMPLENwGPyoj59pFA41NKC1kVgkREXAMPLEj1zxQ7y52gekeVEXAMPLEDiB9ST3Uuysg  
sKdEXAMPLE1TVastU1A0SKFEXAMPLEiywCC/Cs8EXAMPLEpZgOs+6hz4AP4KEXAMPLERbASP  
+4eZScEXAMPLEsnf87e  
NhyDHq61kBQ==",  
        "Expiration": "2014-12-11T23:08:07Z",  
        "AccessKeyId": "AKIAIOSFODNN7EXAMPLE"  
    }  
}
```

3. David sees the three pieces that he needs in the Credentials section of the output.

- `AccessKeyId`

- `SecretAccessKey`
- `SessionToken`

David needs to configure the AWS CLI environment to use these parameters in subsequent calls. For information about the various ways to configure your credentials, see [Configuring the AWS Command Line Interface](#). You cannot use the `aws configure` command because it does not support capturing the session token. However, you can manually type the information into a configuration file. Because these are temporary credentials with a relatively short expiration time, it is easiest to add them to the environment of your current command line session.

4. To add the three values to the environment, David cuts and pastes the output of the previous step into the following commands. You might want to cut and paste into a simple text editor to address line wrap issues in the output of the session token. It must be added as a single long string, even though it is shown line wrapped here for clarity.

Note

The following example shows commands given in the Windows environment, where "set" is the command to create an environment variable. On a Linux or macOS computer, you would use the command "export" instead. All other parts of the example are valid in all three environments.

```
set AWS_ACCESS_KEY_ID=AKIAIOSFODNN7EXAMPLE
set AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRficyEXAMPLEKEY
set AWS_SESSION_TOKEN=AQoDYXdzEGcaEXAMPLE2gsYULO
+Im5ZEXAMLeeYjs1M2FUigIJx9tQqNMBEXAMPLEcvS
Ryh0FW7jEXAMLeW+vE/7s1HRpXviG7b+qYf4nD00EXAMPLEmj4wxS04L/
uZEXAMPLICihzFB51TYLto9dyBgSDyEXA
MPLKEY9/
g7QRUhZp4bqbEXAMPLENwGPyOj59pFA41NKC1kVgkREXAMPLEjlzxQ7y52gekeVEXAMPLEDiB9ST3UusKd
EXAMPLE1T vastU1A0SKFEXAMPLEiywCC/Cs8EXAMPLEpZgOs+6hz4AP4KEXAMPLErbASP
+4eZScEXAMPLENhhykxiHen
DHq6ikBQ==
```

At this point, any following commands run under the permissions of the role identified by those credentials. In David's case, the `UpdateApp` role.

5. Run the command to access the resources in the Production account. In this example, David simply lists the contents of his S3 bucket with the following command.

```
aws s3 ls s3://productionapp
```

Because Amazon S3 bucket names are universally unique, there is no need to specify the account ID that owns the bucket. To access resources for other AWS services, refer to the AWS CLI documentation for that service for the commands and syntax that are required to reference its resources.

Using AssumeRole (AWS API)

When David needs to make an update to the Production account from code, he makes an `AssumeRole` call to assume the `UpdateApp` role. The call returns temporary credentials that he can use to access the `productionapp` bucket in the Production account. With those credentials, David can make API calls to update the `productionapp` bucket. However, he cannot make API calls to access any other resources in the Production account, even though he has power-user permissions in the Development account.

To assume a role

1. David calls `AssumeRole` as part of an application. He must specify the `UpdateApp` ARN: `arn:aws:iam::999999999999:role/UpdateApp`.

The response from the `AssumeRole` call includes the temporary credentials with an `AccessKeyId` and a `SecretAccessKey`. It also includes an `Expiration` time that indicates when the credentials expire and you must request new ones.

2. With the temporary credentials, David makes an `s3:PutObject` call to update the `productionapp` bucket. He would pass the credentials to the API call as the `AuthParams` parameter. Because the temporary role credentials have only read and write access to the `productionapp` bucket, any other actions in the Production account are denied.

For a code example (using Python), see [Switching to an IAM Role \(AWS API\) \(p. 268\)](#).

Related Resources

- For more information about IAM users and groups, see [Identities \(Users, Groups, and Roles\) \(p. 82\)](#).
- For more information about Amazon S3 buckets, see [Create a Bucket](#) in the *Amazon Simple Storage Service Getting Started Guide*.
- To learn whether principals in accounts outside of your zone of trust (trusted organization, OU, or account) have access to assume your roles, see [What is IAM Access Analyzer?](#).

Summary

You have completed the cross-account API access tutorial. You created a role to establish trust with another account and defined what actions trusted entities can take. Then, you modified a group policy to control which IAM users can access the role. As a result, developers from the Development account can make updates to the `productionapp` bucket in the Production account by using temporary credentials.

Tutorial: Create and Attach Your First Customer Managed Policy

In this tutorial, you use the AWS Management Console to create a [customer managed policy \(p. 366\)](#) and then attach that policy to an IAM user in your AWS account. The policy you create allows an IAM test user to sign in directly to the AWS Management Console with read-only permissions.

This workflow has three basic steps:

Step 1: Create the Policy (p. 40)

By default, IAM users do not have permissions to do anything. They cannot access the AWS Management Console or manage the data within unless you allow it. In this step, you create a customer managed policy that allows any attached user to sign in to the console.

Step 2: Attach the Policy (p. 40)

When you attach a policy to a user, the user inherits all of the access permissions that are associated with that policy. In this step, you attach the new policy to a test user account.

Step 3: Test User Access (p. 41)

Once the policy is attached, you can sign in as the user and test the policy.

Prerequisites

To perform the steps in this tutorial, you need to already have the following:

- An AWS account that you can sign in to as an IAM user with administrative permissions.
- A test IAM user that has no permissions assigned or group memberships as follows:

User Name	Group	Permissions
PolicyUser	<none>	<none>

Step 1: Create the Policy

In this step, you create a customer managed policy that allows any attached user to sign in to the AWS Management Console with read-only access to IAM data.

To create the policy for your test user

1. Sign in to the IAM console at <https://console.aws.amazon.com/iam/> with your user that has administrator permissions.
2. In the navigation pane, choose **Policies**.
3. In the content pane, choose **Create policy**.
4. Choose the **JSON** tab and copy the text from the following JSON policy document. Paste this text into the **JSON** text box.

```
{  
    "Version": "2012-10-17",  
    "Statement": [ {  
        "Effect": "Allow",  
        "Action": [  
            "iam:GenerateCredentialReport",  
            "iam:Get*",  
            "iam>List*"  
        ],  
        "Resource": "*"  
    } ]  
}
```

5. When you are finished, choose **Review policy**. The [Policy Validator \(p. 448\)](#) reports any syntax errors.

Note

You can switch between the **Visual editor** and **JSON** tabs anytime. However, if you make changes or choose **Review policy** in the **Visual editor** tab, IAM might restructure your policy to optimize it for the visual editor. For more information, see [Policy Restructuring \(p. 548\)](#).

6. On the **Review** page, type **UsersReadOnlyAccessToIAMConsole** for the policy name. Review the policy **Summary** to see the permissions granted by your policy, and then choose **Create policy** to save your work.

The new policy appears in the list of managed policies and is ready to attach.

Step 2: Attach the Policy

Next you attach the policy you just created to your test IAM user.

To attach the policy to your test user

1. In the IAM console, in the navigation pane, choose **Policies**.
2. At the top of the policy list, in the search box, start typing **UsersReadOnlyAccessToIAMConsole** until you can see your policy. Then check the box next to **UsersReadOnlyAccessToIAMConsole** in the list.
3. Choose the **Policy actions** button, and then chose **Attach**.
4. For **Filter**, choose **Users**.
5. In the search box, start typing **PolicyUser** until that user is visible on the list. Then check the box next to that user in the list.
6. Choose **Attach Policy**.

You have attached the policy to your IAM test user, which means that user now has read-only access to the IAM console.

Step 3: Test User Access

For this tutorial, we recommend that you test access by signing in as the test user so you can see what your users might experience.

To test access by signing in with your test user account

1. Sign in to the IAM console at <https://console.aws.amazon.com/iam/> with your **PolicyUser** test user.
2. Browse through the pages of the console and try to create a new user or group. Notice that **PolicyUser** can display data but cannot create or modify existing IAM data.

Related Resources

For related information in the *IAM User Guide*, see the following resources:

- [Managed Policies and Inline Policies \(p. 364\)](#)
- [Controlling User Access to the AWS Management Console \(p. 75\)](#)
- [Create Individual IAM Users \(p. 61\)](#)

Summary

You've now successfully completed all of the steps necessary to create and attach a customer managed policy. As a result, you are able to sign in to the IAM console with your test account to see what the experience is like for your users.

Tutorial: Using Tags for Attribute-Based Access Control in AWS

Attribute-based access control (ABAC) is an authorization strategy that defines permissions based on attributes. In AWS, these attributes are called *tags*. You can attach tags to IAM principals (users or roles) and to AWS resources. You can then define policies that use tag condition keys to grant permissions to your principals based on their tags. When you use tags to control access to your AWS resources, you allow your teams and resources to grow with fewer changes to AWS policies. ABAC policies are

more flexible than traditional AWS policies, which require you to list each individual resource. For more information about ABAC and its advantage over traditional policies, see [What Is ABAC for AWS? \(p. 12\)](#).

Topics

- [Tutorial Overview \(p. 42\)](#)
- [Prerequisites \(p. 43\)](#)
- [Step 1: Create Test Users \(p. 43\)](#)
- [Step 2: Create the ABAC Policy \(p. 44\)](#)
- [Step 3: Create Roles \(p. 47\)](#)
- [Step 4: Test Creating Secrets \(p. 47\)](#)
- [Step 5: Test Viewing Secrets \(p. 49\)](#)
- [Step 6: Test Scalability \(p. 51\)](#)
- [Step 7: Test Updating and Deleting Secrets \(p. 52\)](#)
- [Summary \(p. 53\)](#)
- [Related Resources \(p. 53\)](#)
- [Using SAML Session Tags for ABAC \(p. 53\)](#)

Tutorial Overview

This tutorial shows how to create and test a policy that allows IAM roles with principal tags to access resources with matching tags. When a principal makes a request to AWS, their permissions are granted based on whether the principal and resource tags match. This strategy allows individuals to view or edit only the AWS resources required for their jobs.

Scenario

Assume that you're a lead developer at a large company named Example Corporation, and you're an experienced IAM administrator. You're familiar with creating and managing IAM users, roles, and policies. You want to ensure that your development engineers and quality assurance team members can access the resources they need. You also need a strategy that scales as your company grows.

You choose to use AWS resource tags and IAM role principal tags to implement an ABAC strategy for services that support it, beginning with AWS Secrets Manager. To learn which services support authorization based on tags, see [AWS Services That Work with IAM \(p. 586\)](#). To learn which tagging condition keys you can use in a policy with each service's actions and resources, see [Actions, Resources, and Condition Keys for AWS Services \(p. 686\)](#). You can configure your SAML-based or web identity provider to pass [session tags \(p. 299\)](#) to AWS. When your employees federate into AWS, their attributes are applied to their resulting principal in AWS. You can then use ABAC to allow or deny permissions based on those attributes. To learn how using session tags with a SAML federated identity differs from this tutorial, see [Using SAML Session Tags for ABAC \(p. 53\)](#).

Your Engineering and Quality Assurance team members are on either the **Pegasus** or **Unicorn** project. You choose the following 3-character project and team tag values:

- `access-project = peg` for the **Pegasus** project
- `access-project = uni` for the **Unicorn** project
- `access-team = eng` for the Engineering team
- `access-team = qas` for the Quality Assurance team

Additionally, you choose to require the `cost-center` cost allocation tag to enable custom AWS billing reports. For more information, see [Using Cost Allocation Tags](#) in the *AWS Billing and Cost Management User Guide*.

Summary of Key Decisions

- Employees sign in with IAM user credentials and then assume the IAM role for their team and project. If your company has its own identity system, you can set up federation to allow employees to assume a role without IAM users. For more information, see [Using SAML Session Tags for ABAC \(p. 53\)](#).
- The same policy is attached to all of the roles. Actions are allowed or denied based on tags.
- Employees can create new resources, but only if they attach the same tags to the resource that are applied to their role. This ensures that employees can view the resource after they create it. Administrators are no longer required to update policies with the ARN of new resources.
- Employees can read resources owned by their team, regardless of the project.
- Employees can update and delete resources owned by their own team and project.
- IAM administrators can add a new role for new projects. They can create and tag a new IAM user to allow access to the appropriate role. Administrators are not required to edit a policy to support a new project or team member.

In this tutorial, you will tag each resource, tag your project roles, and add policies to the roles to allow the behavior previously described. The resulting policy allows the roles Create, Read, Update, and Delete access to resources that are tagged with the same project and team tags. The policy also allows cross-project Read access for resources that are tagged with the same team.

Prerequisites

To perform the steps in this tutorial, you must already have the following:

- An AWS account that you can sign in to as an IAM user with administrative permissions. If you have a new account and sign in as the AWS account root user, then [create an IAM admin user \(p. 20\)](#).
- Your 12-digit account ID, which you use to create the roles in step 3.

To find your account ID, open the IAM console at <https://console.aws.amazon.com/iam/>. At the top of the left navigation pane, view your 12-digit account ID.

- Experience creating and editing IAM users, roles, and policies in the AWS Management Console. However, if you need help remembering an IAM management process, this tutorial provides links where you can view step-by-step instructions.

Step 1: Create Test Users

For testing, create four IAM users with permissions to assume roles with the same tags. This makes it easier to add more users to your teams. When you tag the users, they automatically get access to assume the correct role. You don't have to add the users to the trust policy of the role if they work on only one project and team.

1. Create the following customer managed policy named `access-assume-role`. For more information about creating a JSON policy, see [Creating IAM Policies \(Console\) \(p. 443\)](#).

ABAC Policy: Assume Any ABAC Role, But Only When the User and Role Tags Match

The following policy allows a user to assume any role in your account with the `access-` name prefix. The role must also be tagged with the same project, team, and cost center tags as the user.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "TutorialAssumeRole",  
            "Effect": "Allow",  
            "Action": "sts:AssumeRole",  
            "Condition": {"StringLike": {"aws:TagKeys": "Project,Team,CostCenter"}, "StringLike": {"aws:TagValues": "Project,Team,CostCenter"}},  
            "Resource": "arn:aws:iam::123456789012:role/*"  
        }  
    ]  
}
```

```

    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::123456789012:role/access-*",
    "Condition": {
        "StringEquals": {
            "iam:ResourceTag/access-project": "${aws:PrincipalTag/access-project}",
            "iam:ResourceTag/access-team": "${aws:PrincipalTag/access-team}",
            "iam:ResourceTag/cost-center": "${aws:PrincipalTag/cost-center}"
        }
    }
}
]
}
}

```

To scale this tutorial to a large number of users, you can attach the policy to a group and add each user to the group. For more information, see [Creating IAM Groups \(p. 169\)](#) and [Adding and Removing Users in an IAM Group \(p. 171\)](#).

2. Create the following IAM users, attach the access-assume-role permissions policy, and add the following tags. For more information about creating and tagging a new user, see [Creating IAM Users \(Console\) \(p. 87\)](#).

ABAC Users

User Name	User Tags
access-Arnav-peg-eng	access-project = peg access-team = eng cost-center = 987654
access-Mary-peg-qas	access-project = peg access-team = qas cost-center = 987654
access-Saanvi-uni-eng	access-project = uni access-team = eng cost-center = 123456
access-Carlos-uni-qas	access-project = uni access-team = qas cost-center = 123456

Step 2: Create the ABAC Policy

Create the following policy named **access-same-project-team**. You will add this policy to the roles in a later step. For more information about creating a JSON policy, see [Creating IAM Policies \(Console\) \(p. 443\)](#).

For additional policies that you can adapt for this tutorial, see the following pages:

- [Controlling Access for IAM Principals \(p. 391\)](#)

- [Amazon EC2: Allows Starting or Stopping EC2 Instances a User Has Tagged, Programmatically and in the Console \(p. 415\)](#)
- [EC2: Start or Stop Instances Based on Matching Principal and Resource Tags \(p. 416\)](#)
- [EC2: Start or Stop Instances Based on Tags \(p. 415\)](#)
- [IAM: Assume Roles That Have a Specific Tag \(p. 420\)](#)

ABAC Policy: Access Secrets Manager Resources Only When the Principal and Resource Tags Match

The following policy allows principals to create, read, edit, and delete resources, but only when those resources are tagged with the same key-value pairs as the principal. When a principal creates a resource, they must add `access-project`, `access-team`, and `cost-center` tags with values that match the principal's tags. The policy also allows adding optional `Name` or `OwnedBy` tags.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllActionsSecretsManagerSameProjectSameTeam",
            "Effect": "Allow",
            "Action": "secretsmanager:*",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/access-project": "${aws:PrincipalTag/access-project}",
                    "aws:ResourceTag/access-team": "${aws:PrincipalTag/access-team}",
                    "aws:ResourceTag/cost-center": "${aws:PrincipalTag/cost-center}"
                },
                "ForAllValues:StringEquals": {
                    "aws:TagKeys": [
                        "access-project",
                        "access-team",
                        "cost-center",
                        "Name",
                        "OwnedBy"
                    ]
                },
                "StringEqualsIfExists": {
                    "aws:RequestTag/access-project": "${aws:PrincipalTag/access-project}",
                    "aws:RequestTag/access-team": "${aws:PrincipalTag/access-team}",
                    "aws:RequestTag/cost-center": "${aws:PrincipalTag/cost-center}"
                }
            }
        },
        {
            "Sid": "AllResourcesSecretsManagerNoTags",
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetRandomPassword",
                "secretsmanager>ListSecrets"
            ],
            "Resource": "*"
        },
        {
            "Sid": "ReadSecretsManagerSameTeam",
            "Effect": "Allow",
            "Action": [
                "secretsmanager:Describe*",
                "secretsmanager:Get*",
                "secretsmanager>List*"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:TagKeys": [
                        "access-project",
                        "access-team",
                        "cost-center",
                        "Name",
                        "OwnedBy"
                    ]
                }
            }
        }
    ]
}
```

```

        "StringEquals": {
            "aws:ResourceTag/access-team": "${aws:PrincipalTag/access-team}"
        }
    },
    {
        "Sid": "DenyUntagSecretsManagerReservedTags",
        "Effect": "Deny",
        "Action": "secretsmanager:UntagResource",
        "Resource": "*",
        "Condition": {
            "StringLike": {
                "aws:TagKeys": "access-*"
            }
        }
    },
    {
        "Sid": "DenyPermissionsManagement",
        "Effect": "Deny",
        "Action": "secretsmanager:*Policy",
        "Resource": "*"
    }
]
}

```

What does this policy do?

- The `AllActionsSecretsManagerSameProjectSameTeam` statement allows all of this service's actions on all related resources, but only if the resource tags match the principal tags. By adding `"Action": "secretsmanager:/*"` to the policy, the policy grows as Secrets Manager grows. If Secrets Manager adds a new API operation, you are not required to add that action to the statement. The statement implements ABAC using three condition blocks. The request is allowed only if all three blocks return true.
 - The first condition block of this statement returns true if the specified tag keys are present on the resource, and their values match the principal's tags. This block returns false for mismatched tags, or for actions that don't support resource tagging. To learn which actions are not allowed by this block, see [Actions, Resources, and Condition Keys for AWS Secrets Manager](#). That page shows that actions performed on the `Secret` resource type support the `secretsmanager:ResourceTag/tag-key` condition key. Some [Secrets Manager actions](#) don't support that resource type, including `GetRandomPassword` and `ListSecrets`. You must create additional statements to allow those actions.
 - The second condition block returns true if every tag key passed in the request is included in the specified list. This is done using `ForAllValues` with the `StringEquals` condition operator. If no keys are passed, such as for a `Get*` operation, then the condition returns true. If the requester includes a tag key that is not in the list, the condition returns false. Every tag key that is passed in the request must match a member of this list. For more information, see [Using Multiple Keys and Values \(p. 622\)](#).
 - The third condition block returns true if the request supports passing tags, if all three of the tags are present, and if they match the principal tag values. This block also returns true if the request does not support passing tags. This is thanks to [...IfExists \(p. 620\)](#) in the condition operator. The block returns false if there is no tag passed during an action that supports it, or if the tag keys and values don't match.
- The `AllResourcesSecretsManagerNoTags` statement allows the `GetRandomPassword` and `ListSecrets` actions that are not allowed by the first statement.
- The `ReadSecretsManagerSameTeam` statement allows read-only operations if the principal is tagged with the same access-team tag as the resource. This is allowed regardless of the project or cost-center tag.

- The `DenyUntagSecretsManagerReservedTags` statement denies requests to remove tags with keys that begin with "access-" from Secrets Manager. These tags are used to control access to resources, therefore removing tags might remove permissions.
- The `DenyPermissionsManagement` statement denies access to create, edit, or delete Secrets Manager resource-based policies. These policies could be used to change the permissions of the secret.

Important

This policy uses a strategy to allow all actions for a service, but explicitly deny permissions-altering actions. Denying an action overrides any other policy that allows the principal to perform that action. This can have unintended results. As a best practice, use explicit denies only when there is no circumstance that should allow that action. Otherwise, allow a list of individual actions, and the unwanted actions are denied by default.

Step 3: Create Roles

Create the following IAM roles and attach the `access-same-project-team` policy that you created in the previous step. For more information about creating IAM roles, see [Creating a Role to Delegate Permissions to an IAM User \(p. 229\)](#). If you choose to use federation instead of IAM users and roles, see [Using SAML Session Tags for ABAC \(p. 53\)](#).

ABAC Roles

Job Function	Role Tags	Role Name	Role Description
Project Pegasus Engineering	<code>access-project = peg</code> <code>access-team = eng</code> <code>cost-center = 987654</code>	<code>access-peg-engineering</code>	Allows engineers to read all engineering resources and create and manage Pegasus engineering resources.
Project Pegasus Quality Assurance	<code>access-project = peg</code> <code>access-team = qas</code> <code>cost-center = 987654</code>	<code>access-peg-quality-assurance</code>	Allows the QA team to read all QA resources and create and manage all Pegasus QA resources.
Project Unicorn Engineering	<code>access-project = uni</code> <code>access-team = eng</code> <code>cost-center = 123456</code>	<code>access-uni-engineering</code>	Allows engineers to read all engineering resources and create and manage Unicorn engineering resources.
Project Unicorn Quality Assurance	<code>access-project = uni</code> <code>access-team = qas</code> <code>cost-center = 123456</code>	<code>access-uni-quality-assurance</code>	Allows the QA team to read all QA resources and create and manage all Unicorn QA resources.

Step 4: Test Creating Secrets

The permissions policy attached to the roles allows the employees to create secrets. This is allowed only if the secret is tagged with their project, team, and cost center. Confirm that your permissions are working as expected by signing in as your users, assuming the correct role, and testing activity in Secrets Manager.

To test creating a secret with and without the required tags

- In your main browser window, remain signed in as the administrator user so that you can review users, roles, and policies in IAM. Use a browser incognito window or separate browser for your testing. There, sign in as the access-Arnav-peg-eng IAM user and open the Secrets Manager console at <https://console.aws.amazon.com/secretsmanager/>.
- Attempt to switch to the access-uni-engineering role. For more information about switching roles in the AWS Management Console, see [Switching to a Role \(Console\) \(p. 260\)](#).

This operation fails because the access-team tag values do not match on the user and role.

- Switch to the access-peg-engineering role. For more information about switching roles in the AWS Management Console, see [Switching to a Role \(Console\) \(p. 260\)](#).
- Store a new secret using the following information. To learn how to store a secret, see [Creating a Basic Secret](#) in the *AWS Secrets Manager User Guide*.

- In the **Select secret type** section, choose **Other type of secrets**. In the two text boxes, enter `test-access-key` and `test-access-secret`.

You must have additional permissions to save credentials for specific AWS services. For example, to create credentials for an Amazon RDS database, you must have permission to describe RDS instances, RDS clusters, and Amazon Redshift clusters.

- Enter `test-access-peg-eng` for the **Secret name** field.
- Add different tag combinations from the following table and view the expected behavior.
- Choose **Store** to attempt to create the secret. When the storage fails, return to the previous Secrets Manager console pages and use the next tag set from the following table. The last tag set is allowed and will successfully create the secret.

ABAC Tag Combinations for `test-access-peg-eng` Role

access-project Tag Value	access-team Tag Value	cost-center Tag Value	Additional Tags	Expected Behavior
(none)	(none)	(none)	(none)	Denied because the access-project tag value does not match the role's value of peg.
uni	eng	987654	(none)	Denied because the access-project tag value does not match the role's value of peg.
peg	qas	987654	(none)	Denied because the access-team tag value does not match the role's value of eng.
peg	eng	123456	(none)	Denied because the cost-center tag value does not match the role's value of 987654.
peg	eng	987654	owner = Jane	Denied because the additional tag owner is not allowed by the policy, even though all three required tags are present and their values match the role's values.
peg	eng	987654	Name = Jane	Allowed because all three required tags are present and their values

access-project Tag Value	access-team Tag Value	cost-center Tag Value	Additional Tags	Expected Behavior
				match the role's values. You are also allowed to include the optional Name tag.

- Sign out and repeat the first three steps of this procedure for each of the following roles and tag values. In the fourth step in this procedure, test any set of missing tags, optional tags, disallowed tags, and invalid tag values that you choose. Then use the required tags to create a secret with the following tags and name.

ABAC Roles and Tags

User Name	Role Name	Secret Name	Secret Tags
access-Mary-peg-qas	access-peg-quality-assurance	test-access-peg-qas	access-project = peg access-team = qas cost-center = 987654
access-Saanvi-uni-eng	access-uni-engineering	test-access-uni-eng	access-project = uni access-team = eng cost-center = 123456
access-Carlos-uni-qas	access-uni-quality-assurance	test-access-uni-qas	access-project = uni access-team = qas cost-center = 123456

Step 5: Test Viewing Secrets

The policy that you attached to each role allows the employees to view any secrets tagged with their team name, regardless of their project. Confirm that your permissions are working as expected by testing your roles in Secrets Manager.

To test viewing a secret with and without the required tags

- Sign in as one of the following IAM users:
 - access-Arnav-peg-eng
 - access-Mary-peg-qas
 - access-Saanvi-uni-eng
 - access-Carlos-peg-qas
- Switch to the matching role:

- access-peg-engineering
- access-peg-quality-assurance
- access-uni-engineering
- access-peg-quality-assurance

For more information about switching roles in the AWS Management Console, see [Switching to a Role \(Console\) \(p. 260\)](#).

3. In the navigation pane on the left, choose the menu icon to expand the menu and then choose **Secrets**.
4. You should see all four secrets in the table, regardless of your current role. This is expected because the policy named `access-same-project-team` allows the `secretsmanager>ListSecrets` action for all resources.
5. Choose the name of one of the secrets.
6. On the details page for the secret, your role's tags determine whether you can view the page content. Compare the name of your role to the name of your secret. If they share the same team name, then the `access-team` tags match. If they don't match, then access is denied.

ABAC Secret viewing behavior for each role

Role Name	Secret Name	Expected Behavior
access-peg-engineering	test-access-peg-eng	Allowed
	test-access-peg-qas	Denied
	test-access-uni-eng	Allowed
	test-access-uni-qas	Denied
access-peg-quality-assurance	test-access-peg-eng	Denied
	test-access-peg-qas	Allowed
	test-access-uni-eng	Denied
	test-access-uni-qas	Allowed
access-uni-engineering	test-access-peg-eng	Allowed
	test-access-peg-qas	Denied
	test-access-uni-eng	Allowed
	test-access-uni-qas	Denied
access-peg-quality-assurance	test-access-peg-eng	Denied
	test-access-peg-qas	Allowed
	test-access-uni-eng	Denied
	test-access-uni-qas	Allowed

7. From the breadcrumbs at the top of the page, choose **Secrets** to return to the list of secrets. Repeat the steps in this procedure using different roles to test whether you can view each of the secrets.

Step 6: Test Scalability

An important reason for using attribute-based access control (ABAC) over role-based access control (RBAC) is scalability. As your company adds new projects, teams, or people to AWS, you don't need to update your ABAC-driven policies. For example, assume that Example Company is funding a new project, code named **Centaurs**. An engineer named Saanvi Sarkar will be the lead engineer for **Centaurs** while continuing to work on the **Unicorn** project. There are also several newly hired engineers, including Nikhil Jayashankar, who will work on only the **Centaurs** project.

To add the new project to AWS

1. Sign in as the IAM administrator user and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane on the left, choose **Roles** and add an IAM role named **access-cen-engineering**. Attach the **access-same-project-team** permissions policy to the role and add the following tags:
 - **access-project = cen**
 - **access-team = eng**
 - **cost-center = 101010**
3. In the navigation pane on the left, choose **Users**.
4. Add a new user named **access-Nikhil-cen-eng**, and attach the policy named **access-assume-role**.
5. Use the procedures in [Step 4: Test Creating Secrets \(p. 47\)](#) and [Step 5: Test Viewing Secrets \(p. 49\)](#). In another browser window, test that Nikhil can create only **Centaurs** engineering secrets, and that he can view all engineering secrets.
6. In the main browser window where you signed in as the administrator, choose **access-Saanvi-uni-eng**.
7. On the **Permissions** tab, remove the **access-assume-role** permissions policy.
8. Add the following inline policy named **access-assume-specific-roles**. For more information about adding an inline policy to a user, see [To embed an inline policy for a user or role \(console\) \(p. 459\)](#).

ABAC Policy: Assume Only Specific Roles

This policy allows Saanvi to assume the engineering roles for the Pegasus or **Centaurs** projects. It is necessary to create this custom policy because IAM does not support multivalued tags. You can't tag Saanvi's user with **access-project = peg** and **access-project = cen**. Additionally, the AWS authorization model can't match both values. For more information, see [Rules for Tagging in IAM And AWS STS \(p. 295\)](#). Instead, you must manually specify the two roles that she can assume.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "TutorialAssumeSpecificRoles",  
            "Effect": "Allow",  
            "Action": "sts:AssumeRole",  
            "Resource": [  
                "arn:aws:iam::123456789012:role/access-peg-engineering",  
                "arn:aws:iam::123456789012:role/access-cen-engineering"  
            ]  
        }  
    ]  
}
```

9. Use the procedures in [Step 4: Test Creating Secrets \(p. 47\)](#) and [Step 5: Test Viewing Secrets \(p. 49\)](#). In another browser window, confirm that Saanvi can assume both roles. Check that she can create secrets for only her project, team, and cost center, depending on the role's tags. Also confirm that she can view details about any secrets owned by the engineering team, including the ones that she just created.

Step 7: Test Updating and Deleting Secrets

The access-same-project-team policy that is attached to the roles allows the employees to update and delete any secrets tagged with their project, team, and cost center. Confirm that your permissions are working as expected by testing your roles in Secrets Manager.

To test updating and deleting a secret with and without the required tags

1. Sign in as one of the following IAM users:

- access-Arnav-peg-eng
- access-Mary-peg-qas
- access-Saanvi-uni-eng
- access-Carlos-peg-qas
- access-Nikhil-cen-eng

2. Switch to the matching role:

- access-peg-engineering
- access-peg-quality-assurance
- access-uni-engineering
- access-peg-quality-assurance
- access-cen-engineering

For more information about switching roles in the AWS Management Console, see [Switching to a Role \(Console\) \(p. 260\)](#).

3. For each role, try to update the secret description and then try to delete the following secrets. For more information, see [Modifying a Secret](#) and [Deleting and Restoring a Secret](#) in the *AWS Secrets Manager User Guide*.

ABAC Secret updating and deleting behavior for each role

Role Name	Secret Name	Expected Behavior
access-peg-engineering	test-access-peg-eng	Allowed
	test-access-uni-eng	Denied
	test-access-uni-qas	Denied
access-peg-quality-assurance	test-access-peg-qas	Allowed
	test-access-uni-eng	Denied
access-uni-engineering	test-access-uni-eng	Allowed
	test-access-uni-qas	Denied
access-peg-quality-assurance	test-access-uni-qas	Allowed

Summary

You've now successfully completed all of the steps necessary to use tags for attribute-based access control (ABAC). You've learned how to define a tagging strategy. You applied that strategy to your principals and resources. You created and applied a policy that enforces the strategy for Secrets Manager. You also learned that ABAC scales easily when you add new projects and team members. As a result, you are able to sign in to the IAM console with your test roles and experience how to use tags for ABAC in AWS.

Related Resources

For related information in the *IAM User Guide*, see the following resources:

- [What Is ABAC for AWS? \(p. 12\)](#)
- [AWS Global Condition Context Keys \(p. 665\)](#)
- [Creating Your First IAM Admin User and Group \(p. 20\)](#)
- [Creating IAM Users \(Console\) \(p. 87\)](#)
- [Creating a Role to Delegate Permissions to an IAM User \(p. 229\)](#)
- [Tagging IAM Users and Roles \(p. 295\)](#)
- [Controlling Access to AWS Resources Using Resource Tags \(p. 392\)](#)
- [Switching to a Role \(Console\) \(p. 260\)](#)
- [Using SAML Session Tags for ABAC \(p. 53\)](#)

To learn how to monitor the tags in your account, see [Monitor tag changes on AWS resources with serverless workflows and Amazon CloudWatch Events](#).

Using SAML Session Tags for ABAC

Attribute-based access control (ABAC) is an authorization strategy that defines permissions based on attributes. In AWS, these attributes are called tags. You can [attach tags to IAM entities \(p. 295\)](#) (users or roles) and to AWS resources. When the entities are used to make requests to AWS, they become principals and those principals include tags.

You can also pass [session tags \(p. 299\)](#) when you assume a role or federate a user. You can then define policies that use tag condition keys to grant permissions to your principals based on their tags. When you use tags to control access to your AWS resources, you allow your teams and resources to grow with fewer changes to AWS policies. ABAC policies are more flexible than traditional AWS policies, which require you to list each individual resource. For more information about ABAC and its advantage over traditional policies, see [What Is ABAC for AWS? \(p. 12\)](#).

If your company uses a SAML-based identity provider (IdP) to manage corporate user identities, you can use SAML attributes for fine-grained access control in AWS. Attributes can include cost center identifiers, user email addresses, department classifications, and project assignments. When you pass these attributes as session tags, you can then control access to AWS based on these session tags.

To complete the [ABAC tutorial \(p. 41\)](#) by passing SAML attributes to your session principal, complete the tasks in [Tutorial: Using Tags for Attribute-Based Access Control in AWS \(p. 41\)](#), with the changes that are included in this topic.

Prerequisites

To perform the steps to use SAML session tags for ABAC, you must already have the following:

- Access to a SAML-based IdP where you can create test users with specific attributes.
- An AWS account that you can sign in to as an IAM user with administrative permissions. If you have a new account and sign in as the AWS account root user, then [create an IAM admin user \(p. 20\)](#).
- Experience creating and editing IAM users, roles, and policies in the AWS Management Console. However, if you need help remembering an IAM management process, the ABAC tutorial provides links where you can view step-by-step instructions.
- Experience setting up a SAML-based IdP in IAM. To view more details and links to detailed IAM documentation, see [Passing Session Tags using AssumeRoleWithSAML \(p. 303\)](#).

Step 1: Create a Test IAM User

Skip the instructions in [Step 1: Create Test Users \(p. 43\)](#). Because your identities are defined in your provider, it's not necessary for you to add IAM users for your employees.

Step 2: Create the ABAC Policy

Follow the instructions in [Step 2: Create the ABAC Policy \(p. 44\)](#) to create the specified managed policy in IAM.

Step 3: Create and Configure the SAML Role

When you use the ABAC tutorial for SAML, you must perform additional steps to create the role, configure the SAML IdP, and enable AWS Management Console access. For more information, see [Step 3: Create Roles \(p. 47\)](#).

Step 3A: Create the SAML Role

Create a single role that trusts your SAML identity provider and the `test-session-tags` user that you created in step 1. The ABAC tutorial uses separate roles with different role tags. Because you are passing session tags from your SAML IdP, you need only one role. To learn how to create a SAML-based role, see [Creating a Role for SAML 2.0 Federation \(Console\) \(p. 248\)](#).

Name the role `access-session-tags`. Attach the `access-same-project-team` permissions policy to the role. Edit the role trust policy to use the following policy. For detailed instructions on how to edit the trust relationship of a role, see [Modifying a Role \(Console\) \(p. 279\)](#).

The following role trust policy allows your SAML identity provider and the `test-session-tags` user to assume the role. When they assume the role, they must pass the three specified session tags. The `sts:TagSession` action is required to allow passing session tags.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "AllowSamIdentityAssumeRole",  
            "Effect": "Allow",  
            "Action": [  
                "sts:AssumeRoleWithSAML",  
                "sts:TagSession"  
            ],  
            "Principal": {"Federated": "arn:aws:iam::123456789012:saml-provider/ExampleCorpProvider"},  
            "Condition": {  
                "StringLike": {  
                    "aws:RequestTag/cost-center                    "aws:RequestTag/access-project                    "aws:RequestTag/access-team                }  
        }  
    ]  
}
```

```
        "eng",
        "qas"
    ],
    "StringEquals": { "SAML:aud": "https://signin.aws.amazon.com/saml" }
}
]
}
```

The `AllowSamlIdentityAssumeRole` statement allows members of the Engineering and Quality Assurance teams to assume this role when they federate into AWS from the Example Corporation IdP. The `ExampleCorpProvider` SAML provider is defined in IAM. The administrator has already set up the SAML assertion to pass the three required session tags. The assertion can pass additional tags, but these three must be present. The identity's attributes can have any value for the `cost-center` and `access-project` tags. However, the `access-team` attribute value must match `eng` or `qas` to indicate that the identity is on the Engineering or Quality Assurance team.

Step 3B: Configure the SAML IdP

Configure your SAML IdP to pass the `cost-center`, `access-project`, and `access-team` attributes as session tags. For more information, see [Passing Session Tags using AssumeRoleWithSAML \(p. 303\)](#).

To pass these attributes as session tags, include the following elements in your SAML assertion.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:cost-center">
    <AttributeValue>987654</AttributeValue>
</Attribute>
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:access-project">
    <AttributeValue>peg</AttributeValue>
</Attribute>
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:access-team">
    <AttributeValue>eng</AttributeValue>
</Attribute>
```

Step 3B: Enable Console Access

Enable console access for your federated SAML users. For more information, see [Enabling SAML 2.0 Federated Users to Access the AWS Management Console \(p. 211\)](#).

Step 4: Test Creating Secrets

Federate into the AWS Management Console using the `access-session-tags` role. For more information, see [Enabling SAML 2.0 Federated Users to Access the AWS Management Console \(p. 211\)](#). Then follow the instructions in [Step 4: Test Creating Secrets \(p. 47\)](#) to create secrets. Use different SAML identities with attributes to match the tags that are indicated in the ABAC tutorial. For more information, see [Step 4: Test Creating Secrets \(p. 47\)](#).

Step 5: Test Viewing Secrets

Follow the instructions in [Step 5: Test Viewing Secrets \(p. 49\)](#) to view the secrets that you created in the previous step. Use different SAML identities with attributes to match the tags that are indicated in the ABAC tutorial.

Step 6: Test Scalability

Follow the instructions in [Step 6: Test Scalability \(p. 51\)](#) to test scalability. Do this by adding a new identity in your SAML-based IdP with the following attributes:

- `cost-center = 101010`
- `access-project = cen`
- `access-team = eng`

Step 7: Test Updating and Deleting Secrets

Follow the instructions in [Step 7: Test Updating and Deleting Secrets \(p. 52\)](#) to update and delete secrets. Use different SAML identities with attributes to match the tags that are indicated in the ABAC tutorial.

Important

Delete all of the secrets that you created to avoid billing charges. For details about pricing in Secrets Manager, see [AWS Secrets Manager Pricing](#).

Tutorial: Enable Your Users to Configure Their Own Credentials and MFA Settings

You can enable your users to self-manage their own multi-factor authentication (MFA) devices and credentials on the [My Security Credentials](#) page. You can use the AWS Management Console to configure credentials (access keys, passwords, signing certificates, and SSH public keys) and MFA devices for your users. This is useful for a small number of users. But that task could quickly become time consuming as the number of users grows. Security best practices specify that users should regularly change their passwords and rotate their access keys. They should also delete or deactivate credentials that are not needed. We also highly recommend that they use MFA for sensitive operations. This tutorial shows you how to enable these best practices without burdening your administrators.

This tutorial shows how to allow users to access AWS services, but **only** when they sign in with MFA. If they are not signed in with an MFA device, then users cannot access other services.

This workflow has three basic steps.

[Step 1: Create a Policy to Enforce MFA Sign-In \(p. 57\)](#)

Create a customer managed policy that prohibits all actions *except* the few IAM actions that allow a user to change their own credentials and manage their MFA devices on the [My Security Credentials](#) page. For more information about accessing that page

[Step 2: Attach Policies to Your Test Group \(p. 57\)](#)

Create a group whose members have full access to all Amazon EC2 actions if they sign in with MFA. To create such a group, you attach both the AWS managed policy called `AmazonEC2FullAccess` and the customer managed policy you created in the first step.

[Step 3: Test Your User's Access \(p. 58\)](#)

Sign in as the test user to verify that access to Amazon EC2 is blocked *until* the user creates an MFA device. The user can then sign in using that device.

Prerequisites

To perform the steps in this tutorial, you must already have the following:

- An AWS account that you can sign in to as an IAM user with administrative permissions.

- Your account ID number, which you type into the policy in Step 1.

To find your account ID number, on the navigation bar at the top of the page, choose **Support** and then choose **Support Center**. You can find your account ID under this page's **Support** menu.

- A [virtual \(software-based\) MFA device \(p. 122\)](#), [U2F security key \(p. 125\)](#), or [hardware-based MFA device \(p. 130\)](#).
- A test IAM user who is a member of a group as follows:

Create User Account		Create and Configure Group Account		
MFAUser	Choose only the option for AWS Management Console access , and assign a password.	EC2MFA	MFAUser	Do NOT attach any policies or otherwise grant permissions to this group.

Step 1: Create a Policy to Enforce MFA Sign-In

You begin by creating an IAM customer managed policy that denies all permissions except those required for IAM users to manage their own credentials and MFA devices.

1. Sign in to the AWS Management Console as a user with administrator credentials. To adhere to IAM best practices, don't sign in with your AWS account root user credentials. For more information, see [Create individual IAM users](#).
2. Open the IAM console at <https://console.aws.amazon.com/iam/>.
3. In the navigation pane, choose **Policies**, and then choose **Create policy**.
4. Choose the **JSON** tab and copy the text from the following JSON policy document: [AWS: Allows MFA-Authenticated IAM Users to Manage Their Own Credentials on the My Security Credentials Page \(p. 398\)](#).
5. Paste the policy text into the **JSON** text box, then choose **Review policy**. The [Policy Validator \(p. 448\)](#) reports any syntax errors.

Note

You can switch between the **Visual editor** and **JSON** tabs anytime. However, the policy above includes the `NotAction` element, which is not supported in the visual editor. For this policy, you will see a notification on the **Visual editor** tab. Return to the **JSON** tab to continue working with this policy.

6. On the **Review** page, type `Force_MFA` for the policy name. For the policy description, type `This policy allows users to manage their own passwords and MFA devices but nothing else unless they authenticate with MFA.` Review the policy **Summary** to see the permissions granted by your policy, and then choose **Create policy** to save your work.

The new policy appears in the list of managed policies and is ready to attach.

Step 2: Attach Policies to Your Test Group

Next you attach two policies to the test IAM group, which will be used to grant the MFA-protected permissions.

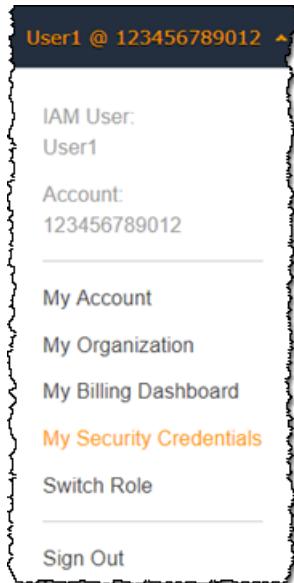
1. In the navigation pane, choose **Groups**.
2. In the search box, type `EC2MFA`, and then choose the group name (not the check box) in the list.
3. On the **Permissions** tab, and click **Attach Policy**.

4. On the **Attach Policy** page, in the search box, type **EC2Full** and then select the check box next to **AmazonEC2FullAccess** in the list. Don't save your changes yet.
5. In the search box, type **Force**, and then select the check box next to **Force_MFA** in the list.
6. Choose **Attach Policy**.

Step 3: Test Your User's Access

In this part of the tutorial, you sign in as the test user and verify that the policy works as intended.

1. Sign in to your AWS account as **MFAUser** with the password you assigned in the previous section. Use the URL: <https://<alias or account ID number>.signin.aws.amazon.com/console>
2. Choose **EC2** to open the Amazon EC2 console and verify that the user has no permissions to do anything.
3. In the navigation bar on the upper right, choose the **MFAUser** user name, and then choose **My Security Credentials**.



4. Now add an MFA device. In the **Multi-factor Authentication (MFA)** section, choose **Assign MFA device**.
- Note**
You might receive an error that you are not authorized to perform `iam>DeleteVirtualMFADevice`. This could happen if someone previously began assigning a virtual MFA device to this user and cancelled the process. To continue, you or another administrator must delete the user's existing MFA device. For more information, see [I Am Not Authorized to Perform: iam>DeleteVirtualMFADevice \(p. 547\)](#).
5. For this tutorial, we use a virtual (software-based) MFA device, such as the Google Authenticator app on a mobile phone. Choose **Virtual MFA device**, and then click **Continue**.
- IAM generates and displays configuration information for the virtual MFA device, including a QR code graphic. The graphic is a representation of the secret configuration key that is available for manual entry on devices that do not support QR codes.
6. Open your virtual MFA app. (For a list of apps that you can use for hosting virtual MFA devices, see [Virtual MFA Applications](#).) If the virtual MFA app supports multiple accounts (multiple virtual MFA devices), choose the option to create a new account (a new virtual MFA device).
 7. Determine whether the MFA app supports QR codes, and then do one of the following:

- From the wizard, choose **Show QR code**. Then use the app to scan the QR code. For example, you might choose the camera icon or choose an option similar to **Scan code**, and then use the device's camera to scan the code.
- In the **Manage MFA Device** wizard, choose **Show secret key**, and then type the secret key into your MFA app.

When you are finished, the virtual MFA device starts generating one-time passwords.

8. In the **Manage MFA Device** wizard, in the **MFA Code 1** box, type the one-time password that currently appears in the virtual MFA device. Wait up to 30 seconds for the device to generate a new one-time password. Then type the second one-time password into the **MFA Code 2** box. Choose **Assign MFA**.

Important

Submit your request immediately after generating the codes. If you generate the codes and then wait too long to submit the request, the MFA device is successfully associated with the user. However, the MFA device is out of sync. This happens because time-based one-time passwords (TOTP) expire after a short period of time. If this happens, you can [resync the device \(p. 138\)](#).

The virtual MFA device is now ready to use with AWS.

9. Sign out of the console and then sign in as **MFAUser** again. This time AWS prompts you for an MFA code from your phone. When you get it, type the code in the box and then choose **Submit**.
10. Choose **EC2** to open the Amazon EC2 console again. Note that this time you can see all the information and perform any actions you want. If you go to any other console as this user, you see access denied messages. The reason is that the policies in this tutorial grant access only to Amazon EC2.

Related Resources

For related information found in the *IAM User Guide*, see the following resources:

- [Using Multi-Factor Authentication \(MFA\) in AWS \(p. 119\)](#)
- [Enabling MFA Devices \(p. 120\)](#)
- [Using MFA Devices With Your IAM Sign-in Page \(p. 78\)](#)

IAM Best Practices and Use Cases

To get the greatest benefits from IAM, take time to learn the recommended best practices. One way to do this is to see how IAM is used in real-world scenarios to work with other AWS services.

Topics

- [IAM Best Practices \(p. 60\)](#)
- [Business Use Cases \(p. 67\)](#)

IAM Best Practices



To help secure your AWS resources, follow these recommendations for the AWS Identity and Access Management (IAM) service.

Topics

- [Lock Away Your AWS Account Root User Access Keys \(p. 60\)](#)
- [Create Individual IAM Users \(p. 61\)](#)
- [Use Groups to Assign Permissions to IAM Users \(p. 61\)](#)
- [Grant Least Privilege \(p. 61\)](#)
- [Get Started Using Permissions with AWS Managed Policies \(p. 62\)](#)
- [Use Customer Managed Policies Instead of Inline Policies \(p. 63\)](#)
- [Use Access Levels to Review IAM Permissions \(p. 63\)](#)
- [Configure a Strong Password Policy for Your Users \(p. 64\)](#)
- [Enable MFA \(p. 64\)](#)
- [Use Roles for Applications That Run on Amazon EC2 Instances \(p. 65\)](#)
- [Use Roles to Delegate Permissions \(p. 65\)](#)
- [Do Not Share Access Keys \(p. 65\)](#)
- [Rotate Credentials Regularly \(p. 65\)](#)
- [Remove Unnecessary Credentials \(p. 66\)](#)
- [Use Policy Conditions for Extra Security \(p. 66\)](#)
- [Monitor Activity in Your AWS Account \(p. 66\)](#)
- [Video Presentation About IAM Best Practices \(p. 67\)](#)

Lock Away Your AWS Account Root User Access Keys

You use an access key (an access key ID and secret access key) to make programmatic requests to AWS. However, do not use your AWS account root user access key. The access key for your AWS account root user gives full access to all your resources for all AWS services, including your billing information. You cannot reduce the permissions associated with your AWS account root user access key.

Therefore, protect your root user access key like you would your credit card numbers or any other sensitive secret. Here are some ways to do that:

- If you don't already have an access key for your AWS account root user, don't create one unless you absolutely need to. Instead, use your account email address and password to sign in to the AWS Management Console and [create an IAM user for yourself \(p. 20\)](#) that has administrative permissions.
- If you do have an access key for your AWS account root user, delete it. If you must keep it, rotate (change) the access key regularly. To delete or rotate your root user access keys, go to the [My Security Credentials page](#) in the AWS Management Console and sign in with your account's email address and password. You can manage your access keys in the **Access keys** section. For more information about rotating access keys, see [Rotating Access Keys \(p. 115\)](#).
- Never share your AWS account root user password or access keys with anyone. The remaining sections of this document discuss various ways to avoid having to share your AWS account root user credentials with other users and to avoid having to embed them in an application.
- Use a strong password to help protect account-level access to the AWS Management Console. For information about managing your AWS account root user password, see [Changing the AWS Account Root User Password \(p. 100\)](#).
- Enable AWS multi-factor authentication (MFA) on your AWS account root user account. For more information, see [Using Multi-Factor Authentication \(MFA\) in AWS \(p. 119\)](#).

Create Individual IAM Users

Don't use your AWS account root user credentials to access AWS, and don't give your credentials to anyone else. Instead, create individual users for anyone who needs access to your AWS account. Create an IAM user for yourself as well, give that user administrative permissions, and use that IAM user for all your work. For information about how to do this, see [Creating Your First IAM Admin User and Group \(p. 20\)](#).

By creating individual IAM users for people accessing your account, you can give each IAM user a unique set of security credentials. You can also grant different permissions to each IAM user. If necessary, you can change or revoke an IAM user's permissions anytime. (If you give out your root user credentials, it can be difficult to revoke them, and it is impossible to restrict their permissions.)

Note

Before you set permissions for individual IAM users, though, see the next point about groups.

Use Groups to Assign Permissions to IAM Users

Instead of defining permissions for individual IAM users, it's usually more convenient to create groups that relate to job functions (administrators, developers, accounting, etc.). Next, define the relevant permissions for each group. Finally, assign IAM users to those groups. All the users in an IAM group inherit the permissions assigned to the group. That way, you can make changes for everyone in a group in just one place. As people move around in your company, you can simply change what IAM group their IAM user belongs to.

For more information, see the following:

- [Creating Your First IAM Admin User and Group \(p. 20\)](#)
- [Managing IAM Groups \(p. 170\)](#)

Grant Least Privilege

When you create IAM policies, follow the standard security advice of granting *least privilege*, or granting only the permissions required to perform a task. Determine what users (and roles) need to do and then craft policies that allow them to perform *only* those tasks.

Start with a minimum set of permissions and grant additional permissions as necessary. Doing so is more secure than starting with permissions that are too lenient and then trying to tighten them later.

You can use access level groupings to understand the level of access that a policy grants. [Policy actions \(p. 606\)](#) are classified as List, Read, Write, Permissions management, or Tagging. For example, you can choose actions from the List and Read access levels to grant read-only access to your users. To learn how to use policy summaries to understand access level permissions, see [Use Access Levels to Review IAM Permissions \(p. 63\)](#).

One feature that can help with this is *service last accessed data*. View this data on the **Access Advisor** tab on the IAM console details page for an IAM user, group, role, or policy. If you sign in using AWS Organizations master account credentials, you can view this data in the **AWS Organizations** section of the IAM console. You can also use the AWS CLI or AWS API to retrieve a report for service last accessed data for entities or policies in IAM or Organizations. You can use this information to identify unnecessary permissions so that you can refine your IAM or Organizations policies to better adhere to the principle of least privilege. For more information, see [Refining Permissions Using Service Last Accessed Data \(p. 475\)](#).

To further reduce permissions, you can view your account's events in AWS CloudTrail **Event history**. CloudTrail event logs include detailed event information that you can use to reduce the policy's permissions. The logs include only the actions and resources that your IAM entities need. For more information, see [Viewing CloudTrail Events in the CloudTrail Console](#) in the *AWS CloudTrail User Guide*.

For more information, see the following:

- [Access Management \(p. 355\)](#)
- Policy topics for individual services, which provide examples of how to write policies for service-specific resources. Examples:
 - [Authentication and Access Control for Amazon DynamoDB](#) in the *Amazon DynamoDB Developer Guide*
 - [Using Bucket Policies and User Policies](#) in the *Amazon Simple Storage Service Developer Guide*
 - [Access Control List \(ACL\) Overview](#) in the *Amazon Simple Storage Service Developer Guide*

Get Started Using Permissions with AWS Managed Policies

Providing your employees with only the permissions they need requires time and detailed knowledge of IAM policies. Employees need time to learn which AWS services they want or need to use. Administrators need time to learn about and test IAM.

To get started quickly, you can use AWS managed policies to give your employees the permissions they need to get started. These policies are already available in your account and are maintained and updated by AWS. For more information about AWS managed policies, see [AWS Managed Policies \(p. 365\)](#).

AWS managed policies are designed to provide permissions for many common use cases. Full access AWS managed policies such as [AmazonDynamoDBFullAccess](#) and [IAMFullAccess](#) define permissions for service administrators by granting full access to a service. Power-user AWS managed policies such as [AWSCodeCommitPowerUser](#) and [AWSKeyManagementServicePowerUser](#) provide multiple levels of access to AWS services without allowing permissions management permissions. Partial-access AWS managed policies such as [AmazonMobileAnalyticsWriteOnlyAccess](#) and [AmazonEC2ReadOnlyAccess](#) provide specific levels of access to AWS services. 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.

AWS managed policies for job functions can span multiple services and align with common job functions in the IT industry. For a list and descriptions of job function policies, see [AWS Managed Policies for Job Functions \(p. 656\)](#).

Use Customer Managed Policies Instead of Inline Policies

For custom policies, we recommend that you use managed policies instead of inline policies. A key advantage of using these policies is that you can view all of your managed policies in one place in the console. You can also view this information with a single AWS CLI or AWS API operation. Inline policies are policies that exist only on an IAM identity (user, group, or role). Managed policies are separate IAM resources that you can attach to multiple identities. For more information, see [Managed Policies and Inline Policies \(p. 364\)](#).

If you have inline policies in your account, you can convert them to managed policies. To do this, copy the policy to a new managed policy, attach the new policy to the identity that has the inline policy. Then delete the inline policy. You can do this using the instructions below.

To convert an inline policy to a managed policy

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Groups, Users, or Roles**.
3. In the list, choose the name of the group, user, or role that has the policy you want to remove.
4. Choose the **Permissions** tab. If you chose **Groups**, expand the **Inline Policies** section if necessary.
5. For groups, choose **Show Policy** next to the inline policy that you want to remove. For users and roles, choose **Show n more**, if necessary, and then choose the arrow next to the inline policy that you want to remove.
6. Copy the JSON policy document for the policy.
7. In the navigation pane, choose **Policies**.
8. Choose **Create policy** and then choose the **JSON** tab.
9. Replace the existing text with your JSON policy text, and then choose **Review policy**.
10. Enter a name for your policy and choose **Create policy**.
11. In the navigation pane, choose **Groups, Users, or Roles**, and again choose the name of the group, user, or role that has the policy you want to remove.
12. For groups, choose **Attach Policy**. For users and roles, choose **Add permissions**.
13. For groups, select the check box next to the name of your new policy, and then choose **Attach Policy**. For users or roles, choose **Add permissions**. On the next page, choose **Attach existing policies directly**, select the check box next to the name of your new policy, choose **Next: Review**, and then choose **Add permissions**.

You are returned to the **Summary** page for your group, user, or role.

14. For groups, choose **Remove Policy** next to the inline policy that you want to remove. For users or roles, choose **X** next to the inline policy that you want to remove.

In some circumstances, we do recommend choosing inline policies over managed policies. For details, see [Choosing Between Managed Policies and Inline Policies \(p. 368\)](#).

Use Access Levels to Review IAM Permissions

To improve the security of your AWS account, you should regularly review and monitor each of your IAM policies. Make sure that your policies grant the [least privilege \(p. 61\)](#) that is needed to perform only the necessary actions.

When you review a policy, you can view the [policy summary \(p. 491\)](#) that includes a summary of the access level for each service within that policy. AWS categorizes each service action into one of five *access levels* based on what each action does: List, Read, Write, Permissions management, or Tagging. You can use these access levels to determine which actions to include in your policies.

For example, in the Amazon S3 service, you might want to allow a large group of users to access List and Read actions. Such actions permit those users to list the buckets and get objects in Amazon S3. However, you should allow only a small group of users to access the Amazon S3 write actions to delete buckets or put objects into an S3 bucket. Additionally, you should reduce permissions to allow only administrators to access the Amazon S3 Permissions management actions. This ensures that only a limited number of people can manage bucket policies in Amazon S3. This is especially important for Permissions management actions in IAM and AWS Organizations services. Allowing Tagging actions grants a user permission to perform actions that only modify tags for a resource. However, some Write actions, such as CreateRole, allow tagging a resource when you create the resource or modify other attributes for that resource. Therefore, denying access to Tagging actions does not prevent a user from tagging resources. For details and examples of the access level classification, see [Understanding Access Level Summaries Within Policy Summaries \(p. 501\)](#).

To view the access level classification that is assigned to each action in a service, see [Actions, Resources, and Condition Keys for AWS Services \(p. 686\)](#).

To see the access levels for a policy, you must first locate the policy's summary. The policy summary is included on the **Policies** page for managed policies, and on the **Users** page for policies that are attached to a user. For more information, see [Policy Summary \(List of Services\) \(p. 492\)](#).

Within a policy summary, the **Access level** column shows that the policy provides **Full** or **Limited** access to one or more of the four AWS access levels for the service. Alternately, it might show that the policy provides **Full access** to all the actions within the service. You can use the information within this **Access level** column to understand the level of access that the policy provides. You can then take action to make your AWS account more secure. For details and examples of the access level classification, see [Understanding Access Level Summaries Within Policy Summaries \(p. 501\)](#).

Configure a Strong Password Policy for Your Users

If you allow users to change their own passwords, require that they create strong passwords and that they rotate their passwords periodically. On the [Account Settings](#) page of the IAM console, you can create a password policy for your account. You can use the password policy to define password requirements, such as minimum length, whether it requires non-alphabetic characters, how frequently it must be rotated, and so on.

For more information, see [Setting an Account Password Policy for IAM Users \(p. 101\)](#).

Enable MFA

For extra security, we recommend that you require multi-factor authentication (MFA) for all users in your account. With MFA, users have a device that generates a response to an authentication challenge. Both the user's credentials and the device-generated response are required to complete the sign-in process. If a user's password or access keys are compromised, your account resources are still secure because of the additional authentication requirement.

The response is generated in one of the following ways:

- Virtual and hardware MFA devices generate a code that you view on the app or device and then enter on the sign-in screen.
- U2F security keys generate a response when you tap the device. The user does not manually enter a code on the sign-in screen.

For privileged IAM users who are allowed to access sensitive resources or API operations, we recommend using U2F or hardware MFA devices.

For more information about MFA, see [Using Multi-Factor Authentication \(MFA\) in AWS \(p. 119\)](#).

To learn how to configure MFA-protected API access for access keys, see [Configuring MFA-Protected API Access \(p. 145\)](#).

Use Roles for Applications That Run on Amazon EC2 Instances

Applications that run on an Amazon EC2 instance need credentials in order to access other AWS services. To provide credentials to the application in a secure way, use IAM *roles*. A role is an entity that has its own set of permissions, but that isn't a user or group. Roles also don't have their own permanent set of credentials the way IAM users do. In the case of Amazon EC2, IAM dynamically provides temporary credentials to the EC2 instance, and these credentials are automatically rotated for you.

When you launch an EC2 instance, you can specify a role for the instance as a launch parameter. Applications that run on the EC2 instance can use the role's credentials when they access AWS resources. The role's permissions determine what the application is allowed to do.

For more information, see [Using an IAM Role to Grant Permissions to Applications Running on Amazon EC2 Instances \(p. 269\)](#).

Use Roles to Delegate Permissions

Don't share security credentials between accounts to allow users from another AWS account to access resources in your AWS account. Instead, use IAM roles. You can define a role that specifies what permissions the IAM users in the other account are allowed. You can also designate which AWS accounts have the IAM users that are allowed to assume the role. To learn whether principals in accounts outside of your zone of trust (trusted organization, OU, or account) have access to assume your roles, see [What is IAM Access Analyzer?](#).

For more information, see [Roles Terms and Concepts \(p. 176\)](#).

Do Not Share Access Keys

Access keys provide programmatic access to AWS. Do not embed access keys within unencrypted code or share these security credentials between users in your AWS account. For applications that need access to AWS, configure the program to retrieve temporary security credentials using an IAM role. To allow your users individual programmatic access, create an IAM user with personal access keys.

For more information, see [Switching to an IAM Role \(AWS API\) \(p. 268\)](#) and [Managing Access Keys for IAM Users \(p. 110\)](#).

Rotate Credentials Regularly

Change your own passwords and access keys regularly, and make sure that all IAM users in your account do as well. That way, if a password or access key is compromised without your knowledge, you limit how long the credentials can be used to access your resources. You can apply a password policy to your account to require all your IAM users to rotate their passwords. You can also choose how often they must do so.

For more information about setting a password policy in your account, see [Setting an Account Password Policy for IAM Users \(p. 101\)](#).

For more information about rotating access keys for IAM users, see [Rotating Access Keys \(p. 115\)](#).

Remove Unnecessary Credentials

Remove IAM user credentials (passwords and access keys) that are not needed. For example, if you created an IAM user for an application that does not use the console, then the IAM user does not need a password. Similarly, if a user only uses the console, remove their access keys. Passwords and access keys that have not been used recently might be good candidates for removal. You can find unused passwords or access keys using the console, using the CLI or API, or by downloading the credentials report.

For more information about finding IAM user credentials that have not been used recently, see [Finding Unused Credentials \(p. 154\)](#).

For more information about deleting passwords for an IAM user, see [Managing Passwords for IAM Users \(p. 104\)](#).

For more information about deactivating or deleting access keys for an IAM user, see [Managing Access Keys for IAM Users \(p. 110\)](#).

For more information about IAM credential reports, see [Getting Credential Reports for Your AWS Account \(p. 156\)](#).

Use Policy Conditions for Extra Security

To the extent that it's practical, define the conditions under which your IAM policies allow access to a resource. For example, you can write conditions to specify a range of allowable IP addresses that a request must come from. You can also specify that a request is allowed only within a specified date range or time range. You can also set conditions that require the use of SSL or MFA (multi-factor authentication). For example, you can require that a user has authenticated with an MFA device in order to be allowed to terminate an Amazon EC2 instance.

For more information, see [IAM JSON Policy Elements: Condition \(p. 611\)](#) in the IAM Policy Elements Reference.

Monitor Activity in Your AWS Account

You can use logging features in AWS to determine the actions users have taken in your account and the resources that were used. The log files show the time and date of actions, the source IP for an action, which actions failed due to inadequate permissions, and more.

Logging features are available in the following AWS services:

- [Amazon CloudFront](#) – Logs user requests that CloudFront receives. For more information, see [Access Logs](#) in the [Amazon CloudFront Developer Guide](#).
- [AWS CloudTrail](#) – Logs AWS API calls and related events made by or on behalf of an AWS account. For more information, see the [AWS CloudTrail User Guide](#).
- [Amazon CloudWatch](#) – Monitors your AWS Cloud resources and the applications you run on AWS. You can set alarms in CloudWatch based on metrics that you define. For more information, see the [Amazon CloudWatch User Guide](#).
- [AWS Config](#) – Provides detailed historical information about the configuration of your AWS resources, including your IAM users, groups, roles, and policies. For example, you can use AWS Config to determine the permissions that belonged to a user or group at a specific time. For more information, see the [AWS Config Developer Guide](#).
- [Amazon Simple Storage Service \(Amazon S3\)](#) – Logs access requests to your Amazon S3 buckets. For more information, see [Server Access Logging](#) in the [Amazon Simple Storage Service Developer Guide](#).

Video Presentation About IAM Best Practices

The following video includes a conference presentation that covers these best practices and shows additional details about how to work with the features discussed here.

[AWS re:Invent 2015 - IAM Best Practices](#)

Business Use Cases

A simple business use case for IAM can help you understand basic ways you might implement the service to control the AWS access that your users have. The use case is described in general terms, without the mechanics of how you'd use the IAM API to achieve the results you want.

This use case looks at two typical ways a fictional company called Example Corp might use IAM. The first scenario considers Amazon Elastic Compute Cloud (Amazon EC2). The second considers Amazon Simple Storage Service (Amazon S3).

For more information about using IAM with other services from AWS, see [AWS Services That Work with IAM \(p. 586\)](#).

Topics

- [Initial Setup of Example Corp \(p. 68\)](#)
- [Use Case for IAM with Amazon EC2 \(p. 68\)](#)
- [Use Case for IAM with Amazon S3 \(p. 69\)](#)

Initial Setup of Example Corp

John is the founder of Example Corp. Upon starting the company, he creates his own AWS account and uses AWS products by himself. Then he hires employees to work as developers, admins, testers, managers, and system administrators.

John uses the AWS Management Console with the AWS account root user credentials to create a user for himself called *John*, and a group called *Admins*. He gives the *Admins* group permissions to perform all actions on all the AWS account's resources using the AWS managed policy [AdministratorAccess](#). Then he adds the *John* user to the *Admins* group. For a step-by-step guide to creating an Administrators group and an IAM user for yourself, then adding your user to the Administrators group, see [Creating Your First IAM Admin User and Group \(p. 20\)](#).

At this point, John can stop using the root user's credentials to interact with AWS, and instead he begins using only his user credentials.

John also creates a group called *AllUsers* so that he can easily apply any account-wide permissions to all users in the AWS account. He adds himself to the group. He then creates a group called *Developers*, a group called *Testers*, a group called *Managers*, and a group called *SysAdmins*. He creates users for each of his employees, and puts the users in their respective groups. He also adds them all to the *AllUsers* group. For information about creating groups, see [Creating IAM Groups \(p. 169\)](#). For information about creating users, see [Creating an IAM User in Your AWS Account \(p. 86\)](#). For information about adding users to groups, see [Managing IAM Groups \(p. 170\)](#).

Use Case for IAM with Amazon EC2

A company like Example Corp typically uses IAM to interact with services like Amazon EC2. To understand this part of the use case, you need a basic understanding of Amazon EC2. For more information about Amazon EC2, go to the [Amazon EC2 User Guide for Linux Instances](#).

Amazon EC2 Permissions for the Groups

To provide "perimeter" control, John attaches a policy to the *AllUsers* group. This policy denies any AWS request from a user if the originating IP address is outside Example Corp's corporate network.

At Example Corp, different groups require different permissions:

- **System administrators** – Need permission to create and manage AMIs, instances, snapshots, volumes, security groups, and so on. John attaches a policy to the *SysAdmins* group that gives members of the group permission to use all the Amazon EC2 actions.
- **Developers** – Need the ability to work with instances only. John therefore attaches a policy to the *Developers* group that allows developers to call `DescribeInstances`, `RunInstances`, `StopInstances`, `StartInstances`, and `TerminateInstances`.

Note

Amazon EC2 uses SSH keys, Windows passwords, and security groups to control who has access to the operating system of specific Amazon EC2 instances. There's no method in the IAM system to allow or deny access to the operating system of a specific instance.

- **Managers** – Should not be able to perform any Amazon EC2 actions except listing the Amazon EC2 resources currently available. Therefore, John attaches a policy to the *Managers* group that only lets them call Amazon EC2 "Describe" API operations.

For examples of what these policies might look like, see [Example IAM Identity-Based Policies \(p. 394\)](#) and [Using AWS Identity and Access Management](#) in the [Amazon EC2 User Guide for Linux Instances](#).

User's Job Function Change

At some point, one of the developers, Paulo, changes job functions and becomes a manager. John moves Paulo from the Developers group to the Managers group. Now that he's in the Managers group, Paulo's ability to interact with Amazon EC2 instances is limited. He can't launch or start instances. He also can't stop or terminate existing instances, even if he was the user who launched or started the instance. He can list only the instances that Example Corp users have launched.

Use Case for IAM with Amazon S3

Companies like Example Corp would also typically use IAM with Amazon S3. John has created an Amazon S3 bucket for the company called *example_bucket*.

Creation of Other Users and Groups

As employees, Zhang and Mary each need to be able to create their own data in the company's bucket. They also need to read and write shared data that all developers work on. To enable this, John logically arranges the data in *example_bucket* using an Amazon S3 key prefix scheme as shown in the following figure.

```
/example_bucket
  /home
    /zhang
    /mary
  /share
    /developers
    /managers
```

John divides the master */example_bucket* into a set of home directories for each employee, and a shared area for groups of developers and managers.

Now John creates a set of policies to assign permissions to the users and groups:

- **Home directory access for Zhang** – John attaches a policy to Zhang that lets him read, write, and list any objects with the Amazon S3 key prefix */example_bucket/home/zhang/*
- **Home directory access for Mary** – John attaches a policy to Mary that lets her read, write, and list any objects with the Amazon S3 key prefix */example_bucket/home/mary/*
- **Shared directory access for the Developers group** – John attaches a policy to the group that lets developers read, write, and list any objects in */example_bucket/share/developers/*
- **Shared directory access for the Managers group** – John attaches a policy to the group that lets managers read, write, and list objects in */example_bucket/share/managers/*

Note

Amazon S3 doesn't automatically give a user who creates a bucket or object permission to perform other actions on that bucket or object. Therefore, in your IAM policies, you must explicitly give users permission to use the Amazon S3 resources they create.

For examples of what these policies might look like, see [Access Control](#) in the *Amazon Simple Storage Service Developer Guide*. For information on how policies are evaluated at runtime, see [Policy Evaluation Logic \(p. 636\)](#).

User's Job Function Change

At some point, one of the developers, Zhang, changes job functions and becomes a manager. We assume that he no longer needs access to the documents in the *share/developers* directory. John, as an

admin, moves Zhang to the Managers group and out of the Developers group. With just that simple reassignment, Zhang automatically gets all permissions granted to the Managers group, but can no longer access data in the share/developers directory.

Integration with a Third-Party Business

Organizations often work with partner companies, consultants, and contractors. Example Corp has a partner called the Widget Company, and a Widget Company employee named Shirley needs to put data into a bucket for Example Corp's use. John creates a group called *WidgetCo* and a user named Shirley and adds Shirley to the WidgetCo group. John also creates a special bucket called *example_partner_bucket* for Shirley to use.

John updates existing policies or adds new ones to accommodate the partner Widget Company. For example, John can create a new policy that denies members of the WidgetCo group the ability to use any actions other than write. This policy would be necessary only if there's a broad policy that gives all users access to a wide set of Amazon S3 actions.

The IAM Console and Sign-In Page

The AWS Management Console provides a web-based way to administer AWS services. You can sign in to the console and create, list, and perform other tasks with AWS services for your account. These tasks might include starting and stopping Amazon EC2 instances and Amazon RDS databases, creating Amazon DynamoDB tables, creating IAM users, and so on.

When you open the AWS Management Console you might see one of three separate sign-in pages:

- **Main Sign-In Page (p. 72)** – From <https://console.aws.amazon.com/>, enter your root user email address or your IAM user account ID.
- **AWS Account Root User Sign-In Page (p. 72)** – Enter your root user password.
- **IAM User Sign-In Page (p. 73)** – Enter your IAM user name and password on this page.

Sign-in Process

The diagram illustrates the sign-in process for the AWS Management Console. It shows three distinct sign-in paths originating from a central 'Sign in' page.

Root user sign in: This path is indicated by a red arrow pointing down to the left from the central sign-in page. It leads to a separate 'Sign In' form for a root user. This form includes fields for 'Email' (containing 'me@example.com') and 'Password', followed by a 'Sign In' button. Below the button are links for 'Sign in to a different account' and 'Forgot your password?'.

Enter root user email address: This path is indicated by a red arrow pointing down from the central sign-in page to the left. It leads to a 'Sign in' page specifically for entering a root user's email address. The page has a field labeled 'Email address of your AWS account' with placeholder text 'Or to sign in as an IAM user, enter your account ID or account alias instead.' Below the field is a 'Next' button.

Enter IAM user account: This path is indicated by a red arrow pointing down from the central sign-in page to the right. It leads to a 'Sign in' page specifically for IAM users. The page has fields for 'Account ID or alias' (containing '111122223333'), 'IAM user name', and 'Password', followed by a 'Sign In' button. Below the button are links for 'Sign in using root account credentials' and 'Forgot password?'.

If you previously signed in on a different page, your browser might remember this preference. You can use links on the root user and IAM user sign-in pages to return to the main page.

Main Sign-In Page

You can access the main sign-in page by going to <https://console.aws.amazon.com/>.

The main AWS sign-in page includes a single box where you can enter your AWS account root user email address to sign in as the account owner. When you enter the root user email address, you are directed to the root user sign-in page.

Alternatively, you can sign in as an IAM user by entering your account ID or account alias. When you enter account information, you are directed to the IAM user sign-in page.

[Main Sign-in Page](#)

Sign in 

Email address of your AWS account

Or to sign in as an IAM user, enter your [account ID](#) or [account alias](#) instead.

Next

New to AWS?

Create a new AWS account

AWS Account Root User Sign-In Page

When you first create an Amazon Web Services (AWS) account, you begin with a single sign-in identity that has complete access to all AWS services and resources in the account. This identity is called the AWS account *root user* and is accessed by signing in with the email address and password that you used to create the account.

Important

We strongly recommend that you do not use the root user for your everyday tasks, even the administrative ones. Instead, adhere to the [best practice of using the root user only to create your first IAM user \(p. 61\)](#). Then securely lock away the root user credentials and use them to perform only a few account and service management tasks. To view the tasks that require you

to sign in as the root user, see [AWS Tasks That Require Root User](#). For a tutorial on how to set up an administrator for daily use, see [Creating Your First IAM Admin User and Group \(p. 20\)](#).

To access the root user sign-in page, you must enter the root user email address on the main sign-in page. Then, you can enter your password on this page.

Root User Sign-in Page

Root user sign in

Email

me@example.com

Password

Sign In

[Sign in to a different account](#)

[Forgot your password?](#)

To return to the main sign-in page, choose **Sign in to a different account**.

If you forgot your root user password, choose **Forgot your password?** to reset it. For more information about resetting passwords, see [Resetting Your Lost or Forgotten Passwords or Access Keys \(p. 118\)](#).

IAM User Sign-In Page

To access the IAM user sign-in page, you must enter your account ID or alias on the main sign-in page. Then you can enter your IAM user name and password on this page.

IAM Sign-in Page

Account ID or alias

111122223333

IAM user name

Password

Sign In

[Sign-in using root account credentials](#)

[Forgot password?](#)

To return to the main sign-in page, choose **Sign-in using root account credentials**.

If you forgot your IAM password, you can't reset it. Only your IAM administrator can reset your password. For more information about resetting passwords, see [Resetting a Lost or Forgotten Root User Password \(p. 119\)](#).

To use the AWS Management Console, IAM users must provide their account ID or account alias in addition to their user name and password. When you, as an administrator, [create an IAM user in the console \(p. 87\)](#), you must send the sign-in credentials to that user, including the user name and the URL to the account sign-in page.

Important

Depending on how you set up the IAM user, provide all users with a temporary password for their first sign-in and, if appropriate, an MFA device. For detailed information about passwords and MFA devices, see [Managing Passwords \(p. 100\)](#) and [Using Multi-Factor Authentication \(MFA\) in AWS \(p. 119\)](#).

Your unique account sign-in page URL is created automatically when you begin using IAM. You do not have to do anything to use this sign-in page.

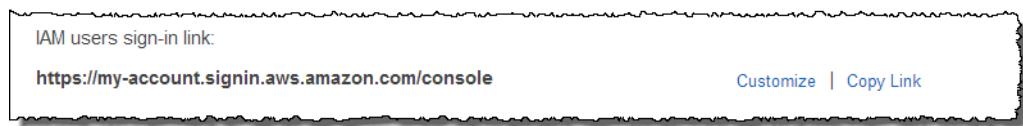
`https://My_AWS_Account_ID.signin.aws.amazon.com/console/`

You can also customize the account sign-in URL for your account if you want the URL to contain your company name (or other friendly identifier) instead of your AWS account ID number. For more information about creating an account alias, see [Your AWS Account ID and Its Alias \(p. 76\)](#).

Tip

To create a bookmark for your account sign-in page in your web browser, you should manually type the sign-in URL for your account in the bookmark entry. Do not use your web browser bookmark feature because redirects can obscure the sign-in URL.

You can find the URL for your account sign-in page anytime by viewing the dashboard of the IAM console.

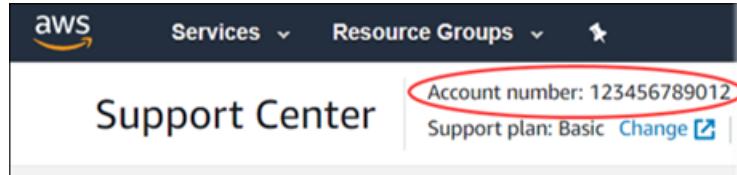


IAM users can also sign in at the following general sign-in endpoint and type an account ID or account alias manually:



Note

To find your AWS account ID number on the AWS Management Console, choose **Support** on the navigation bar on the upper-right, and then choose **Support Center**. Your currently signed-in account number (ID) appears in the **Support Center** title bar.



For convenience, the AWS sign-in page uses a browser cookie to remember the IAM user name and account information. The next time the user goes to any page in the AWS Management Console, the console uses the cookie to redirect the user to the account sign-in page.

Controlling User Access to the AWS Management Console

Users who sign in to your AWS account through the sign-in page can access your AWS resources through the AWS Management Console to the extent that you grant them permission. The following list shows the ways you can grant users access to your AWS account resources through the AWS Management Console. It also shows how users can access other AWS account features through the AWS website.

Note

There is no charge to use IAM.

The AWS Management Console

You create a password for each user who needs access to the AWS Management Console. Users access the console via your IAM-enabled AWS account sign-in page. For information about accessing the sign-in page, see [The IAM Console and Sign-In Page \(p. 71\)](#). For information about creating passwords, see [Managing Passwords \(p. 100\)](#).

Your AWS resources, such as Amazon EC2 instances, Amazon S3 buckets, and so on

Even if your users have passwords, they still need permission to access your AWS resources. When you create a user, that user has no permissions by default. To give your users the permissions they need, you attach policies to them. If you have many users who will be performing the same tasks with the same resources, you can assign those users to a group, then assign the permissions to that group. For information about creating users and groups, see [Identities \(Users, Groups, and Roles\) \(p. 82\)](#). For information about using policies to set permissions, see [Access Management \(p. 355\)](#).

AWS Discussion Forums

Anyone can read the posts on the [AWS Discussion Forums](#). Users who want to post questions or comments to the AWS Discussion Forum can do so using their user name. The first time a user posts to the AWS Discussion Forum, the user is prompted to enter a nickname and email address for use only by that user in the AWS Discussion Forums.

Your AWS account billing and usage information

You can grant users access your AWS account billing and usage information. For more information, see [Controlling Access to Your Billing Information](#) in the *AWS Billing and Cost Management User Guide*.

Your AWS account profile information

Users cannot access your AWS account profile information.

Your AWS account security credentials

Users cannot access your AWS account security credentials.

Note

IAM policies control access regardless of the interface. For example, you could provide a user with a password to access the AWS Management Console, and the policies for that user (or any groups the user belongs to) would control what the user can do in the AWS Management Console. Or, you could provide the user with AWS access keys for making API calls to AWS, and the policies would control which actions the user could call through a library or client that uses those access keys for authentication.

Your AWS Account ID and Its Alias

An account alias substitutes for an account ID in the web address for your account. You can create and manage an account alias from the AWS Management Console, AWS CLI, or AWS API.

Topics

- [Finding Your AWS Account ID \(p. 76\)](#)
- [About Account Aliases \(p. 77\)](#)
- [Creating, Deleting, and Listing an AWS Account Alias \(p. 77\)](#)

Finding Your AWS Account ID

You can find your account ID in the AWS Management Console, or using the AWS CLI or AWS API.

Finding your account ID (Console)

In the navigation bar, choose **Support**, and then **Support Center**. Your currently signed-in 12-digit account number (ID) appears in the **Support Center** title bar.

Finding your account ID (AWS CLI)

To view your user ID, account ID, and your user ARN:

- `aws sts get-caller-identity`

Finding your account ID (AWS API)

To view your user ID, account ID, and your user ARN:

- [GetCallerIdentity](#)

About Account Aliases

If you want the URL for your sign-in page to contain your company name (or other friendly identifier) instead of your AWS account ID, you can create an account alias. This section provides information about AWS account aliases and lists the API operations that you use to create an alias.

Your sign-in page URL has the following format, by default.

```
https://Your_AWS_Account_ID.signin.aws.amazon.com/console/
```

If you create an AWS account alias for your AWS account ID, your sign-in page URL looks like the following example.

```
https://Your_Alias.signin.aws.amazon.com/console/
```

Note

The original URL containing your AWS account ID remains active and can be used after you create your AWS account alias.

Tip

To create a bookmark for your account sign-in page in your web browser, you should manually type the sign-in URL in the bookmark entry. Don't use your web browser's "bookmark this page" feature.

Creating, Deleting, and Listing an AWS Account Alias

You can use the AWS Management Console, the IAM API, or the command line interface to create or delete your AWS account alias.

Important

- Your AWS account can have only one alias. If you create a new alias for your AWS account, the new alias overwrites the previous alias, and the URL containing the previous alias stops working.
- The account alias must be unique across all Amazon Web Services products. It must contain only digits, lowercase letters, and hyphens. For more information on limitations on AWS account entities, see [IAM and STS Limits \(p. 581\)](#).

Creating and Deleting Aliases (Console)

You can create and delete an account alias from the AWS Management Console.

To create or remove an account alias (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Dashboard**.

3. Find the **IAM users sign-in link**, and choose **Customize** to the right of the link.
4. Type the name you want to use for your alias, then choose **Yes, Create**.
5. To remove the alias, choose **Customize**, and then choose **Yes, Delete**. The sign-in URL reverts to using your AWS account ID.

Creating, Deleting, and Listing Aliases (AWS CLI)

To create an alias for your AWS Management Console sign-in page URL, run the following command:

- `aws iam create-account-alias`

To delete an AWS account ID alias, run the following command:

- `aws iam delete-account-alias`

To display your AWS account ID alias, run the following command:

- `aws iam list-account-aliases`

Creating, Deleting, and Listing Aliases (AWS API)

To create an alias for your AWS Management Console sign-in page URL, call the following operation:

- `CreateAccountAlias`

To delete an AWS account ID alias, call the following operation:

- `DeleteAccountAlias`

To display your AWS account ID alias, call the following operation:

- `ListAccountAliases`

Using MFA Devices With Your IAM Sign-in Page

IAM users who are configured with [multi-factor authentication \(MFA\) \(p. 119\)](#) devices must use their MFA devices to sign in to the AWS Management Console. After the user enters the user name and password, AWS checks the user's account to see if MFA is required for that user. The following sections provide information on how users complete signing in when MFA is required.

Topics

- [Signing in with a Virtual MFA Device \(p. 78\)](#)
- [Signing in with a U2F Security Key \(p. 79\)](#)
- [Signing in with a Hardware MFA Device \(p. 79\)](#)

Signing in with a Virtual MFA Device

If MFA is required for the user, a second sign-in page appears. In the **MFA code** box, the user must enter the numeric code provided by the MFA application.

If the MFA code is correct, the user can access the AWS Management Console. If the code is incorrect, the user can try again with another code.

A virtual MFA device can go out of sync. If a user cannot sign in to the AWS Management Console after several tries, the user is prompted to synchronize the virtual MFA device. The user can follow the on-screen prompts to synchronize the virtual MFA device. For information about how you can synchronize a device on behalf of a user in your AWS account, see [Resynchronizing Virtual and Hardware MFA Devices \(p. 138\)](#).

Signing in with a U2F Security Key

If MFA is required for the user, a second sign-in page appears. The user needs to tap the U2F security key.

Unlike other MFA devices, U2F security keys do not go out of sync. Administrators can deactivate a U2F security key if it's lost or broken. For more information, see [Deactivating MFA Devices \(Console\) \(p. 142\)](#).

For information on browsers that support U2F and U2F devices that AWS supports, see [Supported Configurations for Using U2F Security Keys \(p. 129\)](#).

Signing in with a Hardware MFA Device

If MFA is required for the user, a second sign-in page appears. In the **MFA code** box, the user must enter the numeric code provided by a hardware MFA device.

If the MFA code is correct, the user can access the AWS Management Console. If the code is incorrect, the user can try again with another code.

A hardware MFA device can go out of sync. If a user cannot sign in to the AWS Management Console after several tries, the user is prompted to synchronize the MFA token device. The user can follow the on-screen prompts to synchronize the MFA token device. For information about how you can synchronize a device on behalf of a user in your AWS account, see [Resynchronizing Virtual and Hardware MFA Devices \(p. 138\)](#).

IAM Console Search

As you navigate through the IAM Management Console to manage various IAM resources, you often need to locate access keys, Or you might need to browse to the deeply nested IAM resources to find what you need. A faster option is to use the IAM console search page to locate access keys related to your account, IAM entities (such as users, groups, roles, identity providers), policies by name, and more.

The IAM console search feature can locate any of the following:

- IAM entity names that match your search keywords (for users, groups, roles, identity providers, and policies)
- AWS documentation topic names that match your search keywords
- Tasks that match your search keywords

The IAM console search feature does not return information about IAM Access Analyzer.

Every line in the search result is an active link. For example, you can choose the user name in the search result, which takes you to that user's detail page. Or you can choose an action link, for example **Create user**, to go to the [Create User](#) page.

Note

Access key search requires you to type the full access key ID in the search box. The search result shows the user associated with that key. From there you can navigate directly to that user's page, where you can manage their access key.

Using IAM Console Search

Use the **Search** page in the IAM console to find items related to that account.

To search for items in the IAM console

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Search**.
3. In the **Search** box, type your search keywords.
4. Choose a link in the search results list to navigate to the corresponding part of the console or documentation.

Icons in the IAM Console Search Results

The following icons identify the types of items that are found by a search:

Icon	Description
	IAM users
	IAM groups
	IAM roles
	IAM policies
	Tasks such as "create user" or "attach policy"
	Results from the keyword delete
	IAM documentation

Sample Search Phrases

You can use the following phrases in the IAM search. Replace terms in *italics* with the names of actual IAM users, groups, roles, access keys, policies, or identity providers respectively that you want to locate.

- `user_name` or `group_name` or `role_name` or `policy_name` or `identity_provider_name`
- `access_key`
- add user `user_name` to groups or add users to group `group_name`
- remove user `user_name` from groups
- delete `user_name` or delete `group_name` or delete `role_name`, or delete `policy_name`, or delete `identity_provider_name`
- manage access keys `user_name`
- manage signing certificates `user_name`
- users
- manage MFA for `user_name`
- manage password for `user_name`
- create role
- password policy
- edit trust policy for role `role_name`
- show policy document for role `role_name`
- attach policy to `role_name`
- create managed policy
- create user
- create group
- attach policy to `group_name`
- attach entities to `policy_name`
- detach entities to `policy_name`
- what is IAM
- how do I create an IAM user
- how do I use IAM console
- what is a user or what is a group, or what is a policy, or what is a role, or what is an identity provider

Identities (Users, Groups, and Roles)

This section describes *IAM identities*, which you create to provide authentication for people and processes in your AWS account. This section also describes *IAM groups*, which are collections of IAM users that you can manage as a unit. Identities represent the user, and can be authenticated and then authorized to perform actions in AWS. Each of these can be associated with one or more [policies \(p. 355\)](#) to determine what actions a user, role, or member of a group can do with which AWS resources and under what conditions.

The AWS Account Root User (p. 338)

When you first create an Amazon Web Services (AWS) account, you begin with a single sign-in identity that has complete access to all AWS services and resources in the account. This identity is called the AWS account *root user* and is accessed by signing in with the email address and password that you used to create the account.

Important

We strongly recommend that you do not use the root user for your everyday tasks, even the administrative ones. Instead, adhere to the [best practice of using the root user only to create your first IAM user](#). Then securely lock away the root user credentials and use them to perform only a few account and service management tasks. To view the tasks that require you to sign in as the root user, see [AWS Tasks That Require Root User](#).

IAM Users (p. 84)

An IAM [user \(p. 84\)](#) is an entity that you create in AWS. The IAM user represents the person or service who uses the IAM user to interact with AWS. A primary use for IAM users is to give people the ability to sign in to the AWS Management Console for interactive tasks and to make programmatic requests to AWS services using the API or CLI. A user in AWS consists of a name, a password to sign into the AWS Management Console, and up to two access keys that can be used with the API or CLI. When you create an IAM user, you grant it permissions by making it a member of a group that has appropriate permission policies attached (recommended), or by directly attaching policies to the user. You can also clone the permissions of an existing IAM user, which automatically makes the new user a member of the same groups and attaches all the same policies.

IAM Groups (p. 168)

An IAM [group \(p. 168\)](#) is a collection of IAM users. You can use groups to specify permissions for a collection of users, which can make those permissions easier to manage for those users. For example, you could have a group called *Admins* and give that group the types of permissions that administrators typically need. Any user in that group automatically has the permissions that are assigned to the group. If a new user joins your organization and should have administrator privileges, you can assign the appropriate permissions by adding the user to that group. Similarly, if a person changes jobs in your organization, instead of editing that user's permissions, you can remove him or her from the old groups and add him or her to the appropriate new groups. Note that a group is not truly an identity because it cannot be identified as a *Principal* in a [resource-based or trust policy \(p. 379\)](#). It is only a way to attach policies to multiple users at one time.

IAM Roles (p. 175)

An IAM [role \(p. 175\)](#) is very similar to a user, in that it is an identity with permission policies that determine what the identity can and cannot do in AWS. However, a role does not have any credentials (password or access keys) associated with it. Instead of being uniquely associated with one person, a role is intended to be assumable by anyone who needs it. An IAM user can assume a role to temporarily take on different permissions for a specific task. A role can be assigned to a [federated user \(p. 184\)](#) who signs in by using an external identity provider instead of IAM. AWS uses details passed by the identity provider to determine which role is mapped to the federated user.

Temporary Credentials (p. 307)

Temporary credentials are primarily used with IAM roles, but there are also other uses. You can request temporary credentials that have a more restricted set of permissions than your standard IAM user. This prevents you from accidentally performing tasks that are not permitted by the more restricted credentials. A benefit of temporary credentials is that they expire automatically after a set period of time. You have control over the duration that the credentials are valid.

When to Create an IAM User (Instead of a Role)

Because an IAM user is just an identity with specific permissions in your account, you might not need to create an IAM user for every occasion on which you need credentials. In many cases, you can take advantage of IAM *roles* and their temporary security credentials instead of using the long-term credentials associated with an IAM user.

- **You created an AWS account and you're the only person who works in your account.**

It's possible to work with AWS using the root user credentials for your AWS account, but we don't recommend it. Instead, we strongly recommend that you create an IAM user for yourself and use the credentials for that user when you work with AWS. For more information, see [IAM Best Practices \(p. 60\)](#).

- **Other people in your group need to work in your AWS account, and your group is using no other identity mechanism.**

Create IAM users for the individuals who need access to your AWS resources, assign appropriate permissions to each user, and give each user his or her own credentials. We strongly recommend that you never share credentials among multiple users.

When to Create an IAM Role (Instead of a User)

Create an IAM role in the following situations:

You're creating an application that runs on an Amazon Elastic Compute Cloud (Amazon EC2) instance and that application makes requests to AWS.

Don't create an IAM user and pass the user's credentials to the application or embed the credentials in the application. Instead, create an IAM role that you attach to the EC2 instance to give temporary security credentials to applications running on the instance. When an application uses these credentials in AWS, it can perform all of the operations that are allowed by the policies attached to the role. For details, see [Using an IAM Role to Grant Permissions to Applications Running on Amazon EC2 Instances \(p. 269\)](#).

You're creating an app that runs on a mobile phone and that makes requests to AWS.

Don't create an IAM user and distribute the user's access key with the app. Instead, use an identity provider like Login with Amazon, Amazon Cognito, Facebook, or Google to authenticate users and map the users to an IAM role. The app can use the role to get temporary security credentials that have the permissions specified by the policies attached to the role. For more information, see the following:

- [Amazon Cognito Overview in the AWS Mobile SDK for Android Developer Guide](#)
- [Amazon Cognito Overview in the AWS Mobile SDK for iOS Developer Guide](#)
- [About Web Identity Federation \(p. 185\)](#)

Users in your company are authenticated in your corporate network and want to be able to use AWS without having to sign in again—that is, you want to allow users to federate into AWS.

Don't create IAM users. Configure a federation relationship between your enterprise identity system and AWS. You can do this in two ways:

- If your company's identity system is compatible with SAML 2.0, you can establish trust between your company's identity system and AWS. For more information, see [About SAML 2.0-based Federation \(p. 190\)](#).
- Create and use a custom proxy server that translates user identities from the enterprise into IAM roles that provide temporary AWS security credentials. For more information, see [Enabling Custom Identity Broker Access to the AWS Console \(p. 213\)](#).

IAM Users

An AWS Identity and Access Management (IAM) *user* is an entity that you create in AWS to represent the person or application that uses it to interact with AWS. A user in AWS consists of a name and credentials.

An IAM user with administrator permissions is not the same thing as the AWS account root user. For more information about the root user, see [The AWS Account Root User \(p. 338\)](#).

Important

If you arrived at this page while trying to enable Amazon Advertising for your application or web site, see [Sign up for the Product Advertising API](#).

How AWS identifies an IAM user

When you create a user, IAM creates these ways to identify that user:

- A "friendly name" for the user, which is the name that you specified when you created the user, such as Richard or Anaya. These are the names you see in the AWS Management Console.
- An Amazon Resource Name (ARN) for the user. You use the ARN when you need to uniquely identify the user across all of AWS. For example, you could use an ARN to specify the user as a Principal in an IAM policy for an Amazon S3 bucket. An ARN for an IAM user might look like the following:

`arn:aws:iam::account-ID-without-hyphens:user/Richard`

- A unique identifier for the user. This ID is returned only when you use the API, Tools for Windows PowerShell, or AWS CLI to create the user; you do not see this ID in the console.

For more information about these identifiers, see [IAM Identifiers \(p. 575\)](#).

Users and credentials

You can access AWS in different ways depending on the user credentials:

- **Console password (p. 100):** A password that the user can type to sign in to interactive sessions such as the AWS Management Console.
- **Access keys (p. 110):** A combination of an access key ID and a secret access key. You can assign two to a user at a time. These can be used to make programmatic calls to AWS. For example, you might use access keys when using the API for code or at a command prompt when using the AWS CLI or the AWS PowerShell tools.
- **SSH keys for use with CodeCommit (p. 161):** An SSH public key in the OpenSSH format that can be used to authenticate with CodeCommit.
- **Server certificates (p. 163):** SSL/TLS certificates that you can use to authenticate with some AWS services. We recommend that you use AWS Certificate Manager (ACM) to provision, manage, and deploy your server certificates. Use IAM only when you must support HTTPS connections in a region that is not supported by ACM. To learn which regions support ACM, see [AWS Certificate Manager Regions and Endpoints](#) in the *AWS General Reference*.

You can choose the credentials that are right for your IAM user. When you use the AWS Management Console to create a user, you must choose to at least include a console password or access keys. By default, a brand new IAM user created using the AWS CLI or AWS API has no credentials of any kind. You must create the type of credentials for an IAM user based on the needs of your user.

Take advantage of the following options to administer passwords, access keys, and MFA devices:

- **Manage passwords for your IAM users (p. 100).** Create and change the passwords that permit access to the AWS Management Console. Set a password policy to enforce a minimum password complexity. Allow users to change their own passwords.
- **Manage access keys for your IAM users (p. 110).** Create and update access keys for programmatic access to the resources in your account.
- You can enhance the security of the user's credentials by enabling [multi-factor authentication \(MFA\) \(p. 119\)](#) for the user. With MFA, users have to provide two forms of identification: First, they provide the credentials that are part of their user identity (a password or access key). In addition, they provide a temporary numeric code that's generated on a hardware device or by an application on a smartphone or tablet, or sent by AWS to an SMS-compatible mobile device.
- **Find unused passwords and access keys (p. 154).** Anyone who has a password or access keys for your account or an IAM user in your account has access to your AWS resources. The security [best practice](#) is to remove passwords and access keys when users no longer need them.
- **Download a credential report for your account (p. 156).** You can generate and download a credential report that lists all IAM users in your account and the status of their various credentials, including passwords, access keys, and MFA devices. For passwords and access keys, the credential report shows how recently the password or access key has been used.

Users and permissions

By default, a brand new IAM user has no [permissions \(p. 355\)](#) to do anything. The user is not authorized to perform any AWS operations or to access any AWS resources. An advantage of having individual IAM users is that you can assign permissions individually to each user. You might assign administrative permissions to a few users, who then can administer your AWS resources and can even create and manage other IAM users. In most cases, however, you want to limit a user's permissions to just the tasks (AWS actions or operations) and resources that are needed for the job.

Imagine a user named Diego. When you create the IAM user Diego, you can create a password for that user. You also attach permissions to the IAM user that let him launch a specific Amazon EC2 instance and read (GET) information from a table in an Amazon RDS database. For procedures on how to create users and grant them initial credentials and permissions, see [Creating an IAM User in Your AWS Account \(p. 86\)](#). For procedures on how to change the permissions for existing users, see [Changing](#)

[Permissions for an IAM User \(p. 95\)](#). For procedures on how to change the user's password or access keys, see [Managing Passwords \(p. 100\)](#) and [Managing Access Keys for IAM Users \(p. 110\)](#).

You can also add a permissions boundary to your users. A permissions boundary is an advanced feature that allows you to use AWS managed policies to limit the maximum permissions that an identity-based policy can grant to a user or role. For more information about policy types and uses, see [Policies and Permissions \(p. 356\)](#).

Users and accounts

Each IAM user is associated with one and only one AWS account. Because users are defined within your AWS account, they don't need to have a payment method on file with AWS. Any AWS activity performed by users in your account is billed to your account.

There's a limit to the number of IAM users you can have in an AWS account. For more information, see [IAM and STS Limits \(p. 581\)](#).

Users as service accounts

An IAM user is a resource in IAM that has associated credentials and permissions. An IAM user can represent a person or an application that uses its credentials to make AWS requests. This is typically referred to as a *service account*. If you choose to use the long-term credentials of an IAM user in your application, **do not embed access keys directly into your application code**. The AWS SDKs and the AWS Command Line Interface allow you to put access keys in known locations so that you do not have to keep them in code. For more information, see [Manage IAM User Access Keys Properly](#) in the *AWS General Reference*. Alternatively, and as a best practice, you can [use temporary security credentials \(IAM roles\) instead of long-term access keys](#).

Creating an IAM User in Your AWS Account



[Follow us on Twitter](#)

You can create one or more IAM users in your AWS account. You might create an IAM user when someone joins your team, or when you create a new application that needs to make API calls to AWS.

Important

If you arrived at this page while trying to enable Amazon Advertising for your application or website, see [Becoming a Product Advertising API Developer](#).

If you arrived at this page from the IAM console, it is possible that your account does not include IAM users, even though you are signed in. You could be signed in as the AWS account root user, using a role, or signed in with temporary credentials. To learn more about these IAM identities, see [Identities \(Users, Groups, and Roles\) \(p. 82\)](#).

Topics

- [Creating IAM Users \(Console\) \(p. 87\)](#)
- [Creating IAM Users \(AWS CLI\) \(p. 89\)](#)
- [Creating IAM Users \(AWS API\) \(p. 90\)](#)

The process of creating a user and enabling that user to perform work tasks consists of the following steps:

1. Create the user in the AWS Management Console, the AWS CLI, Tools for Windows PowerShell, or using an AWS API operation. If you create the user in the AWS Management Console, then steps 1–4 are handled automatically, based on your choices. If you create the users programmatically, then you must perform each of those steps individually.
2. Create credentials for the user, depending on the type of access the user requires:

- **Programmatic access:** The IAM user might need to make API calls, use the AWS CLI, or use the Tools for Windows PowerShell. In that case, create an access key (access key ID and a secret access key) for that user.
- **AWS Management Console access:** If the user needs to access the AWS Management Console, [create a password for the user \(p. 104\)](#).

As a best practice, create only the credentials that the user needs. For example, for a user who requires access only through the AWS Management Console, do not create access keys.

3. Give the user permissions to perform the required tasks by adding the user to one or more groups. You can also grant permissions by attaching permission policies directly to the user. However, we recommend instead that you put your users in groups and manage permissions through policies that are attached to those groups. You can also use a [permissions boundary \(p. 370\)](#) to limit the permissions that a user can have, though this is not common.
4. (Optional) Add metadata to the user by attaching tags. For more information about using tags in IAM, see [Tagging IAM Users and Roles \(p. 295\)](#).
5. Provide the user with the necessary sign-in information. This includes the password and the console URL for the account sign-in page where the user provides those credentials. For more information, see [How IAM Users Sign In to AWS \(p. 90\)](#).
6. (Optional) Configure [multi-factor authentication \(MFA\) \(p. 119\)](#) for the user. MFA requires the user to provide a one-time-use code each time he or she signs into the AWS Management Console.
7. (Optional) Give users permissions to manage their own security credentials. (By default, users do not have permissions to manage their own credentials.) For more information, see [Permitting IAM Users to Change Their Own Passwords \(p. 108\)](#).

For information about the permissions that you need in order to create a user, see [Permissions Required to Access IAM Resources \(p. 517\)](#).

Creating IAM Users (Console)

You can use the AWS Management Console to create IAM users.

To create one or more IAM users (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Users** and then choose **Add user**.
3. Type the user name for the new user. This is the sign-in name for AWS. If you want to add more than one user at the same time, choose **Add another user** for each additional user and type their user names. You can add up to 10 users at one time.

Note

User names can be a combination of up to 64 letters, digits, and these characters: plus (+), equal (=), comma (,), period (.), at sign (@), underscore (_), and hyphen (-). Names must be unique within an account. They are not distinguished by case. For example, you cannot create two users named *TESTUSER* and *testuser*. For more information about limitations on IAM entities, see [IAM and STS Limits \(p. 581\)](#).

4. Select the type of access this set of users will have. You can select programmatic access, access to the AWS Management Console, or both.
 - Select **Programmatic access** if the users require access to the API, AWS CLI, or Tools for Windows PowerShell. This creates an access key for each new user. You can view or download the access keys when you get to the **Final** page.
 - Select **AWS Management Console access** if the users require access to the AWS Management Console. This creates a password for each new user.

- a. For **Console password**, choose one of the following:
 - **Autogenerated password.** Each user gets a randomly generated password that meets the account password policy in effect (if any). You can view or download the passwords when you get to the **Final** page.
 - **Custom password.** Each user is assigned the password that you type in the box.
- b. (Optional) We recommend that you select **Require password reset** to ensure that users are forced to change their password the first time they sign in.

Note

If you have *not* enabled the account-wide password policy setting **Allow users to change their own password**, then selecting **Require password reset** automatically attaches an AWS managed policy named **IAMUserChangePassword** to the new users that grants them permission to change their own passwords.

5. Choose **Next: Permissions**.
6. On the **Set permissions** page, specify how you want to assign permissions to this set of new users. Choose one of the following three options:
 - **Add user to group.** Choose this option if you want to assign the users to one or more groups that already have permissions policies. IAM displays a list of the groups in your account, along with their attached policies. You can select one or more existing groups, or choose **Create group** to create a new group. For more information, see [Changing Permissions for an IAM User \(p. 95\)](#).
 - **Copy permissions from existing user.** Choose this option to copy all of the group memberships, attached managed policies, embedded inline policies, and any existing [permissions boundaries \(p. 370\)](#) from an existing user to the new users. IAM displays a list of the users in your account. Select the one whose permissions most closely match the needs of your new users.
 - **Attach existing policies to user directly.** Choose this option to see a list of the AWS managed and customer managed policies in your account. Select the policies that you want to attach to the new users or choose **Create policy** to open a new browser tab and create a new policy from scratch. For more information, see step 4 in the procedure [Creating IAM Policies \(Console\) \(p. 443\)](#). After you create the policy, close that tab and return to your original tab to add the policy to the new user. As a [best practice \(p. 61\)](#), we recommend that you instead attach your policies to a group and then make users members of the appropriate groups.
7. (Optional) Set a [permissions boundary \(p. 370\)](#). This is an advanced feature.

Open the **Set permissions boundary** section and choose **Use a permissions boundary to control the maximum user permissions**. IAM displays a list of the AWS managed and customer managed policies in your account. Select the policy to use for the permissions boundary or choose **Create policy** to open a new browser tab and create a new policy from scratch. For more information, see step 4 in the procedure [Creating IAM Policies \(Console\) \(p. 443\)](#). After you create the policy, close that tab and return to your original tab to select the policy to use for the permissions boundary.

8. Choose **Next: Tags**.
9. (Optional) Add metadata to the user by attaching tags as key-value pairs. For more information about using tags in IAM, see [Tagging IAM Users and Roles \(p. 295\)](#).
10. Choose **Next: Review** to see all of the choices you made up to this point. When you are ready to proceed, choose **Create user**.
11. To view the users' access keys (access key IDs and secret access keys), choose **Show** next to each password and access key that you want to see. To save the access keys, choose **Download .csv** and then save the file to a safe location.

Important

This is your only opportunity to view or download the secret access keys, and you must provide this information to your users before they can use the AWS API. Save the user's new

access key ID and secret access key in a safe and secure place. **You will not have access to the secret keys again after this step.**

12. Provide each user with his or her credentials. On the final page you can choose **Send email** next to each user. Your local mail client opens with a draft that you can customize and send. The email template includes the following details to each user:

- User name
- URL to the account sign-in page. Use the following example, substituting the correct account ID number or account alias:

```
https://AWS-account-ID or alias.signin.aws.amazon.com/console
```

For more information, see [How IAM Users Sign In to AWS \(p. 90\)](#).

Important

The user's password is **not** included in the generated email. You must provide them to the customer in a way that complies with your organization's security guidelines.

Creating IAM Users (AWS CLI)

You can use the AWS CLI to create an IAM user.

To create an IAM user (AWS CLI)

1. Create a user.
 - [aws iam create-user](#)
2. (Optional) Give the user access to the AWS Management Console. This requires a password. You must also give the user the [URL of your account's sign-in page. \(p. 90\)](#)
 - [aws iam create-login-profile](#)
3. (Optional) Give the user programmatic access. This requires access keys.
 - [aws iam create-access-key](#)
 - Tools for Windows PowerShell: [New-IAMAccessKey](#)
 - IAM API: [CreateAccessKey](#)

Important

This is your only opportunity to view or download the secret access keys, and you must provide this information to your users before they can use the AWS API. Save the user's new access key ID and secret access key in a safe and secure place. **You will not have access to the secret keys again after this step.**

4. Add the user to one or more groups. The groups that you specify should have attached policies that grant the appropriate permissions for the user.
 - [aws iam add-user-to-group](#)
5. (Optional) Attach a policy to the user that defines the user's permissions. **Note:** We recommend that you manage user permissions by adding the user to a group and attaching a policy to the group instead of attaching directly to a user.
 - [aws iam attach-user-policy](#)
6. (Optional) Add custom attributes to the user by attaching tags. For more information, see [Managing Tags on IAM Entities \(AWS CLI or AWS API\) \(p. 298\)](#).

7. (Optional) Give the user permission to manage their own security credentials. For more information, see [AWS: Allows MFA-Authenticated IAM Users to Manage Their Own Credentials on the My Security Credentials Page \(p. 398\)](#).

Creating IAM Users (AWS API)

You can use the AWS API to create an IAM user.

To create an IAM user from the (AWS API)

1. Create a user.
 - [CreateUser](#)
2. (Optional) Give the user access to the AWS Management Console. This requires a password. You must also give the user the [URL of your account's sign-in page. \(p. 90\)](#)
 - [CreateLoginProfile](#)
3. (Optional) Give the user programmatic access. This requires access keys.
 - [CreateAccessKey](#)
4. Add the user to one or more groups. The groups that you specify should have attached policies that grant the appropriate permissions for the user.
 - [AddUserToGroup](#)
5. (Optional) Attach a policy to the user that defines the user's permissions. **Note:** We recommend that you manage user permissions by adding the user to a group and attaching a policy to the group instead of attaching directly to a user.
 - [AttachUserPolicy](#)
6. (Optional) Add custom attributes to the user by attaching tags. For more information, see [Managing Tags on IAM Entities \(AWS CLI or AWS API\) \(p. 298\)](#).
7. (Optional) Give the user permission to manage their own security credentials. For more information, see [AWS: Allows MFA-Authenticated IAM Users to Manage Their Own Credentials on the My Security Credentials Page \(p. 398\)](#).

How IAM Users Sign In to AWS

To sign in to the AWS Management Console as an IAM user, you must provide your account ID or account alias in addition to your user name and password. When your administrator [created your IAM user in the console \(p. 87\)](#), they should have sent you your sign-in credentials, including your user name and the URL to your account sign-in page that includes your account ID or account alias.

`https://My_AWS_Account_ID.signin.aws.amazon.com/console/`

Tip

To create a bookmark for your account sign-in page in your web browser, you should manually type the sign-in URL for your account in the bookmark entry. Do not use your web browser bookmark feature because redirects can obscure the sign-in URL.

You can also sign in at the following general sign-in endpoint and type your account ID or account alias manually:

```
https://console.aws.amazon.com/
```

For convenience, the AWS sign-in page uses a browser cookie to remember the IAM user name and account information. The next time the user goes to any page in the AWS Management Console, the console uses the cookie to redirect the user to the account sign-in page.

You have access only to the AWS resources that your administrator specifies in the policy that is attached to your IAM user identity. To work in the console, you must have permissions to perform the actions that the console performs, such as listing and creating AWS resources. For more information, see [Access Management \(p. 355\)](#) and [Example IAM Identity-Based Policies \(p. 394\)](#).

Note

If your organization has an existing identity system, you might want to create a single sign-on (SSO) option. SSO gives users access to the AWS Management Console for your account without requiring them to have an IAM user identity. SSO also eliminates the need for users to sign in to your organization's site and to AWS separately. For more information, see [Enabling Custom Identity Broker Access to the AWS Console \(p. 213\)](#).

Logging sign-in details in CloudTrail

If you enable CloudTrail to log sign-in events to your logs, you need to be aware of how CloudTrail chooses where to log the events.

- If your users sign-in directly to a console, they are redirected to either a global or a regional sign-in endpoint, based on whether the selected service console supports regions. For example, the main console home page supports regions, so if you sign in to the following URL:

```
https://alias.signin.aws.amazon.com/console
```

you are redirected to a regional sign-in endpoint such as `https://us-east-2.signin.aws.amazon.com`, resulting in a regional CloudTrail log entry in the user's region's log:

On the other hand, the Amazon S3 console does not support regions, so if you sign in to the following URL

```
https://alias.signin.aws.amazon.com/console/s3
```

AWS redirects you to the global sign-in endpoint at `https://signin.aws.amazon.com`, resulting in a global CloudTrail log entry.

- You can manually request a certain regional sign-in endpoint by signing in to the region-enabled main console home page using a URL syntax like the following:

```
https://alias.signin.aws.amazon.com/console?region=ap-southeast-1
```

AWS redirects you to the `ap-southeast-1` regional sign-in endpoint and results in a regional CloudTrail log event.

For more information about CloudTrail and IAM, see [Logging IAM Events with AWS CloudTrail](#).

If users need programmatic access to work with your account, you can create an access key pair (an access key ID and a secret access key) for each user, as described in [Managing Access Keys \(Console\) \(p. 112\)](#).

Managing IAM Users

Amazon Web Services offers multiple tools for managing the IAM users in your AWS account. You can list the IAM users in your account or in a group, or list all groups that a user is a member of. You can rename or change the path of an IAM user. You can also delete an IAM user from your AWS account.

For more information about adding, changing, or removing managed policies for an IAM user, see [Changing Permissions for an IAM User \(p. 95\)](#). For information about managing inline policies for IAM users, see [Adding and Removing IAM Identity Permissions \(p. 457\)](#), [Editing IAM Policies \(p. 468\)](#), and [Deleting IAM Policies \(p. 472\)](#). As a best practice, use managed policies instead of inline policies.

For information about managing IAM user passwords, see [Managing Passwords for IAM Users \(p. 104\)](#).

Topics

- [View User Access \(p. 92\)](#)
- [Listing IAM Users \(p. 92\)](#)
- [Renaming an IAM User \(p. 93\)](#)
- [Deleting an IAM User \(p. 93\)](#)

View User Access

Before you delete a user, you should review its recent service-level activity. This is important because you don't want to remove access from a principal (person or application) who is using it. For more information about viewing service last accessed data, see [Refining Permissions Using Service Last Accessed Data \(p. 475\)](#).

Listing IAM Users

You can list the IAM users in your AWS account or in a specific IAM group, and list all the groups that a user is in. For information about the permissions that you need in order to list users, see [Permissions Required to Access IAM Resources \(p. 517\)](#).

To list all the users in the account

- [AWS Management Console](#): In the navigation pane, choose **Users**. The console displays the users in your AWS account.
- [AWS CLI](#): `aws iam list-users`
- [AWS API](#): `ListUsers`

To list the users in a specific group

- [AWS Management Console](#): In the navigation pane, choose **Groups**, choose the name of the group, and then choose the **Users** tab.
- [AWS CLI](#): `aws iam get-group`
- [AWS API](#): `GetGroup`

To list all the groups that a user is in

- [AWS Management Console](#): In the navigation pane, choose **Users**, choose the user name, and then choose the **Groups** tab.
- [AWS CLI](#): `aws iam list-groups-for-user`

- AWS API: [ListGroupForUser](#)

Renaming an IAM User

To change a user's name or path, you must use the AWS CLI, Tools for Windows PowerShell, or AWS API. There is no option in the console to rename a user. For information about the permissions that you need in order to rename a user, see [Permissions Required to Access IAM Resources \(p. 517\)](#).

When you change a user's name or path, the following happens:

- Any policies attached to the user stay with the user under the new name.
- The user stays in the same groups under the new name.
- The unique ID for the user remains the same. For more information about unique IDs, see [Unique Identifiers \(p. 579\)](#).
- Any resource or role policies that refer to the user *as a principal* (the user is being granted access) are automatically updated to use the new name or path. For example, any queue-based policies in Amazon SQS or resource-based policies in Amazon S3 are automatically updated to use the new name and path.

IAM does not automatically update policies that refer to the user *as a resource* to use the new name or path; you must manually do that. For example, imagine that user Richard has a policy attached to him that lets him manage his security credentials. If an administrator renames Richard to Rich, the administrator also needs to update that policy to change the resource from this:

```
arn:aws:iam::111122223333:user/division_abc/subdivision_xyz/Richard
```

to this:

```
arn:aws:iam::111122223333:user/division_abc/subdivision_xyz/Rich
```

This is true also if an administrator changes the path; the administrator needs to update the policy to reflect the new path for the user.

To rename a user

- AWS CLI: [aws iam update-user](#)
- AWS API: [UpdateUser](#)

Deleting an IAM User

You might delete an IAM user from your account if someone quits your company. If the user is only temporarily away from your company, you can disable the user's credentials instead of deleting the user entirely from the AWS account. That way, you can prevent the user from accessing the AWS account's resources during the absence but you can re-enable the user later.

For more information about disabling credentials, see [Managing Access Keys for IAM Users \(p. 110\)](#). For information about the permissions that you need in order to delete a user, see [Permissions Required to Access IAM Resources \(p. 517\)](#).

Topics

- [Deleting an IAM User \(Console\) \(p. 94\)](#)
- [Deleting an IAM User \(AWS CLI\) \(p. 94\)](#)

Deleting an IAM User (Console)

When you use the AWS Management Console to delete an IAM user, IAM automatically deletes the following information for you:

- The user
- Any group memberships—that is, the user is removed from any IAM groups that the user was a member of
- Any password associated with the user
- Any access keys belonging to the user
- All inline policies embedded in the user (policies that are applied to a user via group permissions are not affected)

Note

Any managed policies attached to the user are detached from the user when the user is deleted. Managed policies are not deleted when you delete a user.

- Any associated MFA device

To delete an IAM user (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Users**, and then select the check box next to the user name that you want to delete, not the name or row itself.
3. At the top of the page, choose **Delete user**.
4. In the confirmation dialog box, wait for the service last accessed data to load before you review the data. The dialog box shows when each of the selected users last accessed an AWS service. If you attempt to delete a user that has been active within the last 30 days, you must select an additional check box to confirm that you want to delete the active user. If you want to proceed, choose **Yes, Delete**.

Deleting an IAM User (AWS CLI)

Unlike the AWS Management Console, when you delete a user with the AWS CLI, you must delete the items attached to the user manually. This procedure illustrates the process.

To delete a user from your account (AWS CLI)

1. Delete the user's password, if the user has one.

```
aws iam delete-login-profile
```

2. Delete the user's access keys, if the user has them.

```
aws iam list-access-keys
```

 (to list the user's access keys) and

```
aws iam delete-access-key
```

3. Delete the user's signing certificate. Note that when you delete a security credential, it's gone forever and can't be retrieved.

```
aws iam list-signing-certificates
```

 (to list the user's signing certificates) and

```
aws iam delete-signing-certificate
```

4. Delete the user's SSH public key, if the user has them.

```
aws iam list-ssh-public-keys
```

 (to list the user's SSH public keys) and

```
aws iam delete-ssh-public-key
```

5. Delete the user's Git credentials.

-
-
-
-
-
6. Deactivate the user's multi-factor authentication (MFA) device, if the user has one.
`aws iam list-mfa-devices` (to list the user's MFA devices), `aws iam deactivate-mfa-device` (to deactivate the device), and `aws iam delete-virtual-mfa-device` (to permanently delete a virtual MFA device)
7. Delete the user's inline policies.
`aws iam list-user-policies` (to list the inline policies for the user) and `aws iam delete-user-policy` (to delete the policy)
8. Detach any managed policies that are attached to the user.
`aws iam list-attached-user-policies` (to list the managed policies attached to the user) and `aws iam detach-user-policy` (to detach the policy)
9. Remove the user from any groups.
`aws iam list-groups-for-user` (to list the groups to which the user belongs) and `aws iam remove-user-from-group`
10. Delete the user.
`aws iam delete-user`

Changing Permissions for an IAM User

You can change the permissions for an IAM user in your AWS account by changing its group memberships, by copying permissions from an existing user, by attaching policies directly to a user, or by setting a [permissions boundary \(p. 370\)](#). A permissions boundary controls the maximum permissions that a user can have. Permissions boundaries are an advanced AWS feature.

For information about the permissions that you need in order to modify the permissions for a user, see [Permissions Required to Access IAM Resources \(p. 517\)](#).

Topics

- [View User Access \(p. 95\)](#)
- [Adding Permissions to a User \(Console\) \(p. 95\)](#)
- [Changing Permissions for a User \(Console\) \(p. 98\)](#)
- [Removing a Permissions Policy from a User \(Console\) \(p. 99\)](#)
- [Removing the Permissions Boundary from a User \(Console\) \(p. 99\)](#)
- [Adding and Removing a User's Permissions \(AWS CLI or AWS API\) \(p. 99\)](#)

View User Access

Before you change the permissions for a user, you should review its recent service-level activity. This is important because you don't want to remove access from a principal (person or application) who is using it. For more information about viewing service last accessed data, see [Refining Permissions Using Service Last Accessed Data \(p. 475\)](#).

Adding Permissions to a User (Console)

IAM offers three ways to add permissions policies to a user:

- **Add user to group** – Make the user a member of a group. The policies from the group are attached to the user.
- **Copy permissions from existing user** – Copy all group memberships, attached managed policies, inline policies, and any existing permissions boundaries from the source user.
- **Attach policies directly to user** – Attach a managed policy directly to the user. As a [best practice \(p. 61\)](#), we recommend that you instead attach your policies to a group and then make users members of the appropriate groups.

Important

If the user has a permissions boundary, then you cannot add more permissions to a user than are allowed by the permissions boundary.

Adding Permissions by Adding the User to a Group

Adding a user to a group affects the user immediately.

To add permissions to a user by adding the user to a group

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Users**.
3. Review the current group memberships for users in the **Groups** column of the console. If necessary, add the column to the users table by completing the following steps:
 1. Above the table on the far right, choose the settings symbol ().
 2. In the **Manage Columns** dialog box, select the **Groups** column. Optionally, you can also clear the check box for any column headings that you do not want to appear in the users table.
 3. Choose **Close** to return to the list of users.
4. Choose the name of the user whose permissions you want to modify.
5. Choose the **Permissions** tab, and then choose **Add permissions**. Choose **Add user to group**.
6. Select the check box for each group that you want the user to join. The list shows each group's name and the policies that the user receives if made a member of that group.
7. (Optional) In addition to selecting from existing groups, you can choose **Create group** to define a new group:
 - a. In the new tab, for **Group name**, type a name for your new group.

Note

Group names can be a combination of up to 128 letters, digits, and these characters: plus (+), equal (=), comma (,), period (.), at sign (@), and hyphen (-). Names must be unique within an account. They are not distinguished by case. For example, you cannot create two groups named *TESTGROUP* and *testgroup*. For more information about limitations on IAM entities, see [IAM and STS Limits \(p. 581\)](#).

- b. Select one or more check boxes for the managed policies that you want to attach to the group. You can also create a new managed policy by choosing **Create policy**. If you do, return to this

browser tab or window when the new policy is done; choose **Refresh**; and then choose the new policy to attach it to your group. For more information, see [Creating IAM Policies \(p. 443\)](#).

- c. Choose **Create group**.
- d. Return to the original tab, refresh your list of groups. Then select the check box for your new group.
8. Choose **Next: Review** to see the list of group memberships to be added to the user. Then choose **Add permissions**.

Adding Permissions by Copying from Another User

Copying permissions affects the user immediately.

To add permissions to a user by copying permissions from another user

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. Choose **Users** in the navigation pane, choose the name of the user whose permissions you want to modify, and then choose the **Permissions** tab.
3. Choose **Add permissions**, and then choose **Copy permissions from existing user**. The list displays available users along with their group memberships and attached policies. If the full list of groups or policies don't fit on one line, you can choose the link for **and n more**. Doing that opens a new browser tab and see the full list of policies (**Permissions** tab) and groups (**Groups** tab).
4. Select the radio button next to the user whose permissions you want to copy.
5. Choose **Next: Review** to see the list of changes that are to be made to the user. Then choose **Add permissions**.

Adding Permissions by Attaching Policies Directly to the User

Attaching policies affects the user immediately.

To add permissions to a user by directly attaching managed policies

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. Choose **Users** in the navigation pane, choose the name of the user whose permissions you want to modify, and then choose the **Permissions** tab.
3. Choose **Add permissions**, and then choose **Attach existing policies directly to user**.
4. Select one or more check boxes for the managed policies that you want to attach to the user. You can also create a new managed policy by choosing **Create policy**. If you do, return to this browser tab or window when the new policy is done. Choose **Refresh**; and then select the check box for the new policy to attach it to your user. For more information, see [Creating IAM Policies \(p. 443\)](#).
5. Choose **Next: Review** to see the list of policies that are to be attached to the user. Then choose **Add permissions**.

Setting the Permissions Boundary for a User

Setting a permissions boundary affects the user immediately.

To set the permissions boundary for a user

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.

2. In the navigation pane, choose **Users**.
3. Choose the name of the user whose permissions boundary you want to change.
4. Choose the **Permissions** tab. If necessary, open the **Permissions boundary** section and then choose **Set boundary**.
5. Select the policy that you want to use for the permissions boundary.
6. Choose **Set boundary**.

Changing Permissions for a User (Console)

IAM offers three ways to change permissions that are associated with a user:

- **Edit a permissions policy** – Edit a user's inline policy, the inline policy of the user's group, or edit a managed policy that is attached to the user directly or from a group. If the user has a permissions boundary, then you cannot provide more permissions than are allowed by the policy that was used as the user's permissions boundary.
- **Changing the permissions boundary** – Change the policy that is used as the permissions boundary for the user. This can expand or restrict the maximum permissions that a user can have.

Editing a Permissions Policy Attached to a User

Changing permissions affects the user immediately.

To edit a user's attached managed policies

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Users**.
3. Choose the name of the user whose permissions policy you want to change.
4. Choose the **Permissions** tab. If necessary, open the **Permissions policies** section.
5. Choose the name of the policy that you want to edit to view details about the policy. Choose the **Used as** tab to view other entities that might be affected if you edit the policy.
6. Choose the **Permissions tab** and review the permissions granted by the policy. Then choose **Edit policy**.
7. Edit the policy using the **Visual editor** tab or the **JSON** tab. For more information, see [Editing IAM Policies \(p. 468\)](#).
8. Choose **Review policy**, review the policy summary, and then choose **Save changes**.

Changing the Permissions Boundary for a User

Changing a permissions boundary affects the user immediately.

To change the policy used to set the permissions boundary for a user

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Users**.
3. Choose the name of the user whose permissions boundary you want to change.
4. Choose the **Permissions** tab. If necessary, open the **Permissions boundary** section and then choose **Change boundary**.

5. Select the policy that you want to use for the permissions boundary.
6. Choose **Change boundary**.

Removing a Permissions Policy from a User (Console)

Removing a policy affects the user immediately.

To revoke permissions for IAM users

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Users**.
3. Choose the name of the user whose permissions boundary you want to remove.
4. Choose the **Permissions** tab.
5. If you want to revoke permissions by removing an existing policy, view the **Policy type** to understand how the user is getting that policy before choosing **X** to remove the policy:
 - If the policy applies because of group membership, then choosing **X** removes the user from the group. Remember that you might have multiple policies attached to a single group. If you remove a user from a group, the user loses access to *all* policies that it received through that group membership.
 - If the policy is a managed policy attached directly to the user, then choosing **X** detaches the policy from the user. This does not affect the policy itself or any other entity that the policy might be attached to.
 - If the policy is an inline embedded policy, then choosing **X** removes the policy from IAM. Inline policies that are attached directly to a user exist only on that user.

Removing the Permissions Boundary from a User (Console)

Removing a permissions boundary affects the user immediately.

To remove the permissions boundary from a user

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Users**.
3. Choose the name of the user whose permissions boundary you want to remove.
4. Choose the **Permissions** tab. If necessary, open the **Permissions boundary** section and then choose **Remove boundary**.
5. Choose **Remove** to confirm that you want to remove the permissions boundary.

Adding and Removing a User's Permissions (AWS CLI or AWS API)

To add or remove permissions programmatically, you must add or remove the group memberships, attach or detach the managed policies, or add or delete the inline policies. For more information, see the following topics:

- [Adding and Removing Users in an IAM Group \(p. 171\)](#)
- [Adding and Removing IAM Identity Permissions \(p. 457\)](#)

Managing Passwords

You can manage passwords for your AWS account root user and for IAM users in your account. IAM users need passwords in order to access the AWS Management Console. Users do not need passwords to access AWS resources programmatically by using the AWS CLI, Tools for Windows PowerShell, the AWS SDKs or APIs. For those environments, users need [access keys \(p. 110\)](#) instead.

Topics

- [Changing the AWS Account Root User Password \(p. 100\)](#)
- [Setting an Account Password Policy for IAM Users \(p. 101\)](#)
- [Managing Passwords for IAM Users \(p. 104\)](#)
- [Permitting IAM Users to Change Their Own Passwords \(p. 108\)](#)
- [How an IAM User Changes Their Own Password \(p. 109\)](#)

Changing the AWS Account Root User Password

You must be signed in as the AWS account root user and not an IAM user to change the root user password. To learn how to reset a *forgotten* root user password, see [Resetting Your Lost or Forgotten Passwords or Access Keys \(p. 118\)](#).

To change the password for the root user

1. Use your AWS account email address and password to sign in to the [AWS Management Console](#) as the root user.

Note

If you previously signed in to the console with [IAM user \(p. 84\)](#) credentials, your browser might remember this preference and open your account-specific sign-in page. You cannot use the IAM user sign-in page to sign in with your AWS account root user credentials. If you see the IAM user sign-in page, choose **Sign-in using root account credentials** near the bottom of the page to return to the main sign-in page. From there, you can type your AWS account email address and password.

2. In the upper right corner of the console, choose your account name or number and then choose **My Account**.
3. On the right side of the page, next to the **Account Settings** section, choose **Edit**.
4. On the **Password** line choose **Edit** to change your password.
5. Choose a strong password. Although you can [set an account password policy for IAM users \(p. 101\)](#), that policy does not apply to your AWS account root user.

AWS requires that your password meet these conditions:

- have a minimum of 8 characters and a maximum of 128 characters
- include a minimum of three of the following mix of character types: uppercase, lowercase, numbers, and ! @ # \$ % ^ & * () <> [] {} | _+=- symbols
- not be identical to your AWS account name or email address

Note

AWS is rolling out improvements to the sign-in process. One of those improvements is to enforce a more secure password policy for your account. If your account has been upgraded, you are required to meet the password policy above. If your account has not yet been upgraded, then AWS does not enforce this policy, but highly recommends that you follow its guidelines for a more secure password.

To protect your password, it's important to follow these best practices:

- Change your password periodically and keep your password private, since anyone who knows your password may access your account.
- Use a different password on AWS than you use on other sites.
- Avoid passwords that are easy to guess. These include passwords such as `secret`, `password`, `amazon`, or `123456`. They also include things like a dictionary word, your name, email address, or other personal information that can easily be obtained.

Setting an Account Password Policy for IAM Users

You can set a password policy on your AWS account to specify complexity requirements and mandatory rotation periods for your IAM users' passwords.

You can use a password policy to do these things:

- Set a minimum password length.
- Require specific character types, including uppercase letters, lowercase letters, numbers, and non-alphanumeric characters. Be sure to remind your users that passwords are case sensitive.
- Allow all IAM users to change their own passwords.

Note

When you allow your IAM users to change their own passwords, IAM automatically allows them to view the password policy. IAM users need permission to view the account's password policy in order to create a password that complies with the policy.

- Require IAM users to change their password after a specified period of time (enable password expiration).
- Prevent IAM users from reusing previous passwords.
- Force IAM users to contact an account administrator when the user has allowed his or her password to expire.

Important

The password settings described here apply only to **passwords** assigned to IAM users and do not affect any access keys they might have. If a password expires, the user cannot sign in to the AWS Management Console. However, if the user has valid access keys, then the user can still run any AWS CLI or Tools for Windows PowerShell commands. Users can also call any API operations through an application that the user's permissions allow.

When you create or change a password policy, most of the password policy settings are enforced the next time your users change their passwords. However, some of the settings are enforced immediately. For example:

- When you set minimum length and character type requirements, the settings are enforced the next time your users change their passwords. Users are not forced to change their existing passwords, even if the existing passwords do not adhere to the updated password policy.
- When you set a password expiration period, the expiration period is enforced immediately. For example, assume that you set a password expiration period of 90 days. In that case, all IAM users that currently have an existing password that is older than 90 days are required to change their password at next sign-in.

For information about the permissions that you need in order to set a password policy, see [Permitting IAM Users to Change Their Own Passwords \(p. 108\)](#).

The IAM password policy does not apply to the AWS account root user password.

The options currently available do not allow you to create what is commonly called a "lockout policy." Such a policy locks a user out of the account after a specified number of failed sign-in attempts. To get that kind of enhanced security, we recommend that you combine password policies together with multi-factor authentication (MFA). For more information about MFA, see [Using Multi-Factor Authentication \(MFA\) in AWS \(p. 119\)](#).

Topics

- [Password Policy Options \(p. 102\)](#)
- [Setting a Password Policy \(Console\) \(p. 103\)](#)
- [Setting a Password Policy \(AWS CLI\) \(p. 104\)](#)
- [Setting a Password Policy \(AWS API\) \(p. 104\)](#)

Password Policy Options

The following list describes the options that are available when you configure a password policy for your account.

Minimum password length

You can specify the minimum number of characters allowed in an IAM user password. You can type any number from 6 to 128.

Require at least one uppercase letter

You can require that IAM user passwords contain at least one uppercase character from the ISO basic Latin alphabet (A to Z).

Require at least one lowercase letter

You can require that IAM user passwords contain at least one lowercase character from the ISO basic Latin alphabet (a to z).

Require at least one number

You can require that IAM user passwords contain at least one numeric character (0 to 9).

Require at least one nonalphanumeric character

You can require that IAM user passwords contain at least one of the following nonalphanumeric characters:

! @ # \$ % ^ & * () _ + - = [] { } | '

Allow users to change their own password

You can permit all IAM users in your account to use the IAM console to change their own passwords, as described in [Permitting IAM Users to Change Their Own Passwords \(p. 108\)](#).

Alternatively, you can let only some users manage passwords, either for themselves or for others. To do so, you clear the **Allow users to change their own password** check box. For more information about using policies to limit who can manage passwords, see [Permitting IAM Users to Change Their Own Passwords \(p. 108\)](#).

Note

When you allow your IAM users to change their own passwords, IAM automatically allows them to view the password policy. IAM users need permission to view the account's password policy in order to create a password that complies with the policy.

Enable password expiration

You can set IAM user passwords to be valid for only the specified number of days. You specify the number of days that passwords remain valid after they are set. For example, when you enable

password expiration and set the password expiration period to 90 days, an IAM user can use a password for up to 90 days. After 90 days, the password expires and the IAM user must set a new password before accessing the AWS Management Console. You can choose a password expiration period between 1 and 1095 days, inclusive.

Note

The AWS Management Console warns IAM users when they are within 15 days of password expiration. IAM users can change their password at any time (as long as they have been given permission to do so). When they set a new password, the rotation period for that password starts over. An IAM user can have only one valid password at a time.

Prevent password reuse

You can prevent IAM users from reusing a specified number of previous passwords. You can set the number of previous passwords from 1 to 24, inclusive.

Password expiration requires administrator reset

You can prevent IAM users from choosing a new password after their current password has expired. For example, a password policy can specify a password expiration period. If an IAM user fails to choose a new password before the expiration period ends, the IAM user cannot set a new password. In that case, the IAM user must request a password reset from an account administrator in order to regain access to the AWS Management Console. You can also leave this check box cleared. If an IAM user allows his or her password to expire, the user in this scenario is required to set a new password before accessing the AWS Management Console.

Warning

Before you enable this option, be sure that your AWS account has more than one user with administrative permissions (that is, permission to reset IAM user passwords). Or you can ensure that your administrators also have access keys that enable them to use the AWS CLI or Tools for Windows PowerShell separately from the AWS Management Console. When this option is enabled and one administrator's password expires, a second administrator is required to sign-in to the console to reset the expired password of the first administrator. However, if the administrator with the expired password has valid access keys then he or she can run an AWS CLI or Tools for Windows PowerShell command. These commands can reset the administrator's password. The requirement for a second administrator applies only if a password expires and the first administrator has no access keys.

Setting a Password Policy (Console)

You can use the AWS Management Console to create, change, or delete a password policy. As part of managing the password policy, you can let all users manage their own passwords.

To create or change a password policy (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, click **Account Settings**.
3. In the **Password Policy** section, select the options you want to apply to your password policy.
4. Click **Apply Password Policy**.

To delete a password policy

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, click **Account Settings**, and then in the **Password Policy** section, click **Delete Password Policy**.

Setting a Password Policy (AWS CLI)

To manage an account password policy from the AWS CLI, run the following commands:

- To create or change a password policy: `aws iam update-account-password-policy`
- To retrieve the password policy: `aws iam get-account-password-policy`
- To delete a password policy: `aws iam delete-account-password-policy`

Setting a Password Policy (AWS API)

To manage an account password policy from the AWS API, call the following operations:

- To create or change a password policy: `UpdateAccountPasswordPolicy`
- To retrieve the password policy: `GetAccountPasswordPolicy`
- To delete a password policy: `DeleteAccountPasswordPolicy`

Managing Passwords for IAM Users

IAM users who use the AWS Management Console to work with AWS resources must have a password in order to sign in. You can create, change, or delete a password for an IAM user in your AWS account.

After you have assigned a password to a user, the user can sign in to the AWS Management Console using the sign-in URL for your account, which looks like this:

```
https://12-digit-AWS-account-ID or alias.signin.aws.amazon.com/console
```

For more information about how IAM users sign in to the AWS Management Console, see [The IAM Console and Sign-In Page \(p. 71\)](#).

In addition to manually creating individual passwords for your IAM users, you can create a password policy that applies to all IAM user passwords in your AWS account.

You can use a password policy to do these things:

- Set a minimum password length.
- Require specific character types, including uppercase letters, lowercase letters, numbers, and non-alphanumeric characters. Be sure to remind your users that passwords are case sensitive.
- Allow all IAM users to change their own passwords.

Note

When you allow your IAM users to change their own passwords, IAM automatically allows them to view the password policy. IAM users need permission to view the account's password policy in order to create a password that complies with the policy.

- Require IAM users to change their password after a specified period of time (enable password expiration).
- Prevent IAM users from reusing previous passwords.
- Force IAM users to contact an account administrator when the user has allowed his or her password to expire.

For information about managing your account's password policy, see [Setting an Account Password Policy for IAM Users \(p. 101\)](#).

Even if your users have their own passwords, they still need permissions to access your AWS resources. By default, a user has no permissions. To give your users the permissions they need, you assign policies to them or to the groups they belong to. For information about creating users and groups, see [Identities \(Users, Groups, and Roles\) \(p. 82\)](#). For information about using policies to set permissions, see [Changing Permissions for an IAM User \(p. 95\)](#).

You can grant users permission to change their own passwords. For more information, see [Permitting IAM Users to Change Their Own Passwords \(p. 108\)](#). For information about how users access your account sign-in page, see [The IAM Console and Sign-In Page \(p. 71\)](#).

Topics

- [Creating, Changing, or Deleting an IAM User Password \(Console\) \(p. 105\)](#)
- [Creating, Changing, or Deleting an IAM User Password \(AWS CLI\) \(p. 106\)](#)
- [Creating, Changing, or Deleting an IAM User Password \(AWS API\) \(p. 107\)](#)

Creating, Changing, or Deleting an IAM User Password (Console)

You can use the AWS Management Console to manage passwords for your IAM users.

When users leave your organization or no longer need AWS access, it is important to find the credentials that they were using and ensure that they are no longer operational. Ideally, you delete credentials if they are no longer needed. You can always recreate them at a later date if the need arises. At the very least, you should change the credentials so that the former users no longer have access.

To add a password for an IAM user (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Users**.
3. Choose the name of the user whose password you want to create.
4. Choose the **Security credentials** tab, and then under **Sign-in credentials**, choose **Manage password** next to **Console password**.
5. In **Manage console access**, for **Console access** choose **Enable** if not already selected. If console access is disabled, then no password is needed.
6. For **Set password**, choose whether to have IAM generate a password or create a custom password:
 - To have IAM generate a password, choose **Autogenerated password**.
 - To create a custom password, choose **Custom password**, and type the password.

Note

The password that you create must meet the account's [password policy \(p. 101\)](#), if one is currently set.

7. To require the user to create a new password when signing in, choose **Require password reset**. Then choose **Apply**.

Important

If you select the **Require password reset** option, make sure that the user has permission to change his or her password. For more information, see [Permitting IAM Users to Change Their Own Passwords \(p. 108\)](#).

8. If you choose the option to generate a password, choose **Show** in the **New password** dialog box. This lets you view the password so you can share it with the user.

Important

For security reasons, you cannot access the password after completing this step, but you can create a new password at any time.

To change the password for an IAM user (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Users**.
3. Choose the name of the user whose password you want to change.
4. Choose the **Security credentials** tab, and then under **Sign-in credentials**, choose **Manage password** next to **Console password**.
5. In **Manage console access**, for **Console access** choose **Enable** if not already selected. If console access is disabled, then no password is needed.
6. For **Set password**, choose whether to have IAM generate a password or create a custom password:
 - To have IAM generate a password, choose **Autogenerated password**.
 - To create a custom password, choose **Custom password**, and type the password.

Note

The password that you create must meet the account's [password policy \(p. 101\)](#), if one is currently set.

7. To require the user to create a new password when signing in, choose **Require password reset**. Then choose **Apply**.

Important

If you select the **Require password reset** option, make sure that the user has permission to change his or her password. For more information, see [Permitting IAM Users to Change Their Own Passwords \(p. 108\)](#).

8. If you choose the option to generate a password, choose **Show** in the **New password** dialog box. This lets you view the password so you can share it with the user.

Important

For security reasons, you cannot access the password after completing this step, but you can create a new password at any time.

To delete (disable) an IAM user's password (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Users**.
3. Choose the name of the user whose password you want to delete.
4. Choose the **Security credentials** tab, and then under **Sign-in credentials**, choose **Manage password** next to **Console password**.
5. For **Console access**, choose **Disable**, and then choose **Apply**.

Important

When you delete a user's password, the user can no longer sign in to the AWS Management Console. If the user has active access keys, they continue to function and allow access through the AWS CLI, Tools for Windows PowerShell, or AWS API function calls.

Creating, Changing, or Deleting an IAM User Password (AWS CLI)

You can use the AWS CLI API to manage passwords for your IAM users.

To create a password (AWS CLI)

1. (Optional) To determine whether a user has a password, run this command: `aws iam get-login-profile`

2. To create a password, run this command: [aws iam create-login-profile](#)

To change a user's password (AWS CLI)

1. (Optional) To determine whether a user has a password, run this command: [aws iam get-login-profile](#)
2. To change a password, run this command: [aws iam update-login-profile](#)

To delete (disable) a user's password (AWS CLI)

1. (Optional) To determine whether a user has a password, run this command: [aws iam get-login-profile](#)
2. (Optional) To determine when a password was last used, run this command: [aws iam get-user](#)
3. To delete a password, run this command: [aws iam delete-login-profile](#)

Important

When you delete a user's password, the user can no longer sign in to the AWS Management Console. If the user has active access keys, they continue to function and allow access through the AWS CLI, Tools for Windows PowerShell, or AWS API function calls. When you use the AWS CLI, Tools for Windows PowerShell, or AWS API to delete a user from your AWS account, you must first delete the password using this operation. For more information, see [Deleting an IAM User \(AWS CLI\) \(p. 94\)](#).

Creating, Changing, or Deleting an IAM User Password (AWS API)

You can use the AWS API to manage passwords for your IAM users.

To create a password (AWS API)

1. (Optional) To determine whether a user has a password, call this operation: [GetLoginProfile](#)
2. To create a password, call this operation: [CreateLoginProfile](#)

To change a user's password (AWS API)

1. (Optional) To determine whether a user has a password, call this operation: [GetLoginProfile](#)
2. To change a password, call this operation: [UpdateLoginProfile](#)

To delete (disable) a user's password (AWS API)

1. (Optional) To determine whether a user has a password, run this command: [GetLoginProfile](#)
2. (Optional) To determine when a password was last used, run this command: [GetUser](#)
3. To delete a password, run this command: [DeleteLoginProfile](#)

Important

When you delete a user's password, the user can no longer sign in to the AWS Management Console. If the user has active access keys, they continue to function and allow access through the AWS CLI, Tools for Windows PowerShell, or AWS API function calls. When you use the AWS CLI, Tools for Windows PowerShell, or AWS API to delete a user from your AWS account, you must first delete the password using this operation. For more information, see [Deleting an IAM User \(AWS CLI\) \(p. 94\)](#).

Permitting IAM Users to Change Their Own Passwords

You can grant IAM users the permission to change their own passwords for signing in to the AWS Management Console. You can do this in one of two ways:

- [Allow all IAM users in the account to change their own passwords \(p. 108\)](#).
- [Allow only selected IAM users to change their own passwords \(p. 108\)](#). In this scenario, you disable the option for all users to change their own passwords and you use an IAM policy to grant permissions to only some users to change their own passwords and optionally other credentials like their own access keys.

Important

We recommend that you [set a password policy \(p. 101\)](#) so that users create strong passwords.

To allow all IAM users change their own passwords

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, click **Account Settings**.
3. In the **Password Policy** section, select **Allow users to change their own password**, and then click **Apply Password Policy**.
4. Point users to the following instructions that show how they can change their passwords: [How an IAM User Changes Their Own Password \(p. 109\)](#).

For information about the AWS CLI, Tools for Windows PowerShell, and API commands that you can use to change the account's password policy (which includes letting all users change their own passwords), see [Setting a Password Policy \(AWS CLI\) \(p. 104\)](#).

To allow selected IAM users change their own passwords

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, click **Account Settings**.
3. In the **Account Settings** section, make sure that **Allow users to change their own password** is not selected. If this check box is selected, all users can change their own passwords. (See the previous procedure.)
4. Create the users who should be able to change their own password, if they do not already exist. For details, see [Creating an IAM User in Your AWS Account \(p. 86\)](#).
5. Create an IAM group for the users who should be allowed to change their passwords, and then add the users from the previous step to the group. For details, see [Creating Your First IAM Admin User and Group \(p. 20\)](#) and [Managing IAM Groups \(p. 170\)](#).

This step is optional, but it's a best practice to use groups to manage permissions so that you can add and remove users and change the permissions for the group as a whole.

6. Assign the following policy to the group. For details, see [Managing IAM Policies \(p. 442\)](#).

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": "iam:GetAccountPasswordPolicy",  
            "Resource": "*"  
        },  
        {  
            "Effect": "Allow",  
            "Action": "iam:ChangePassword",  
            "Resource": "*"  
        }  
    ]  
}
```

```
        "Effect": "Allow",
        "Action": "iam:ChangePassword",
        "Resource": "arn:aws:iam::account-id-without-hyphens:user/${aws:username}"
    }
}
```

This policy grants access to the [ChangePassword](#) action, which lets users change only their own passwords from the console, the AWS CLI, Tools for Windows PowerShell, or the API. It also grants access to the [GetAccountPasswordPolicy](#) action, which lets the user view the current password policy; this permission is required so that the user can display the **Change Password** page in the console. The user must be able to read the current password policy to ensure the changed password meets the requirements of the policy.

7. Point users to the following instructions that show how they can change their passwords: [How an IAM User Changes Their Own Password \(p. 109\)](#).

For More Information

For more information on managing credentials, see the following topics:

- [Permitting IAM Users to Change Their Own Passwords \(p. 108\)](#)
- [Managing Passwords \(p. 100\)](#)
- [Setting an Account Password Policy for IAM Users \(p. 101\)](#)
- [Managing IAM Policies \(p. 442\)](#)
- [How an IAM User Changes Their Own Password \(p. 109\)](#)

How an IAM User Changes Their Own Password

If you have been granted permission to change your own IAM user password, you can use a special page in the AWS Management Console to do this. You can also use the AWS CLI or AWS API.

Topics

- [Permissions Required \(p. 109\)](#)
- [How IAM Users Change Their Own Password \(Console\) \(p. 109\)](#)
- [How IAM Users Change Their Own Password \(AWS CLI or AWS API\) \(p. 110\)](#)

Permissions Required

To change the password for your own IAM user, you must have the permissions from the following policy: [AWS: Allows IAM Users to Change Their Own Console Password on the My Security Credentials Page \(p. 406\)](#).

How IAM Users Change Their Own Password (Console)

The following procedure describes how IAM users can use the AWS Management Console to change their own password.

To change your own IAM user password (console)

1. Use your AWS account ID or account alias, your IAM user name, and your password to sign in to the [IAM console](#).

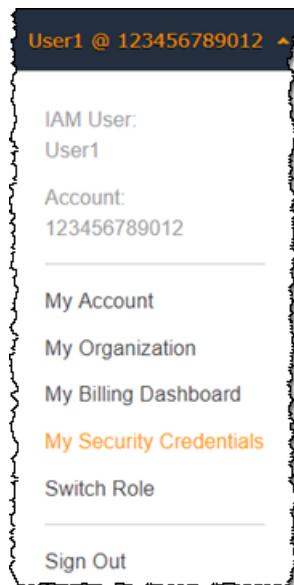
Note

For your convenience, the AWS sign-in page uses a browser cookie to remember your IAM user name and account information. If you previously signed in as a different user, choose

Sign in to a different account near the bottom of the page to return to the main sign-in page. From there, you can type your AWS account ID or account alias to be redirected to the IAM user sign-in page for your account.

To get your AWS account ID, contact your administrator.

2. In the navigation bar on the upper right, choose your user name, and then choose **My Security Credentials**.



3. On the **AWS IAM Credentials** tab, choose **Change password**.
4. For **Current password**, enter your current password. Enter a new password for **New password** and **Confirm new password**. Then choose **Change password**.

Note

If the account has a password policy, the new password must meet the requirements of that policy. For more information, see [Setting an Account Password Policy for IAM Users \(p. 101\)](#).

How IAM Users Change Their Own Password (AWS CLI or AWS API)

The following procedure describes how IAM users can use the AWS CLI or AWS API to change their own password.

To change your own IAM password, use the following:

- AWS CLI: `aws iam change-password`
- AWS API: `ChangePassword`

Managing Access Keys for IAM Users

Follow us on Twitter

Note

If you found this topic because you are trying to configure the Product Advertising API to sell Amazon products on your website, see these topics:

- [Getting Started with the Product Advertising API](#)

- [Getting Started as a Product Advertising API Developer](#)

Access keys are long-term credentials for an IAM user or the AWS account root user. You can use access keys to sign programmatic requests to the AWS CLI or AWS API (directly or using the AWS SDK). For more information, see [Signing AWS API Requests](#) in the *Amazon Web Services General Reference*.

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

Important

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

As a best practice, use temporary security credentials (IAM roles) instead of access keys, and disable any AWS account root user access keys. For more information, see [Best Practices for Managing AWS Access Keys](#) in the *Amazon Web Services General Reference*.

If you still need to use long-term access keys, you can create, modify, view, or rotate your access keys (access key IDs and secret access keys). You can have a maximum of two access keys. This allows you to rotate the active keys according to best practices.

When you create an access key pair, save the access key ID and secret access key in a secure location. The secret access key is available only at the time you create it. If you lose your secret access key, you must delete the access key and create a new one. For more details, see [Resetting Your Lost or Forgotten Passwords or Access Keys \(p. 118\)](#).

Topics

- [Permissions Required \(p. 111\)](#)
- [Managing Access Keys \(Console\) \(p. 112\)](#)
- [Managing Access Keys \(AWS CLI\) \(p. 114\)](#)
- [Managing Access Keys \(AWS API\) \(p. 114\)](#)
- [Rotating Access Keys \(p. 115\)](#)
- [Auditing Access Keys \(p. 118\)](#)

Permissions Required

To create access keys for your own IAM user, you must have the permissions from the following policy:

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "CreateOwnAccessKeys",  
            "Effect": "Allow",  
            "Action": [  
                "iam:CreateAccessKey",  
                "iam:GetUser",  
                "iam>ListAccessKeys"  
            ],  
            "Resource": "arn:aws:iam::*:user/${aws:username}"  
        }  
    ]  
}
```

To rotate access keys for your own IAM user, you must have the permissions from the following policy:

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "ManageOwnAccessKeys",  
            "Effect": "Allow",  
            "Action": [  
                "iam:CreateAccessKey",  
                "iam>DeleteAccessKey",  
                "iam:GetAccessKeyLastUsed",  
                "iam:GetUser",  
                "iam>ListAccessKeys",  
                "iam:UpdateAccessKey"  
            ],  
            "Resource": "arn:aws:iam::*:user/${aws:username}"  
        }  
    ]  
}
```

Managing Access Keys (Console)

You can use the AWS Management Console to manage an IAM user's access keys.

To create, modify, or delete your own IAM user access keys (console)

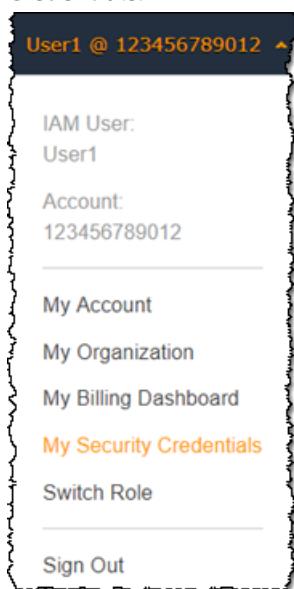
1. Use your AWS account ID or account alias, your IAM user name, and your password to sign in to the [IAM console](#).

Note

For your convenience, the AWS sign-in page uses a browser cookie to remember your IAM user name and account information. If you previously signed in as a different user, choose **Sign in to a different account** near the bottom of the page to return to the main sign-in page. From there, you can type your AWS account ID or account alias to be redirected to the IAM user sign-in page for your account.

To get your AWS account ID, contact your administrator.

2. In the navigation bar on the upper right, choose your user name, and then choose **My Security Credentials**.



3. On the **AWS IAM Credentials** tab, in the **Access keys for CLI, SDK, and API access** section, do any of the following:
 - To create an access key, choose **Create access key**. Then choose **Download .csv file** to save the access key ID and secret access key to a .csv file on your computer. Store the file in a secure location. You will not have access to the secret access key again after this dialog box closes. After you have downloaded the .csv file, choose **Close**. When you create an access key, the key pair is active by default, and you can use the pair right away.
 - To disable an active access key, choose **Make inactive**.
 - To reenable an inactive access key, choose **Make active**.
 - To delete an access key, choose its X button at the far right of the row. Then choose **Delete** to confirm. When you delete an access key, it's gone forever and cannot be retrieved. However, you can always create new keys.

To create, modify, or delete another IAM user's access keys (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Users**.
3. Choose the name of the user whose access keys you want to manage, and then choose the **Security credentials** tab.
4. In the **Access keys** section, do any of the following:
 - To create an access key, choose **Create access key**. Then choose **Download .csv file** to save the access key ID and secret access key to a CSV file on your computer. Store the file in a secure location. You will not have access to the secret access key again after this dialog box closes. After you download the CSV file, choose **Close**. When you create an access key, the key pair is active by default, and you can use the pair right away.
 - To disable an active access key, choose **Make inactive**.
 - To reenable an inactive access key, choose **Make active**.
 - To delete an access key, choose its X button at the far right of the row. Then choose **Delete** to confirm. When you delete an access key, it's gone forever and cannot be retrieved. However, you can always create new keys.

To list the access keys for an IAM user (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Users**.
3. Choose the name of the intended user, and then choose the **Security credentials** tab. The user's access keys and the status of each key is displayed.

Note

Only the user's access key ID is visible. The secret access key can only be retrieved when the key is created.

To list the access key IDs for multiple IAM users (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Users**.
3. If necessary, add the **Access key ID** column to the users table by completing the following steps:

- a. Above the table on the far right, choose the settings icon ().
 - b. In **Manage columns**, select **Access key ID**.
 - c. Choose **Close** to return to the list of users.
4. The **Access key ID** column shows each access key ID, followed by its state; for example, **23478207027842073230762374023 (Active)** or **22093740239670237024843420327 (Inactive)**.

You can use this information to view and copy the access keys for users with one or two access keys. The column displays **None** for users with no access key.

Note

Only the user's access key ID and status is visible. The secret access key can only be retrieved when the key is created.

To find which IAM user owns a specific access key (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Users**.
3. In the search box, type or paste the access key ID of the user you want to find.
4. If necessary, add the **Access key ID** column to the users table by completing the following steps:
 - a. Above the table on the far right, choose the settings icon ().
 - b. In **Manage columns**, select **Access key ID**.
 - c. Choose **Close** to return to the list of users and confirm that the filtered user owns the specified access key.

Managing Access Keys (AWS CLI)

To manage an IAM user's access keys from the AWS CLI, run the following commands.

- To create an access key: `aws iam create-access-key`
- To disable or reenable an access key: `aws iam update-access-key`
- To list a user's access keys: `aws iam list-access-keys`
- To determine when an access key was most recently used: `aws iam get-access-key-last-used`
- To delete an access key: `aws iam delete-access-key`

Managing Access Keys (AWS API)

To manage an IAM user's access keys from the AWS API, call the following operations.

- To create an access key: `CreateAccessKey`
- To disable or reenable an access key: `UpdateAccessKey`
- To list a user's access keys: `ListAccessKeys`
- To determine when an access key was most recently used: `GetAccessKeyLastUsed`
- To delete an access key: `DeleteAccessKey`

Rotating Access Keys

As a security best practice, we recommend that you regularly rotate (change) IAM user access keys. If your administrator granted you the necessary permissions, you can rotate your own access keys.

Administrators, for details about granting your users permissions to rotate their own access keys, see [AWS: Allows IAM Users to Manage Their Own Password, Access Keys, and SSH Public Keys on the My Security Credentials Page \(p. 406\)](#). You can also apply a password policy to your account to require that all of your IAM users periodically rotate their passwords. You can choose how often they must do so. For more information, see [Setting an Account Password Policy for IAM Users \(p. 101\)](#).

Important

As a best practice, do not use your AWS account root user. If you use the AWS account root user credentials, we recommend that you also regularly rotate them. The account password policy does not apply to the root user credentials. IAM users cannot manage credentials for the AWS account root user, so you must use the root user credentials (not a user's) to change the root user credentials. Note that we recommend against using the root user for everyday work in AWS.

Topics

- [Rotating IAM User Access Keys \(Console\) \(p. 115\)](#)
- [Rotating Access Keys \(AWS CLI\) \(p. 116\)](#)
- [Rotating Access Keys \(AWS API\) \(p. 117\)](#)

Rotating IAM User Access Keys (Console)

You can rotate access keys from the AWS Management Console.

To rotate access keys for an IAM user without interrupting your applications (console)

1. While the first access key is still active, create a second access key.
 - a. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
 - b. In the navigation pane, choose **Users**.
 - c. Choose the name of the intended user, and then choose the **Security credentials** tab.
 - d. Choose **Create access key** and then choose **Download .csv file** to save the access key ID and secret access key to a .csv file on your computer. Store the file in a secure location. You will not have access to the secret access key again after this closes. After you have downloaded the .csv file, choose **Close**.

The new access key is active by default. At this point, the user has two active access keys.

- 2. Update all applications and tools to use the new access key.
- 3. Determine whether the first access key is still in use by reviewing the **Last used** column for the oldest access key. One approach is to wait several days and then check the old access key for any use before proceeding.
- 4. Even if the **Last used** column value indicates that the old key has never been used, we recommend that you do not immediately delete the first access key. Instead, choose **Make inactive** to deactivate the first access key.
- 5. Use only the new access key to confirm that your applications are working. Any applications and tools that still use the original access key will stop working at this point because they no longer have access to AWS resources. If you find such an application or tool, you can choose **Make active** to reenable the first access key. Then return to [Step 3 \(p. 115\)](#) and update this application to use the new key.
- 6. After you wait some period of time to ensure that all applications and tools have been updated, you can delete the first access key:

- a. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
- b. In the navigation pane, choose **Users**.
- c. Choose the name of the intended user, and then choose the **Security credentials** tab.
- d. Locate the access key to delete and choose its X button at the far right of the row. Then choose **Delete** to confirm.

To determine when access keys need rotating (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Users**.
3. If necessary, add the **Access key age** column to the users table by completing the following steps:
 - a. Above the table on the far right, choose the settings icon ().
 - b. In **Manage columns**, select **Access key age**.
 - c. Choose **Close** to return to the list of users.
4. The **Access key age** column shows the number of days since the oldest active access key was created. You can use this information to find users with access keys that need rotating. The column displays **None** for users with no access key.

Rotating Access Keys (AWS CLI)

You can rotate access keys from the AWS Command Line Interface.

To rotate access keys without interrupting your applications (AWS CLI)

1. While the first access key is still active, create a second access key, which is active by default. Run the following command:
 - `aws iam create-access-key`

At this point, the user has two active access keys.
2. Update all applications and tools to use the new access key.
3. Determine whether the first access key is still in use by using this command:
 - `aws iam get-access-key-last-used`

One approach is to wait several days and then check the old access key for any use before proceeding.

4. Even if step **Step 3** indicates no use of the old key, we recommend that you do not immediately delete the first access key. Instead, change the state of the first access key to **Inactive** using this command:
 - `aws iam update-access-key`
5. Use only the new access key to confirm that your applications are working. Any applications and tools that still use the original access key will stop working at this point because they no longer have access to AWS resources. If you find such an application or tool, you can switch its state back to **Active** to reenable the first access key. Then return to step **Step 2** and update this application to use the new key.

6. After you wait some period of time to ensure that all applications and tools have been updated, you can delete the first access key with this command:

- `aws iam delete-access-key`

For more information, see the following:

- [How to Rotate Access Keys for IAM Users](#). This entry on the *AWS Security Blog* provides more information on key rotation.
- [IAM Best Practices \(p. 60\)](#). This page provides general recommendations for helping to secure your AWS resources.

Rotating Access Keys (AWS API)

You can rotate access keys using the AWS API.

To rotate access keys without interrupting your applications (AWS API)

1. While the first access key is still active, create a second access key, which is active by default. Call the following operation:

- `CreateAccessKey`

At this point, the user has two active access keys.

2. Update all applications and tools to use the new access key.
3. Determine whether the first access key is still in use by calling this operation:

- `GetAccessKeyLastUsed`

One approach is to wait several days and then check the old access key for any use before proceeding.

4. Even if step [Step 3](#) indicates no use of the old key, we recommend that you do not immediately delete the first access key. Instead, change the state of the first access key to `Inactive` calling this operation:

- `UpdateAccessKey`

5. Use only the new access key to confirm that your applications are working. Any applications and tools that still use the original access key will stop working at this point because they no longer have access to AWS resources. If you find such an application or tool, you can switch its state back to `Active` to reenable the first access key. Then return to step [Step 2](#) and update this application to use the new key.

6. After you wait some period of time to ensure that all applications and tools have been updated, you can delete the first access key calling this operation:

- `DeleteAccessKey`

For more information, see the following:

- [How to Rotate Access Keys for IAM Users](#). This entry on the *AWS Security Blog* provides more information on key rotation.
- [IAM Best Practices \(p. 60\)](#). This page provides general recommendations for helping to secure your AWS resources.

Auditing Access Keys

You can review the AWS access keys in your code to determine whether the keys are from an account that you own. You can pass an access key ID using the `aws sts get-access-key-info` AWS CLI command or the `GetAccessKeyInfo` AWS API operation.

The AWS CLI and AWS API operations return the ID of the AWS account to which the access key belongs. Access key IDs beginning with `AKIA` are long-term credentials for an IAM user or an AWS account root user. Access key IDs beginning with `ASIA` are temporary credentials that are created using AWS STS operations. If the account in the response belongs to you, you can sign in as the root user and review your root user access keys. Then, you can pull a [credentials report \(p. 156\)](#) to learn which IAM user owns the keys. To learn who requested the temporary credentials for an `ASIA` access key, view the AWS STS events in your [CloudTrail logs \(p. 341\)](#).

This operation does not indicate the state of the access key. The key might be active, inactive, or deleted. Active keys might not have permissions to perform an operation. Providing a deleted access key might return an error that the key doesn't exist.

Resetting Your Lost or Forgotten Passwords or Access Keys

Having trouble signing in? Make sure that you're on the correct [AWS sign-in page \(p. 71\)](#) for your type of user. The sign-in page that you see is different, depending on the user type. You can sign in as the AWS account root user (account owner) or an IAM user that was created by an account administrator. On the main sign-in page, you must enter your email address to sign in as the root user, or enter your account ID to sign in as an IAM user. You can provide your password only on the sign-in page that matches your user type. For more information about the sign-in pages, see [The IAM Console and Sign-In Page \(p. 71\)](#).

If you are on the correct sign-in page and lose or forget your passwords or access keys, you *cannot* retrieve them from IAM. Instead, you can reset them using the following methods:

- **AWS account root user password** – If you forget your root user password, you can reset the password from the AWS Management Console. For details, see [the section called "Resetting a Lost or Forgotten Root User Password" \(p. 119\)](#) later in this topic.
- **AWS account access keys** – If you forget your account access keys, you can create new access keys without disabling the existing access keys. If you are not using the existing keys, you can delete those. For details, see [Creating Access Keys for the Root User \(p. 339\)](#) and [Deleting Access Keys from the Root User \(p. 340\)](#).
- **IAM user password** – If you are an IAM user and you forget your password, you must ask your administrator to reset your password. To learn how an administrator can manage your password, see [Managing Passwords for IAM Users \(p. 104\)](#).
- **IAM user access keys** – If you are an IAM user and you forget your access keys, you will need new access keys. If you have permission to create your own access keys, you can find instructions for creating a new one at [Managing Access Keys \(Console\) \(p. 112\)](#). If you do not have the required permissions, you must ask your administrator to create new access keys. If you are still using your old keys, ask your administrator not to delete the old keys. To learn how an administrator can manage your access keys, see [Managing Access Keys for IAM Users \(p. 110\)](#).

You should follow the [AWS best practice \(p. 65\)](#) of periodically changing your password and AWS access keys. In AWS, you change access keys by *rotating* them. This means that you create a new one, configure your applications to use the new key, and then delete the old one. You are allowed to have two access key pairs active at the same time for just this reason. For more information, see [Rotating Access Keys \(p. 115\)](#).

Resetting a Lost or Forgotten Root User Password

When you first created your AWS account, you provided an email address and password. These are your AWS account root user credentials. If you forget your root user password, you can reset the password from the AWS Management Console.

To reset your root user password:

1. Use your AWS account email address to begin signing in to the [AWS Management Console](#) as the root user.

Note
If you are signed in to the [AWS Management Console](#) with *IAM user* credentials, then you must sign out before you can reset the root user password. If you see the account-specific IAM user sign-in page, choose **Sign-in using root account credentials** near the bottom of the page. If necessary, provide your account email address and choose **Next** to access the **Root user sign in** page.
2. Choose **Forgot your password?**.
3. Provide the email address that you used to create the account. Then provide the CAPTCHA text and choose **Continue**.
4. Check the email that is associated with your AWS account for a message from Amazon Web Services. The email will come from an address ending in @amazon.com or @aws.amazon.com. Follow the directions in the email. If you don't see the email in your account, check your spam folder. If you no longer have access to the email, see [I Need to Access an Old Account \(p. 543\)](#).

Using Multi-Factor Authentication (MFA) in AWS



For increased security, we recommend that you configure multi-factor authentication (MFA) to help protect your AWS resources. You can enable MFA for IAM users or the AWS account root user. When you enable MFA for the root user, it affects only the root user credentials. IAM users in the account are distinct identities with their own credentials, and each identity has its own MFA configuration.

Topics

- [What Is MFA? \(p. 119\)](#)
- [Enabling MFA Devices \(p. 120\)](#)
- [Checking MFA Status \(p. 137\)](#)
- [Resynchronizing Virtual and Hardware MFA Devices \(p. 138\)](#)
- [Deactivating MFA Devices \(p. 142\)](#)
- [What If an MFA Device Is Lost or Stops Working? \(p. 144\)](#)
- [Configuring MFA-Protected API Access \(p. 145\)](#)
- [Sample Code: Requesting Credentials with Multi-factor Authentication \(p. 151\)](#)

What Is MFA?

MFA adds extra security because it requires users to provide unique authentication from an AWS supported MFA mechanism in addition to their regular sign-in credentials when they access AWS websites or services:

- **Virtual MFA devices.** A software app that runs on a phone or other device and emulates a physical device. The device generates a six-digit numeric code based upon a time-synchronized one-time

password algorithm. The user must type a valid code from the device on a second webpage during sign-in. Each virtual MFA device assigned to a user must be unique. A user cannot type a code from another user's virtual MFA device to authenticate. Because they can run on unsecured mobile devices, virtual MFA might not provide the same level of security as U2F devices or hardware MFA devices. We do recommend that you use a virtual MFA device while waiting for hardware purchase approval or while you wait for your hardware to arrive. For a list of a few supported apps that you can use as virtual MFA devices, see [Multi-Factor Authentication](#). For instructions on setting up a virtual MFA device with AWS, see [Enabling a Virtual Multi-factor Authentication \(MFA\) Device \(Console\) \(p. 122\)](#).

- **U2F security key.** A device that you plug into a USB port on your computer. U2F is an open authentication standard hosted by the [FIDO Alliance](#). When you enable a U2F security key, you sign in by entering your credentials and then tapping the device instead of manually entering a code. For information on supported AWS U2F security keys, see [Multi-Factor Authentication](#). For instructions on setting up a U2F security key with AWS, see [Enabling a U2F Security Key \(Console\) \(p. 125\)](#).
- **Hardware MFA device.** A hardware device that generates a six-digit numeric code based upon a time-synchronized one-time password algorithm. The user must type a valid code from the device on a second webpage during sign-in. Each MFA device assigned to a user must be unique. A user cannot type a code from another user's device to be authenticated. For information on supported hardware MFA devices, see [Multi-Factor Authentication](#). For instructions on setting up a hardware MFA device with AWS, see [Enabling a Hardware MFA Device \(Console\) \(p. 130\)](#).
- **SMS text message-based MFA.** A type of MFA in which the IAM user settings include the phone number of the user's SMS-compatible mobile device. When the user signs in, AWS sends a six-digit numeric code by SMS text message to the user's mobile device. The user is required to type that code on a second webpage during sign-in. Note that SMS-based MFA is available only for IAM users. You cannot use this type of MFA with the AWS account root user. For more information about enabling SMS text messaging-based MFA, see [PREVIEW – Enabling SMS Text Message MFA Devices \(p. 135\)](#).

Note

AWS will soon end support for SMS multi-factor authentication (MFA). We are not allowing new customers to preview this feature. We recommend that existing customers switch to one of the following alternative methods of MFA: [virtual \(software-based\) MFA device \(p. 122\)](#), [U2F security key \(p. 125\)](#), or [hardware MFA device \(p. 130\)](#). You can view users in your account with an assigned SMS MFA device. To do so, go to the IAM console, choose **Users** from the navigation pane, and look for users with **SMS** in the **MFA** column of the table.

For answers to commonly asked questions about AWS MFA, go to the [AWS Multi-Factor Authentication FAQs](#).

Enabling MFA Devices

The steps for configuring MFA depend on the type of MFA device you are using.

Topics

- [General Steps for Enabling MFA Devices \(p. 120\)](#)
- [Enabling a Virtual Multi-factor Authentication \(MFA\) Device \(Console\) \(p. 122\)](#)
- [Enabling a U2F Security Key \(Console\) \(p. 125\)](#)
- [Enabling a Hardware MFA Device \(Console\) \(p. 130\)](#)
- [PREVIEW – Enabling SMS Text Message MFA Devices \(p. 135\)](#)
- [Enabling and Managing Virtual MFA Devices \(AWS CLI or AWS API\) \(p. 136\)](#)

General Steps for Enabling MFA Devices

The following overview procedure describes how to set up and use MFA and provides links to related information.

1. *Get an MFA device such as one of the following.* You can enable only one MFA device per AWS account root user or IAM user.
 - A virtual MFA device, which is a software app that is compliant with [RFC 6238, a standards-based TOTP \(time-based one-time password\) algorithm](#). You can install the app on a phone or other device. For a list of a few supported apps that you can use as virtual MFA devices, see [Multi-Factor Authentication](#).
 - A U2F security key with an [AWS supported configuration \(p. 129\)](#), such as one of the U2F devices discussed on the [Multi-Factor Authentication](#) page.
 - A hardware-based MFA device, such as one of the AWS supported hardware token devices discussed on the [Multi-Factor Authentication](#) page.
 - A mobile phone that can receive standard short message service (SMS) text messages.

Notes

- If you choose to use SMS-based MFA, text messaging charges from your mobile device's carrier may apply.
- SMS-based MFA is only available for IAM users and cannot be used for the root user.

2. *Enable the MFA device.*

- IAM users with virtual or hardware MFA devices: Enable from the AWS Management Console, AWS CLI, or the IAM API.
- IAM users with U2F security keys or a mobile phone that can receive SMS text messages: Enable from the AWS Management Console only.
- AWS account root users with any type of MFA device (except SMS MFA, which is not supported for root users): Enable from the AWS Management Console only.

For information about enabling each type of MFA device, see the following pages:

- Virtual MFA device: [Enabling a Virtual Multi-factor Authentication \(MFA\) Device \(Console\) \(p. 122\)](#)
- U2F security key: [Enabling a U2F Security Key \(Console\) \(p. 125\)](#)
- Hardware MFA device: [Enabling a Hardware MFA Device \(Console\) \(p. 130\)](#)
- SMS MFA device: [PREVIEW – Enabling SMS Text Message MFA Devices \(p. 135\)](#)

3. *Use the MFA device when you log in to or access AWS resources.* Note the following:

- U2F security keys: To access an AWS website, enter your credentials and then tap the U2F security key when prompted.
- Virtual MFA devices, hardware MFA devices, and SMS MFA devices: To access an AWS website, you need an MFA code from the device in addition to your user name and password. If AWS determines that the IAM user you sign in as is MFA-enabled with SMS, then it automatically sends the MFA code to the configured phone number.

To access MFA-protected API operations, you need the following:

- An MFA code
- The identifier for the MFA device (the device serial number of a physical device or the ARN of a virtual or SMS device defined in AWS)
- The usual access key ID and secret access key

Notes

- You cannot pass the MFA information for a U2F security key or SMS MFA device to AWS STS API operations to request temporary credentials.
- You cannot use AWS CLI commands or AWS API operations to enable [U2F security keys \(p. 125\)](#).

For more information, see [Using MFA Devices With Your IAM Sign-in Page \(p. 78\)](#).

Enabling a Virtual Multi-factor Authentication (MFA) Device (Console)

You can use a phone or other device as a virtual multi-factor authentication (MFA) device. To do this, install a mobile app that is compliant with [RFC 6238, a standards-based TOTP \(time-based one-time password\) algorithm](#). These apps generate a six-digit authentication code. Because they can run on unsecured mobile devices, virtual MFA might not provide the same level of security as U2F devices or hardware MFA devices. We do recommend that you use a virtual MFA device while waiting for hardware purchase approval or while you wait for your hardware to arrive.

Most virtual MFA apps support creating multiple virtual devices, allowing you to use the same app for multiple AWS accounts or users. However, you can enable only one MFA device per user.

For a list of virtual MFA apps that you can use, see [Multi-Factor Authentication](#). Note that AWS requires a virtual MFA app that produces a six-digit OTP.

Topics

- [Permissions Required \(p. 122\)](#)
- [Enable a Virtual MFA Device for an IAM User \(Console\) \(p. 122\)](#)
- [Enable a Virtual MFA Device for Your AWS Account Root User \(Console\) \(p. 123\)](#)
- [Replace or "Rotate" a Virtual MFA Device \(p. 125\)](#)

Permissions Required

To manage virtual MFA devices for your IAM user, you must have the permissions from the following policy: [AWS: Allows MFA-Authenticated IAM Users to Manage Their Own MFA Device on the My Security Credentials Page \(p. 404\)](#).

Enable a Virtual MFA Device for an IAM User (Console)

You can use IAM in the AWS Management Console to enable and manage a virtual MFA device for an IAM user in your account. To enable and manage an MFA device using the AWS CLI or AWS API, see [Enabling and Managing Virtual MFA Devices \(AWS CLI or AWS API\) \(p. 136\)](#).

Note

You must have physical access to the hardware that will host the user's virtual MFA device in order to configure MFA. For example, you might configure MFA for a user who will use a virtual MFA device running on a smartphone. In that case, you must have the smartphone available in order to finish the wizard. Because of this, you might want to let users configure and manage their own virtual MFA devices. In that case, you must grant users the permissions to perform the necessary IAM actions. For more information and for an example of an IAM policy that grants these permissions, see [AWS: Allows MFA-Authenticated IAM Users to Manage Their Own MFA Device on the My Security Credentials Page \(p. 404\)](#).

To enable a virtual MFA device for an IAM user (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Users**.
3. In the **User Name** list, choose the name of the intended MFA user.
4. Choose the **Security credentials** tab. Next to **Assigned MFA device**, choose **Manage**.
5. In the **Manage MFA Device** wizard, choose **Virtual MFA device**, and then choose **Continue**.

IAM generates and displays configuration information for the virtual MFA device, including a QR code graphic. The graphic is a representation of the "secret configuration key" that is available for manual entry on devices that do not support QR codes.

6. Open your virtual MFA app. For a list of apps that you can use for hosting virtual MFA devices, see [Multi-Factor Authentication](#).

If the virtual MFA app supports multiple virtual MFA devices or accounts, choose the option to create a new virtual MFA device or account.

7. Determine whether the MFA app supports QR codes, and then do one of the following:

- From the wizard, choose **Show QR code**, and then use the app to scan the QR code. For example, you might choose the camera icon or choose an option similar to **Scan code**, and then use the device's camera to scan the code.
- In the **Manage MFA Device** wizard, choose **Show secret key**, and then type the secret key into your MFA app.

When you are finished, the virtual MFA device starts generating one-time passwords.

8. In the **Manage MFA Device** wizard, in the **MFA code 1** box, type the one-time password that currently appears in the virtual MFA device. Wait up to 30 seconds for the device to generate a new one-time password. Then type the second one-time password into the **MFA code 2** box. Choose **Assign MFA**.

Important

Submit your request immediately after generating the codes. If you generate the codes and then wait too long to submit the request, the MFA device successfully associates with the user but the MFA device is out of sync. This happens because time-based one-time passwords (TOTP) expire after a short period of time. If this happens, you can [resync the device \(p. 138\)](#).

The virtual MFA device is now ready for use with AWS. For information about using MFA with the AWS Management Console, see [Using MFA Devices With Your IAM Sign-in Page \(p. 78\)](#).

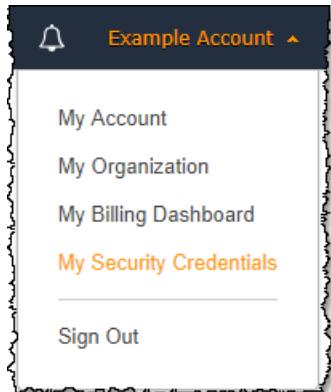
[Enable a Virtual MFA Device for Your AWS Account Root User \(Console\)](#)

You can use the AWS Management Console to configure and enable a virtual MFA device for your root user. To enable MFA devices for the AWS account, you must be signed in to AWS using your root user credentials.

Before you enable MFA for your root user, review your account settings and contact information to make sure that you have access to the email and phone number. If your MFA device is lost, stolen, or not working, you can still sign in as the root user by verifying your identity using that email and phone number. To learn about signing in using these alternative factors of authentication, see [What If an MFA Device Is Lost or Stops Working? \(p. 144\)](#).

To configure and enable a virtual MFA device for use with your root user (console)

1. Sign in to the AWS Management Console.
2. On the right side of the navigation bar, choose your account name, and choose **My Security Credentials**. If necessary, choose **Continue to Security Credentials**. Then expand the **Multi-Factor Authentication (MFA)** section on the page.



3. Choose **Activate MFA**.
4. In the wizard, choose **Virtual MFA device**, and then choose **Continue**.

IAM generates and displays configuration information for the virtual MFA device, including a QR code graphic. The graphic is a representation of the secret configuration key that is available for manual entry on devices that do not support QR codes.

5. Open the virtual MFA app on the device.
If the virtual MFA app supports multiple virtual MFA devices or accounts, choose the option to create a new virtual MFA device or account.
6. The easiest way to configure the app is to use the app to scan the QR code. If you cannot scan the code, you can type the configuration information manually. The QR code and secret configuration key generated by IAM are tied to your AWS account and cannot be used with a different account. They can, however, be reused to configure a new MFA device for your account in case you lose access to the original MFA device.
 - To use the QR code to configure the virtual MFA device, from the wizard, choose **Show QR code**. Then follow the app instructions for scanning the code. For example, you might need to choose the camera icon or choose a command like **Scan account barcode**, and then use the device's camera to scan the QR code.
 - In the **Manage MFA Device** wizard, choose **Show secret key**, and then type the secret key into your MFA app.

Important

Make a secure backup of the QR code or secret configuration key, or make sure that you enable multiple virtual MFA devices for your account. A virtual MFA device might become unavailable, for example, if you lose the smartphone where the virtual MFA device is hosted). If that happens, you will not be able to sign in to your account and you will have to contact customer service to remove MFA protection for the account.

The device starts generating six-digit numbers.

7. In the **Manage MFA Device** wizard, in the **MFA Code 1** box, enter the six-digit number that's currently displayed by the MFA device. Wait up to 30 seconds for the device to generate a new number, and then type the new six-digit number into the **MFA Code 2** box.

Important

Submit your request immediately after generating the codes. If you generate the codes and then wait too long to submit the request, the MFA device successfully associates with the user but the MFA device is out of sync. This happens because time-based one-time passwords (TOTP) expire after a short period of time. If this happens, you can [resync the device \(p. 138\)](#).

8. Choose **Assign MFA**, and then choose **Finish**.

The device is ready for use with AWS. For information about using MFA with the AWS Management Console, see [Using MFA Devices With Your IAM Sign-in Page \(p. 78\)](#).

[Replace or "Rotate" a Virtual MFA Device](#)

You can have only one MFA device assigned to a user at a time. If the user loses a device or needs to replace it for any reason, you must first deactivate the old device. Then you can add the new device for the user.

- To deactivate the device currently associated with another IAM user, see [Deactivating MFA Devices \(p. 142\)](#).
- To add a replacement virtual MFA device for another IAM user, follow the steps in the procedure [Enable a Virtual MFA Device for an IAM User \(Console\) \(p. 122\)](#) above.
- To add a replacement virtual MFA device for the AWS account root user, follow the steps in the procedure [Enable a Virtual MFA Device for Your AWS Account Root User \(Console\) \(p. 123\)](#) earlier in this topic.

[Enabling a U2F Security Key \(Console\)](#)

Universal 2nd Factor (U2F) security keys are a type of [MFA device \(p. 119\)](#) that you can use to protect your AWS resources. You plug your U2F security key into a USB port on your computer and enable it using the instructions that follow. After you enable it, you tap it when prompted to securely complete the sign-in process. If you already use a U2F security key with other services, and it has an [AWS supported configuration \(p. 129\)](#) (for example, the Yubikey 4 or 5 from Yubico), you can also use it with AWS. Otherwise, you need to purchase a U2F security key if you want to use U2F for MFA in AWS. For specifications and purchase information, see [Multi-Factor Authentication](#).

U2F is an open authentication standard hosted by the [FIDO Alliance](#). When you enable a U2F key in AWS, the U2F security key creates a new key pair for use with only AWS. First, you enter your credentials. When prompted, you tap the U2F security key, which responds to the authentication challenge issued by AWS. To learn more about the U2F standard, see [Universal 2nd Factor](#).

You can enable **one** MFA device (of any kind) per root user or IAM user.

Topics

- [Permissions Required \(p. 125\)](#)
- [Enable a U2F Security Key for Your Own IAM User \(Console\) \(p. 126\)](#)
- [Enable a U2F Security Key for Another IAM User \(Console\) \(p. 127\)](#)
- [Enable a U2F Security Key for the AWS Account Root User \(Console\) \(p. 128\)](#)
- [Replace a U2F Security Key \(p. 129\)](#)
- [Supported Configurations for Using U2F Security Keys \(p. 129\)](#)

Permissions Required

To manage a U2F security key for your own IAM user while protecting sensitive MFA-related actions, you must have the permissions from the following policy:

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

```
{  
    "Sid": "AllowManageOwnUserMFA",  
    "Effect": "Allow",  
    "Action": [  
        "iam:DeactivateMFADevice",  
        "iam:EnableMFADevice",  
        "iam:GetUser",  
        "iam>ListMFADevices",  
        "iam:ResyncMFADevice"  
    ],  
    "Resource": "arn:aws:iam::*:user/${aws:username}"  
},  
{  
    "Sid": "DenyAllExceptListedIfNoMFA",  
    "Effect": "Deny",  
    "NotAction": [  
        "iam:EnableMFADevice",  
        "iam:GetUser",  
        "iam>ListMFADevices",  
        "iam:ResyncMFADevice"  
    ],  
    "Resource": "arn:aws:iam::*:user/${aws:username}",  
    "Condition": {  
        "BoolIfExists": {  
            "aws:MultiFactorAuthPresent": "false"  
        }  
    }  
}  
]  
}
```

Enable a U2F Security Key for Your Own IAM User (Console)

You can enable a U2F security key for your own IAM user from the AWS Management Console only, not from the AWS CLI or AWS API.

Note

Before you can enable a U2F security key, you must have physical access to the device.

To enable a U2F security key for your own IAM user (console)

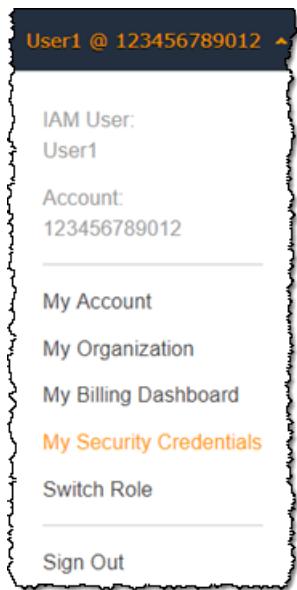
1. Use your AWS account ID or account alias, your IAM user name, and your password to sign in to the [IAM console](#).

Note

For your convenience, the AWS sign-in page uses a browser cookie to remember your IAM user name and account information. If you previously signed in as a different user, choose **Sign in to a different account** near the bottom of the page to return to the main sign-in page. From there, you can type your AWS account ID or account alias to be redirected to the IAM user sign-in page for your account.

To get your AWS account ID, contact your administrator.

2. In the navigation bar on the upper right, choose your user name, and then choose **My Security Credentials**.



3. On the **AWS IAM credentials** tab, in the **Multi-factor authentication** section, choose **Manage MFA device**.
4. In the **Manage MFA device** wizard, choose **U2F security key**, and then choose **Continue**.
5. Insert the U2F security key into your computer's USB port.



6. Tap the U2F security key, and then choose **Close** when U2F setup is complete.

The U2F security key is ready for use with AWS. For information about using MFA with the AWS Management Console, see [Using MFA Devices With Your IAM Sign-in Page \(p. 78\)](#).

[Enable a U2F Security Key for Another IAM User \(Console\)](#)

You can enable a U2F security key for another IAM user from the AWS Management Console only, not from the AWS CLI or AWS API.

To enable a U2F security key for another IAM user (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Users**.
3. Choose the name of the user for whom you want to enable MFA, and then choose the **Security credentials** tab.
4. Next to **Assigned MFA device**, choose **Manage**.
5. In the **Manage MFA device** wizard, choose **U2F security key**, and then choose **Continue**.
6. Insert the U2F security key into your computer's USB port.



7. Tap the U2F security key, and then choose **Close** when U2F setup is complete.

The U2F security key is ready for use with AWS. For information about using MFA with the AWS Management Console, see [Using MFA Devices With Your IAM Sign-in Page \(p. 78\)](#).

[Enable a U2F Security Key for the AWS Account Root User \(Console\)](#)

You can configure and enable a virtual MFA device for your root user from the AWS Management Console only, not from the AWS CLI or AWS API.

If your U2F security key is lost, stolen, or not working, you can still sign in using alternative factors of authentication. To learn about signing in using alternative factors of authentication, see [What If an MFA Device Is Lost or Stops Working? \(p. 144\)](#). To disable this feature, contact [AWS Support](#).

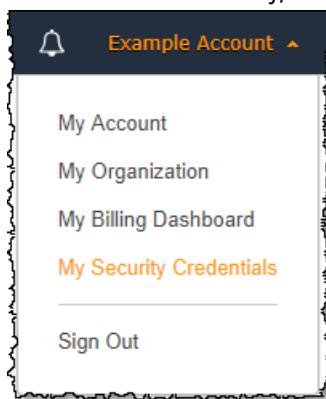
To enable the U2F key for your root user (console)

1. Use your AWS account email address and password to sign in to the [AWS Management Console](#) as the AWS account root user.

Note

If you previously signed in to the console with [IAM user](#) credentials, your browser might remember this preference and open your account-specific sign-in page. You cannot use the IAM user sign-in page to sign in with your AWS account root user credentials. If you see the IAM user sign-in page, choose **Sign-in using root user credentials** near the bottom of the page to return to the main sign-in page. From there, you can enter your AWS account email address and password.

2. On the right side of the navigation bar, choose on your account name, and then choose **My Security Credentials**. If necessary, choose **Continue to Security Credentials**.



3. Expand the **Multi-factor authentication (MFA)** section.
4. Choose **Manage MFA** or **Activate MFA**, depending on which option you chose in the preceding step.
5. In the wizard, choose **U2F security key** and then choose **Continue**.
6. Insert the U2F security key into your computer's USB port.



7. Tap the U2F security key, and then choose **Close** when U2F setup is complete.

The U2F security key is ready for use with AWS. The next time you use your root user credentials to sign in, you must tap your U2F security key to complete the sign-in process.

Replace a U2F Security Key

You can have only one MFA device (virtual, U2F security key, or hardware) assigned to a user at a time. If the user loses a U2F key or needs to replace it for any reason, you must first deactivate the old U2F key. Then you can add a new MFA device for the user.

- To deactivate the device currently associated with a user, see [Deactivating MFA Devices \(p. 142\)](#).
- To add a new U2F security for an IAM user, see [Enabling a U2F Security Key \(Console\) \(p. 125\)](#).

If you don't have access to a new U2F security key, you can enable a new virtual MFA device or hardware MFA device. See one of the following for instructions:

- [Enabling a Virtual Multi-factor Authentication \(MFA\) Device \(Console\) \(p. 122\)](#)
- [Enabling a Hardware MFA Device \(Console\) \(p. 130\)](#)

Supported Configurations for Using U2F Security Keys

You can use U2F as a multi-factor authentication (MFA) method in AWS using currently supported configurations. These include U2F devices supported by AWS and browsers that support U2F.

U2F Devices Supported by AWS

AWS currently supports U2F-compliant security devices that plug into USB ports on your computer.

Note

AWS requires access to the physical USB port on your computer to verify your U2F device. U2F MFA will not work with a virtual machine or remote connection.

For information on purchasing a supported device, see [Multi-Factor Authentication](#).

Browsers That Support U2F

The following browsers currently support the use of U2F security keys:

- Google Chrome, version 38 and later.
- Opera, version 40 and later.
- Mozilla Firefox, version 57 and later.

Note

Most Firefox versions that currently support U2F do not enable support by default.

For instructions on enabling U2F support in Firefox, see [Troubleshooting U2F Security Keys \(p. 561\)](#).

Browser Plugins

AWS currently supports only browsers that natively support the U2F standard. AWS does not support using plugins to add U2F browser support. Also note that some browser plugins are incompatible with the U2F standard and can cause unexpected results with U2F security keys.

For information on disabling browser plugins and other troubleshooting tips, see [I Can't Enable My U2F Security Key \(p. 562\)](#).

Mobile Environments

AWS does not currently support the use of U2F security keys with mobile browsers or non-USB U2F devices.

The AWS Console Mobile App does not currently support using U2F security keys for MFA.

AWS CLI and AWS API

AWS currently supports using U2F security keys only in the AWS Management Console. Using U2F security keys for MFA is not currently supported in the [AWS CLI](#) and [AWS API](#), or for access to [MFA-protected API operations \(p. 145\)](#).

Additional Resources

- For more information on using U2F security keys in AWS, see [Enabling a U2F Security Key \(Console\) \(p. 125\)](#).
- For help with troubleshooting U2F in AWS, see [Troubleshooting U2F Security Keys \(p. 561\)](#).
- For general industry information on U2F support, see [Universal 2nd Factor](#).

Enabling a Hardware MFA Device (Console)

A hardware MFA device generates a six-digit numeric code based upon a time-synchronized one-time password algorithm. The user must type a valid code from the device when prompted during the sign-in process. Each MFA device assigned to a user must be unique; a user cannot type a code from another user's device to be authenticated.

Hardware MFA devices and [U2F security keys \(p. 125\)](#) are both physical devices that you purchase. The difference is that hardware MFA devices generate a code that you view and then enter when prompted when signing it to AWS. With a U2F security key, you don't see or type an authentication code. Instead, the U2F security key generates a response without presenting it to the user and the service validates it. For specifications and purchase information for both device types, see [Multi-Factor Authentication](#).

You can enable a hardware MFA device for an IAM user from the AWS Management Console, the command line, or the IAM API. To enable an MFA device for your AWS account root user, see [Enable a Hardware MFA Device for the AWS Account Root User \(Console\) \(p. 133\)](#).

You can enable **one** MFA device (of any kind) per root user or IAM user.

Note

If you want to enable the device from the command line, use `iam-userenablemfadevice aws iam enable-mfa-device`. To enable the MFA device with the IAM API, use the `EnableMFADevice` operation.

Topics

- [Permissions Required \(p. 131\)](#)
- [Enable a Hardware MFA Device for Your Own IAM User \(Console\) \(p. 131\)](#)
- [Enable a Hardware MFA Device for Another IAM User \(Console\) \(p. 132\)](#)

- [Enable a Hardware MFA Device for the AWS Account Root User \(Console\) \(p. 133\)](#)
- [Replace or "Rotate" a Physical MFA Device \(p. 134\)](#)

Permissions Required

To manage a hardware MFA device for your own IAM user while protecting sensitive MFA-related actions, you must have the permissions from the following policy:

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "AllowManageOwnUserMFA",  
            "Effect": "Allow",  
            "Action": [  
                "iam:DeactivateMFADevice",  
                "iam:EnableMFADevice",  
                "iam:GetUser",  
                "iam>ListMFADevices",  
                "iam:ResyncMFADevice"  
            ],  
            "Resource": "arn:aws:iam::*:user/${aws:username}"  
        },  
        {  
            "Sid": "DenyAllExceptListedIfNoMFA",  
            "Effect": "Deny",  
            "NotAction": [  
                "iam:EnableMFADevice",  
                "iam:GetUser",  
                "iam>ListMFADevices",  
                "iam:ResyncMFADevice"  
            ],  
            "Resource": "arn:aws:iam::*:user/${aws:username}",  
            "Condition": {  
                "BoolIfExists": {  
                    "aws:MultiFactorAuthPresent": "false"  
                }  
            }  
        }  
    ]  
}
```

Enable a Hardware MFA Device for Your Own IAM User (Console)

You can enable your own hardware MFA device from the AWS Management Console.

Note

Before you can enable a hardware MFA device, you must have physical access to the device.

To enable a hardware MFA device for your own IAM user (console)

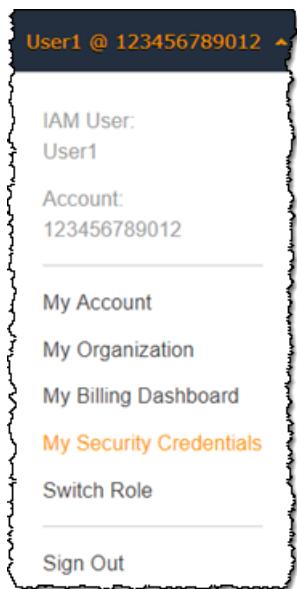
1. Use your AWS account ID or account alias, your IAM user name, and your password to sign in to the [IAM console](#).

Note

For your convenience, the AWS sign-in page uses a browser cookie to remember your IAM user name and account information. If you previously signed in as a different user, choose **Sign in to a different account** near the bottom of the page to return to the main sign-in page. From there, you can type your AWS account ID or account alias to be redirected to the IAM user sign-in page for your account.

To get your AWS account ID, contact your administrator.

2. In the navigation bar on the upper right, choose your user name, and then choose **My Security Credentials**.



3. On the **AWS IAM credentials** tab, in the **Multi-factor authentication** section, choose **Manage MFA device**.
4. In the **Manage MFA device** wizard, choose **Hardware MFA device** and then choose **Continue**.
5. Type the device serial number. The serial number is usually on the back of the device.
6. In the **MFA code 1** box, type the six-digit number displayed by the MFA device. You might need to press the button on the front of the device to display the number.



7. Wait 30 seconds while the device refreshes the code, and then type the next six-digit number into the **MFA code 2** box. You might need to press the button on the front of the device again to display the second number.
8. Choose **Assign MFA**.

Important

Submit your request immediately after generating the authentication codes. If you generate the codes and then wait too long to submit the request, the MFA device successfully associates with the user but the MFA device becomes out of sync. This happens because time-based one-time passwords (TOTP) expire after a short period of time. If this happens, you can [resync the device \(p. 138\)](#).

The device is ready for use with AWS. For information about using MFA with the AWS Management Console, see [Using MFA Devices With Your IAM Sign-in Page \(p. 78\)](#).

[Enable a Hardware MFA Device for Another IAM User \(Console\)](#)

You can enable a hardware MFA device for another IAM user from the AWS Management Console.

To enable a hardware MFA device for another IAM user (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.

2. In the navigation pane, choose **Users**.
3. Choose the name of the user for whom you want to enable MFA, and then choose the **Security credentials** tab.
4. Next to **Assigned MFA device**, choose **Manage**.
5. In the **Manage MFA device** wizard, choose **Hardware MFA device** and then choose **Continue**.
6. Type the device serial number. The serial number is usually on the back of the device.
7. In the **MFA code 1** box, type the six-digit number displayed by the MFA device. You might need to press the button on the front of the device to display the number.



8. Wait 30 seconds while the device refreshes the code, and then type the next six-digit number into the **MFA code 2** box. You might need to press the button on the front of the device again to display the second number.
9. Choose **Assign MFA**.

Important

Submit your request immediately after generating the authentication codes. If you generate the codes and then wait too long to submit the request, the MFA device successfully associates with the user but the MFA device becomes out of sync. This happens because time-based one-time passwords (TOTP) expire after a short period of time. If this happens, you can [resync the device \(p. 138\)](#).

The device is ready for use with AWS. For information about using MFA with the AWS Management Console, see [Using MFA Devices With Your IAM Sign-in Page \(p. 78\)](#).

[Enable a Hardware MFA Device for the AWS Account Root User \(Console\)](#)

You can configure and enable a virtual MFA device for your root user from the AWS Management Console only, not from the AWS CLI or AWS API.

If your MFA device is lost, stolen, or not working, you can still sign in using alternative factors of authentication. If you can't sign in with your MFA device, you can sign in by verifying your identity using the email and phone that are registered with your account. Before you enable MFA for your root user, review your account settings and contact information to make sure that you have access to the email and phone number. To learn about signing in using alternative factors of authentication, see [What If an MFA Device Is Lost or Stops Working? \(p. 144\)](#). To disable this feature, contact [AWS Support](#).

Note

You might see different text, such as **Sign in using MFA** and **Troubleshoot your authentication device**. However, the same features are provided. In either case, if you cannot verify your account email address and phone number using alternative factors of authentication, contact [AWS Support](#) to deactivate your MFA setting.

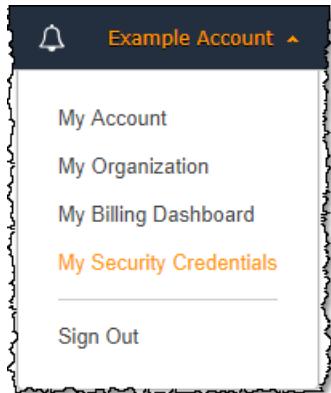
To enable the MFA device for your root user (console)

1. Use your AWS account email address and password to sign in to the [AWS Management Console](#) as the AWS account root user.

Note

If you previously signed in to the console with [IAM user](#) credentials, your browser might remember this preference and open your account-specific sign-in page. You cannot use the IAM user sign-in page to sign in with your AWS account root user credentials. If you see the IAM user sign-in page, choose **Sign-in using root user credentials** near the bottom of the page to return to the main sign-in page. From there, you can enter your AWS account email address and password.

2. On the right side of the navigation bar, choose on your account name, and then choose **My Security Credentials**. If necessary, choose **Continue to Security Credentials**.



3. Expand the **Multi-factor authentication (MFA)** section.
4. Choose **Manage MFA** or **Activate MFA**, depending on which option you chose in the preceding step.
5. In the wizard, choose **Hardware MFA device** and then choose **Continue**.
6. In the **Serial number** box, type the serial number that is found on the back of the MFA device.
7. In the **MFA code 1** box, type the six-digit number displayed by the MFA device. You might need to press the button on the front of the device to display the number.



8. Wait 30 seconds while the device refreshes the code, and then type the next six-digit number into the **MFA code 2** box. You might need to press the button on the front of the device again to display the second number.
9. Choose **Assign MFA**. The MFA device is now associated with the AWS account.

Important

Submit your request immediately after generating the authentication codes. If you generate the codes and then wait too long to submit the request, the MFA device successfully associates with the user but the MFA device becomes out of sync. This happens because time-based one-time passwords (TOTP) expire after a short period of time. If this happens, you can [resync the device \(p. 138\)](#).

The next time you use your root user credentials to sign in, you must type a code from the MFA device.

Replace or "Rotate" a Physical MFA Device

You can have only one MFA device assigned to a user at a time. If the user loses a device or needs to replace it for any reason, you must first deactivate the old device. Then you can add the new device for the user.

- To deactivate the device currently associated with a user, see [Deactivating MFA Devices \(p. 142\)](#).
- To add a replacement hardware MFA device for an IAM user, follow the steps in the procedure [Enable a Hardware MFA Device for Another IAM User \(Console\) \(p. 132\)](#) earlier in this topic.
- To add a replacement virtual MFA device for the AWS account root user, follow the steps in the procedure [Enable a Hardware MFA Device for the AWS Account Root User \(Console\) \(p. 133\)](#) earlier in this topic.

PREVIEW – Enabling SMS Text Message MFA Devices

AWS will soon end support for SMS multi-factor authentication (MFA). We are not allowing new customers to preview this feature. We recommend that existing customers switch to one of the following alternative methods of MFA:

- A [virtual \(software-based\) \(p. 122\)](#) MFA device
- A [U2F security key \(p. 125\)](#)
- A [hardware-based \(p. 130\)](#) MFA device

Tip

You can view users in your account with an assigned SMS MFA device. In the IAM console, choose **Users** from the navigation pane, and look for users with **SMS** in the **MFA** column of the table.

An SMS (short message service) MFA device can be any mobile device with a phone number that can receive standard [SMS text messages](#). When an MFA code is needed, AWS sends it to the phone number that is configured for the IAM user.

Note

SMS MFA can be used only with IAM users. It cannot be used with the AWS account root user. To protect the root user with MFA, you must use a virtual MFA device, U2F security key, or hardware MFA device.

Enable an SMS MFA Device for an IAM User (Console)

You can use IAM in the AWS Management Console to configure an IAM user with a phone number to enable SMS MFA.

Note

Currently, you can manage SMS MFA only in the AWS Management Console.

To enable SMS MFA for an IAM user (console)

1. Use your AWS account ID or account alias, your IAM user name, and your password to sign in to the [IAM console](#).

Note

For your convenience, the AWS sign-in page uses a browser cookie to remember your IAM user name and account information. If you previously signed in as a different user, choose **Sign in to a different account** near the bottom of the page to return to the main sign-in page. From there, you can type your AWS account ID or account alias to be redirected to the IAM user sign-in page for your account.

2. In the navigation pane, choose **Users**.
3. In the **User Name** list, choose the name (not the check box) of the intended MFA user.
4. Choose the **Security credentials** tab. Next to **Assigned MFA device**, choose **Manage**.
5. In the **Manage MFA Device** wizard, choose **An SMS MFA device**, and then choose **Continue**.
6. Type the phone number to which you want to send MFA codes for this IAM user, and then choose **Continue**.
7. A six-digit authentication code is immediately sent to the specified phone number for verification. Type the six-digit code and then choose **Continue**. If the code does not arrive in a reasonable amount of time, choose **Resend Code**. Note that SMS is not a service with a guaranteed delivery time.
8. If AWS successfully verifies the code, the wizard ends. Otherwise, choose **Finish** to close the wizard.

Change the Phone Number for SMS MFA for an IAM User

To change the phone number of the SMS MFA device assigned to an IAM user, you must delete the current MFA device. Then create a new device with the new phone number. To learn how to delete a device, see [Deactivating MFA Devices \(p. 142\)](#).

Enabling and Managing Virtual MFA Devices (AWS CLI or AWS API)

You can use AWS CLI commands or AWS API operations to enable a virtual MFA device for an IAM user. You cannot enable an MFA device for the AWS account root user with the AWS CLI, AWS API, Tools for Windows PowerShell, or any other command line tool. However, you can use the AWS Management Console to enable an MFA device for the root user.

When you enable an MFA device from the AWS Management Console, the console performs multiple steps for you. If you instead create a virtual device using the AWS CLI, Tools for Windows PowerShell, or AWS API, then you must perform the steps manually and in the correct order. For example, to create a virtual MFA device, you must create the IAM object and extract the code as either a string or a QR code graphic. Then you must sync the device and associate it with an IAM user. See the **Examples** section of [New-IAMVirtualMFADevice](#) for more details. For a physical device, you skip the creation step and go directly to syncing the device and associating it with the user.

To create the virtual device entity in IAM to represent a virtual MFA device

These commands provide an ARN for the device that is used in place of a serial number in many of the following commands.

- AWS CLI: `aws iam create-virtual-mfa-device`
- AWS API: `CreateVirtualMFADevice`

To enable an MFA device for use with AWS

These commands synchronize the device with AWS and associate it with a user or the root user. If the device is virtual, use the ARN of the virtual device as the serial number.

Important

Submit your request immediately after generating the authentication codes. If you generate the codes and then wait too long to submit the request, the MFA device successfully associates with the user but the MFA device becomes out of sync. This happens because time-based one-time passwords (TOTP) expire after a short period of time. If this happens, you can resynchronize the device using the commands described below.

- AWS CLI: `aws iam enable-mfa-device`
- AWS API: `EnableMFADevice`

To deactivate a device

Use these commands to disassociate the device from the user and deactivate it. If the device is virtual, use the ARN of the virtual device as the serial number. You must also separately delete the virtual device entity.

- AWS CLI: `aws iam deactivate-mfa-device`
- AWS API: `DeactivateMFADevice`

To list virtual MFA device entities

Use these commands to list virtual MFA device entities.

- AWS CLI: `aws iam list-virtual-mfa-devices`

- AWS API: [ListVirtualMFADevices](#)

To resynchronize an MFA device

Use these commands if the device is generating codes that are not accepted by AWS. If the device is virtual, use the ARN of the virtual device as the serial number.

- AWS CLI: `aws iam resync-mfa-device`
- AWS API: [ResyncMFADevice](#)

To delete a virtual MFA device entity in IAM

After the device is disassociated from the user, you can delete the device entity.

- AWS CLI: `aws iam delete-virtual-mfa-device`
- AWS API: [DeleteVirtualMFADevice](#)

To recover a virtual MFA device that is lost or not working

Sometimes, an IAM user's device that hosts the virtual MFA app is lost, replaced, or not working. When this happens, the user can't recover it on their own. IAM users must contact an administrator to deactivate the device. For more information, see [What If an MFA Device Is Lost or Stops Working? \(p. 144\)](#).

Checking MFA Status

Use the IAM console to check whether an AWS account root user or IAM user has a valid MFA device enabled.

To check the MFA status of a root user

1. Sign in to the AWS Management Console with your root user credentials and then open the IAM console at <https://console.aws.amazon.com/iam/>.
2. Check under **Security Status** to see whether MFA is enabled or disabled. If MFA has not been activated, an alert symbol ( A) is displayed next to **Activate MFA on your root user**.

If you want to enable MFA for the account, see one of the following:

- [Enable a Virtual MFA Device for Your AWS Account Root User \(Console\) \(p. 123\)](#)
- [Enable a U2F Security Key for the AWS Account Root User \(Console\) \(p. 128\)](#)
- [Enable a Hardware MFA Device for the AWS Account Root User \(Console\) \(p. 133\)](#)

To check the MFA status of IAM users

1. Open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Users**.
3. If necessary, add the **MFA** column to the users table by completing the following steps:
 - a. Above the table on the far right, choose the settings icon ().
 - b. In **Manage Columns**, select **MFA**.
 - c. (Optional) Clear the check box for any column headings that you do not want to appear in the users table.

- d. Choose **Close** to return to the list of users.
4. The **MFA** column tells you about the MFA device that is enabled. If no MFA device is active for the user, the console displays **Not enabled**. If the user has an MFA device enabled, the **MFA** column shows the type of device that is enabled with a value of **Virtual**, **U2F Security Key**, **Hardware**, or **SMS**.
5. To view additional information about the MFA device for a user, choose the name of the user whose MFA status you want to check. Then choose the **Security credentials** tab.
6. If no MFA device is active for the user, the console displays **No** next to **Assigned MFA device**. If the user has an MFA device enabled, the **Assigned MFA device** item shows a value for the device:
 - The device serial number of a hardware device (usually the number from the back of the device), such as GAHT12345678
 - The ARN in AWS for an SMS device, such as `arn:aws:iam::123456789012:sms-mfa/username`
 - The ARN in AWS for a virtual device, such as `arn:aws:iam::123456789012:mfa/username`

If you want to change the current setting, choose **Manage** next to **Assigned MFA Device**.

For more information on enabling MFA, see the following:

- [Enabling a Virtual Multi-factor Authentication \(MFA\) Device \(Console\) \(p. 122\)](#)
- [Enabling a U2F Security Key \(Console\) \(p. 125\)](#)
- [Enabling a Hardware MFA Device \(Console\) \(p. 130\)](#)
- [PREVIEW – Enabling SMS Text Message MFA Devices \(p. 135\)](#)

Resynchronizing Virtual and Hardware MFA Devices

You can use AWS to resynchronize your virtual and hardware multi-factor authentication (MFA) devices. If your device is not synchronized when you try to use it, the sign-in attempt fails and IAM prompts you to resynchronize the device.

Note

U2F security keys do not go out of sync. If a U2F security key is lost or broken, you can deactivate it. For instructions on deactivating any MFA device type, see [To deactivate an MFA device for another IAM user \(console\) \(p. 142\)](#).

As an AWS administrator, you can resynchronize your IAM users' virtual and hardware MFA devices if they get out of synchronization.

If your AWS account root user MFA device is not working, you can resynchronize your device using the IAM console with or without completing the sign-in process.

Topics

- [Permissions Required \(p. 138\)](#)
- [Resynchronizing Virtual and Hardware MFA Devices \(IAM Console\) \(p. 139\)](#)
- [Resynchronizing Virtual and Hardware MFA Devices \(AWS CLI\) \(p. 141\)](#)
- [Resynchronizing Virtual and Hardware MFA Devices \(AWS API\) \(p. 142\)](#)

Permissions Required

To resynchronize virtual or hardware MFA devices for your own IAM user, you must have the permissions from the following policy: This policy does not allow you to create or deactivate a device.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "AllowListActions",  
            "Effect": "Allow",  
            "Action": [  
                "iam>ListVirtualMFADevices"  
            ],  
            "Resource": "*"  
        },  
        {  
            "Sid": "AllowUserToViewAndManageTheirOwnUserMFA",  
            "Effect": "Allow",  
            "Action": [  
                "iam>ListMFADevices",  
                "iam>ResyncMFADevice"  
            ],  
            "Resource": "arn:aws:iam::*:user/${aws:username}"  
        },  
        {  
            "Sid": "BlockAllExceptListedIfNoMFA",  
            "Effect": "Deny",  
            "NotAction": [  
                "iam>ListMFADevices",  
                "iam>ListVirtualMFADevices",  
                "iam>ResyncMFADevice"  
            ],  
            "Resource": "*",  
            "Condition": {  
                "BoolIfExists": {  
                    "aws:MultiFactorAuthPresent": "false"  
                }  
            }  
        }  
    ]  
}
```

Resynchronizing Virtual and Hardware MFA Devices (IAM Console)

You can use the IAM console to resynchronize virtual and hardware MFA devices.

To resynchronize a virtual or hardware MFA device for your own IAM user (console)

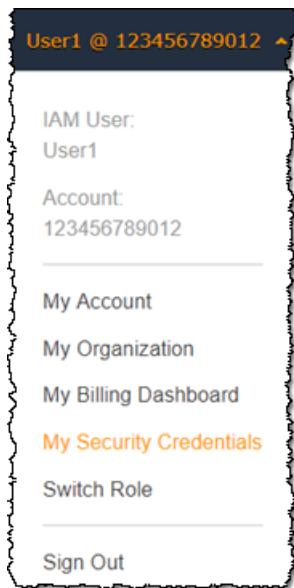
1. Use your AWS account ID or account alias, your IAM user name, and your password to sign in to the [IAM console](#).

Note

For your convenience, the AWS sign-in page uses a browser cookie to remember your IAM user name and account information. If you previously signed in as a different user, choose **Sign in to a different account** near the bottom of the page to return to the main sign-in page. From there, you can type your AWS account ID or account alias to be redirected to the IAM user sign-in page for your account.

To get your AWS account ID, contact your administrator.

2. In the navigation bar on the upper right, choose your user name, and then choose **My Security Credentials**.



3. On the **AWS IAM credentials** tab, in the **Multi-factor authentication** section, choose **Manage MFA device**.
4. In the **Manage MFA device** wizard, choose **Resync**, and then choose **Continue**.
5. Type the next two sequentially generated codes from the device into **MFA code 1** and **MFA code 2**. Then choose **Continue**.

Important

Submit your request immediately after generating the codes. If you generate the codes and then wait too long to submit the request, the request appears to work but the device remains out of sync. This happens because time-based one-time passwords (TOTP) expire after a short period of time.

To resynchronize a virtual or hardware MFA device for another IAM user (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Users**, and then choose the name of the user whose MFA device needs to be resynchronized.
3. Choose the **Security credentials** tab. Next to **Assigned MFA device**, choose **Manage**.
4. In the **Manage MFA device** wizard, choose **Resync**, and then choose **Continue**.
5. Type the next two sequentially generated codes from the device into **MFA code 1** and **MFA code 2**. Then choose **Continue**.

Important

Submit your request immediately after generating the codes. If you generate the codes and then wait too long to submit the request, the request appears to work but the device remains out of sync. This happens because time-based one-time passwords (TOTP) expire after a short period of time.

To resynchronize your root user MFA before signing in (console)

1. On the **Amazon Web Services Sign In With Authentication Device** page, choose **Having problems with your authentication device? Click here**.

Note

You might see different text, such as **Sign in using MFA** and **Troubleshoot your authentication device**. However, the same features are provided.

2. In the **Re-Sync With Our Servers** section, type the next two sequentially generated codes from the device into **MFA code 1** and **MFA code 2**. Then choose **Re-sync authentication device**.
3. If necessary, type your password again and choose **Sign in**. Then complete the sign-in using your MFA device.

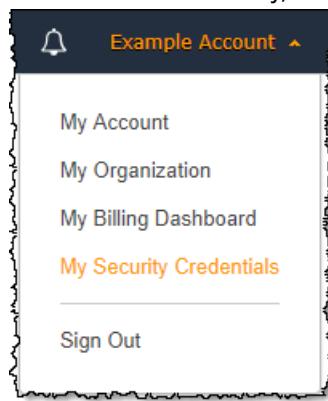
To resynchronize your root user MFA device after signing in (console)

1. Use your AWS account email address and password to sign in to the [AWS Management Console](#) as the AWS account root user.

Note

If you previously signed in to the console with [IAM user](#) credentials, your browser might remember this preference and open your account-specific sign-in page. You cannot use the IAM user sign-in page to sign in with your AWS account root user credentials. If you see the IAM user sign-in page, choose **Sign-in using root user credentials** near the bottom of the page to return to the main sign-in page. From there, you can enter your AWS account email address and password.

2. On the right side of the navigation bar, choose on your account name, and then choose **My Security Credentials**. If necessary, choose **Continue to Security Credentials**.



3. Expand the **Multi-factor authentication (MFA)** section on the page.
4. Next to your active MFA device, choose **Resync**.
5. In the **Manage MFA device** dialog box, type the next two sequentially generated codes from the device into **MFA code 1** and **MFA code 2**. Then choose **Continue**.

Important

Submit your request immediately after generating the codes. If you generate the codes and then wait too long to submit the request, the MFA device is successfully associated with the user, but the MFA device is out of sync. This happens because time-based one-time passwords (TOTP) expire after a short period of time.

Resynchronizing Virtual and Hardware MFA Devices (AWS CLI)

You can resynchronize virtual and hardware MFA devices from the AWS CLI.

To resynchronize a virtual or hardware MFA device for an IAM user (AWS CLI)

At a command prompt, issue the `aws iam resync-mfa-device` command:

- Virtual MFA device: Specify Amazon Resource Name (ARN) of device as the serial number.

```
$ aws iam resync-mfa-device --user-name Richard --serial-number  
arn:aws:iam::123456789012:mfa/RichardsMFA --authentication-code-1 123456 --  
authentication-code-2 987654
```

- Hardware MFA device: Specify hardware device's serial number as serial number. The format is vendor-specific. For example, you can purchase a gemalto token from Amazon. Its serial number is typically four letters followed by four numbers.

```
$ aws iam resync-mfa-device --user-name Richard --serial-number ABCD12345678 --  
authentication-code-1 123456 --authentication-code-2 987654
```

Important

Submit your request immediately after generating the codes. If you generate the codes and then wait too long to submit the request, the request fails because the codes expire after a short time.

Resynchronizing Virtual and Hardware MFA Devices (AWS API)

IAM has an API call that performs synchronization. In this case, we recommend that you give your virtual and hardware MFA device users permission to access this API call. Then build a tool based on that API call so your users can resynchronize their devices whenever they need to.

To resynchronize a virtual or hardware MFA device for an IAM user (AWS API)

- Send the [ResyncMFADevice](#) request.

Deactivating MFA Devices

If you have trouble signing in with a multi-factor authentication (MFA) device as an IAM user, contact your administrator for help.

As an administrator, you can deactivate the device for another IAM user. This allows the user to sign in without using MFA. You might do this as a temporary solution while the MFA device is replaced, or if the device is temporarily unavailable. However, we recommend that you enable a new device for the user as soon as possible. To learn how to enable a new MFA device, see [the section called “Enabling MFA Devices” \(p. 120\)](#).

Note

If you use the API or AWS CLI to delete a user from your AWS account, you must deactivate or delete the user's MFA device. You make this change as part of the process of removing the user. For more information about deleting users, see [Managing IAM Users \(p. 92\)](#).

Topics

- [Deactivating MFA Devices \(Console\) \(p. 142\)](#)
- [Deactivating MFA Devices \(AWS CLI\) \(p. 143\)](#)
- [Deactivating MFA Devices \(AWS API\) \(p. 143\)](#)

Deactivating MFA Devices (Console)

To deactivate an MFA device for another IAM user (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.

2. In the navigation pane, choose **Users**.
3. To deactivate the MFA device for a user, choose the name of the user whose MFA you want to remove.
4. Choose the **Security credentials** tab. Next to **Assigned MFA device**, choose **Manage**.
5. In the **Manage MFA device** wizard, choose **Deactivate MFA device**, and then choose **Continue**.

The device is removed from AWS. It cannot be used to sign in or authenticate requests until it is reactivated and associated with an AWS user or AWS account root user.

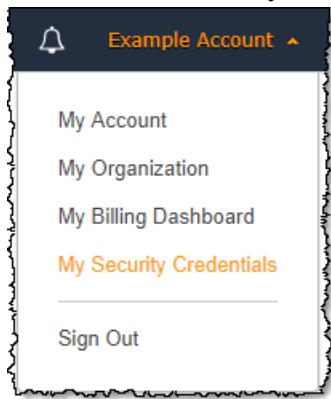
To deactivate the MFA device for your AWS account root user (console)

1. Use your AWS account email address and password to sign in to the [AWS Management Console](#) as the AWS account root user.

Note

If you previously signed in to the console with [IAM user](#) credentials, your browser might remember this preference and open your account-specific sign-in page. You cannot use the IAM user sign-in page to sign in with your AWS account root user credentials. If you see the IAM user sign-in page, choose **Sign-in using root user credentials** near the bottom of the page to return to the main sign-in page. From there, you can enter your AWS account email address and password.

2. On the right side of the navigation bar, choose on your account name, and then choose **My Security Credentials**. If necessary, choose **Continue to Security Credentials**.



3. Expand the **Multi-factor authentication (MFA)** section.
4. In the row for the MFA device that you want to deactivate, choose **Deactivate**.

The MFA device is deactivated for the AWS account.

Deactivating MFA Devices (AWS CLI)

To deactivate an MFA device for an IAM user (AWS CLI)

- Run this command: `aws iam deactivate-mfa-device`

Deactivating MFA Devices (AWS API)

To deactivate an MFA device for an IAM user (AWS API)

- Call this operation: `DeactivateMFADevice`

What If an MFA Device Is Lost or Stops Working?

If your [virtual MFA device \(p. 122\)](#) or [hardware MFA device \(p. 130\)](#) appears to be functioning properly, but you cannot use it to access your AWS resources, it might be out of synchronization with AWS. For information about synchronizing a virtual MFA device or hardware MFA device, see [Resynchronizing Virtual and Hardware MFA Devices \(p. 138\)](#). U2F security keys ([p. 125](#)) do not go out of sync.

If your AWS account root user [multi-factor authentication \(MFA\) device \(p. 119\)](#) is lost, damaged, or not working, you can recover access to your account. IAM users must contact an administrator to deactivate the device.

Recovering a Root User MFA Device

If your AWS account root user [multi-factor authentication \(MFA\) device \(p. 119\)](#) is lost, damaged, or not working, you can sign in using alternative methods of authentication. This means that if you can't sign in with your MFA device, you can sign in by verifying your identity using the email and phone that are registered with your account.

Before you sign in as a root user using alternative factors of authentication, make sure that you have access to the email and phone number that are associated with your account. If you no longer have access to the email or phone, you must contact [AWS Support](#). They can disable your MFA device so that you can sign in and add a new one.

To sign in using alternative factors of authentication as an AWS account root user

1. Use your AWS account email address and password to sign in to the [AWS Management Console](#) as the [AWS account root user](#).
2. On the [Amazon Web Services Sign In Using MFA](#) page, choose **Having problems with your authentication device? Click here.**

Note

You might see different text, such as **Sign in using MFA** and **Troubleshoot your authentication device**. However, the same features are provided. In either case, if you cannot verify your account email address and phone number using alternative factors of authentication, contact [AWS Support](#) to deactivate your MFA device.

3. If required, type your password again and choose **Sign in**.
4. In the **Sign In Using Alternative Factors of Authentication** section, choose **Sign in using alternative factors**.
5. To authenticate your account by verifying the email address, choose **Send verification email**.
6. Check the email that is associated with your AWS account for a message from Amazon Web Services (no-reply-aws@amazon.com). Follow the directions in the email.

If you don't see the email in your account, check your spam folder, or return to your browser and choose **Resend the email**.

7. After you verify your email address, you can continue authenticating your account. To verify your phone number, choose **Call me now**.
8. Answer the call from AWS and, when prompted, enter the 6-digit number from the AWS website on your phone keypad.

If you don't receive a call from AWS, choose **Sign in** to sign in to the console again and start over. Or choose [AWS Support](#) to contact support for help.

9. After you verify your phone number, you can sign in to your account by choosing **Sign in to the console**.
10. The next step varies depending on the type of MFA you are using:

- For a virtual MFA device, remove the account from your device. Then go to the [AWS Security Credentials](#) page and delete the old MFA virtual device entity before you create a new one.
- For a U2F security key, go to the [AWS Security Credentials](#) page and deactivate the old U2F key before enabling a new one.
- For a hardware MFA device, contact the third-party provider for help fixing or replacing the device. You can continue to sign in using alternative factors of authentication until you receive your new device. After you have the new hardware MFA device, go to the [AWS Security Credentials](#) page and delete the old MFA hardware device entity before you create a new one.

Note

You don't have to replace a lost or stolen MFA device with the same type of device. For example, if you break your U2F security key and order a new one, you can use virtual MFA or a hardware MFA device until you receive a new U2F security key.

11. If your MFA device is missing or stolen, also [change your AWS password \(p. 100\)](#) in case an attacker has stolen the authentication device and might also have your current password.

Recovering an IAM User MFA Device

If you are an IAM user and your device is lost or stops working, you can't recover it by yourself. You must contact an administrator to deactivate the device. Then you can enable a new device.

To get help for an MFA device as an IAM user

1. Contact the AWS administrator or other person who gave you the user name and password for the IAM user. The administrator must deactivate the MFA device as described in [Deactivating MFA Devices \(p. 142\)](#) so that you can sign in.
2. The next step varies depending on the type of MFA you are using:
 - For a virtual MFA device, remove the account from your device. Then enable the virtual device as described in [Enabling a Virtual Multi-factor Authentication \(MFA\) Device \(Console\) \(p. 122\)](#).
 - For a U2F security key, contact the third-party provider for help replacing the device. When you receive the new U2F security key, enable it as described in [Enabling a U2F Security Key \(Console\) \(p. 125\)](#).
 - For a hardware MFA device, contact the third-party provider for help fixing or replacing the device. After you have the new physical MFA device, enable the device as described in [Enabling a Hardware MFA Device \(Console\) \(p. 130\)](#).

Note

You don't have to replace a lost or stolen MFA device with the same type of device. For example, if you break your U2F security key and order a new one, you can use virtual MFA or a hardware MFA device until you receive a new U2F security key.

3. If your MFA device is missing or stolen, also [change your password \(p. 109\)](#) in case an attacker has stolen the authentication device and might also have your current password.

Configuring MFA-Protected API Access

With IAM policies, you can specify which API operations a user is allowed to call. In some cases, you might want the additional security of requiring users to be authenticated with AWS multi-factor authentication (MFA) before you allow them to perform particularly sensitive actions.

For example, you might have a policy that allows a user to perform the Amazon EC2 `RunInstances`, `DescribeInstances`, and `StopInstances` actions. But you might want to restrict a destructive action

like `TerminateInstances` and ensure that users can perform that action only if they authenticate with an AWS MFA device.

Topics

- [Overview \(p. 146\)](#)
- [Scenario: MFA Protection for Cross-Account Delegation \(p. 148\)](#)
- [Scenario: MFA Protection for Access to API Operations in the Current Account \(p. 149\)](#)
- [Scenario: MFA Protection for Resources That Have Resource-based Policies \(p. 150\)](#)

Overview

Adding MFA protection to API operations involves these tasks:

1. The administrator configures an AWS MFA device for each user who needs to make API requests that require MFA authentication. This process is described at [Enabling MFA Devices \(p. 120\)](#).
2. The administrator creates policies for the users that include a `Condition` element that checks whether the user authenticated with an AWS MFA device.
3. The user calls one of the AWS STS API operations that support the MFA parameters `AssumeRole` or `GetSessionToken`, depending on the scenario for MFA protection, as explained later. As part of the call, the user includes the device identifier for the device that's associated with the user. The user also includes the time-based one-time password (TOTP) that the device generates. In either case, the user gets back temporary security credentials that the user can then use to make additional requests to AWS.

Note

MFA protection for a service's API operations is available only if the service supports temporary security credentials. For a list of these services, see [Using Temporary Security Credentials to Access AWS](#).

If authorization fails, AWS returns an access denied error message (as it does for any unauthorized access). With MFA-protected API policies in place, AWS denies access to the API operations specified in the policies if the user attempts to call an API operation without valid MFA authentication. The operation is also denied if the time stamp of the request for the API operation is outside of the allowed range specified in the policy. The user must be reauthenticated with MFA by requesting new temporary security credentials with an MFA code and device serial number.

IAM Policies with MFA Conditions

Policies with MFA conditions can be attached to the following:

- An IAM user or group
- A resource such as an Amazon S3 bucket, Amazon SQS queue, or Amazon SNS topic
- The trust policy of an IAM role that can be assumed by a user

You can use an MFA condition in a policy to check the following properties:

- Existence—To simply verify that the user did authenticate with MFA, check that the `aws:MultiFactorAuthPresent` key is `True` in a `Bool` condition. The key is only present when the user authenticates with short-term credentials. Long-term credentials, such as access keys, do not include this key.
- Duration—if you want to grant access only within a specified time after MFA authentication, use a numeric condition type to compare the `aws:MultiFactorAuthAge` key's age to a value (such as 3600 seconds). Note that the `aws:MultiFactorAuthAge` key is not present if MFA was not used.

The following example shows the trust policy of an IAM role that includes an MFA condition to test for the existence of MFA authentication. With this policy, users from the AWS account specified in the Principal element (replace ACCOUNT-B-ID with a valid AWS account ID) can assume the role that this policy is attached to. However such users can only assume the role if the user is authenticated using MFA.

```
{  
    "Version": "2012-10-17",  
    "Statement": {  
        "Effect": "Allow",  
        "Principal": {"AWS": "ACCOUNT-B-ID"},  
        "Action": "sts:AssumeRole",  
        "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}  
    }  
}
```

For more information on the condition types for MFA, see [AWS Global Condition Context Keys \(p. 665\)](#), [Numeric Condition Operators \(p. 616\)](#), and [Condition Operator to Check Existence of Condition Keys \(p. 621\)](#).

Choosing Between GetSessionToken and AssumeRole

AWS STS provides two API operations that let users pass MFA information: `GetSessionToken` and `AssumeRole`. The API operation that the user calls to get temporary security credentials depends on which of the following scenarios applies.

Use `GetSessionToken` for the following scenarios:

- Call API operations that access resources in the same AWS account as the IAM user who makes the request. Note that temporary credentials from a `GetSessionToken` request can access IAM and AWS STS API operations *only* if you include MFA information in the request for credentials. Because temporary credentials returned by `GetSessionToken` include MFA information, you can check for MFA in individual API operations made by the credentials.
- Access to resources that are protected with resource-based policies that include an MFA condition.

The purpose of the `GetSessionToken` operation is to authenticate the user using MFA. You cannot use policies to control authentication operations.

Use `AssumeRole` for the following scenarios:

- Call API operations that access resources in the same or a different AWS account. The API calls can include any IAM or AWS STS API. Note that to protect access you enforce MFA at the time when the user assumes the role. The temporary credentials returned by `AssumeRole` do not include MFA information in the context, so you cannot check individual API operations for MFA. This is why you must use `GetSessionToken` to restrict access to resources protected by resource-based policies.

Details about how to implement these scenarios are provided later in this document.

Important Points About MFA-Protected API Access

It's important to understand the following aspects of MFA protection for API operations:

- MFA protection is available only with temporary security credentials, which must be obtained with `AssumeRole` or `GetSessionToken`.
- You cannot use MFA-protected API access with AWS account root user credentials.
- You cannot use MFA-protected API access with U2F security keys.
- Federated users cannot be assigned an MFA device for use with AWS services, so they cannot access AWS resources controlled by MFA. (See next point.)

- Other AWS STS API operations that return temporary credentials do not support MFA. For `AssumeRoleWithWebIdentity` and `AssumeRoleWithSAML`, the user is authenticated by an external provider and AWS cannot determine whether that provider required MFA. For `GetFederationToken`, MFA is not necessarily associated with a specific user.
- Similarly, long-term credentials (IAM user access keys and root user access keys) cannot be used with MFA-protected API access because they don't expire.
- `AssumeRole` and `GetSessionToken` can also be called without MFA information. In that case, the caller gets back temporary security credentials, but the session information for those temporary credentials does not indicate that the user authenticated with MFA.
- To establish MFA protection for API operations, you add MFA conditions to policies. A policy must include the `aws:MultiFactorAuthPresent` condition key to enforce the use of MFA. For cross-account delegation, the role's trust policy must include the condition key.
- When you allow another AWS account to access resources in your account, the security of your resources depends on the configuration of the trusted account (the other account, not yours). This is true even when you require multi-factor authentication. Any identity in the trusted account that has permission to create virtual MFA devices can construct an MFA claim to satisfy that part of your role's trust policy. Before you allow members of another account access to your AWS resources that require multi-factor authentication, you should ensure that the trusted account's owner follows security best practices. For example, the trusted account should restrict access to sensitive API operations, such as MFA device-management API operations, to specific, trusted identities.
- If a policy includes an MFA condition, a request is denied if users have not been MFA authenticated, or if they provide an invalid MFA device identifier or invalid TOTP.

Scenario: MFA Protection for Cross-Account Delegation

In this scenario, you want to delegate access to IAM users in another account, but only if the users are authenticated with an AWS MFA device. (For more information about cross-account delegation, see [Roles Terms and Concepts \(p. 176\)](#).)

Imagine that you have account A (the trusting account that owns the resource to be accessed), with the IAM user Anaya, who has administrator permission. She wants to grant access to user Richard in account B (the trusted account), but wants to make sure that Richard is authenticated with MFA before he assumes the role.

1. In the trusting account A, Anaya creates an IAM role named `CrossAccountRole` and sets the principal in the role's trust policy to the account ID of account B. The trust policy grants permission to the AWS STS `AssumeRole` action. Anaya also adds an MFA condition to the trust policy, as in the following example.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {"AWS": "ACCOUNT-B-ID" },
      "Action": "sts:AssumeRole",
      "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
    }
  ]
}
```

2. Anaya adds a permissions policy to the role that specifies what the role is allowed to do. The permissions policy for a role with MFA protection is no different than any other role-permission policy. The following example shows the policy that Anaya adds to the role; it allows an assuming user to perform any Amazon DynamoDB action on the table `Books` in account A. This policy also allows the `dynamodb>ListTables` action, which is required to perform actions in the console.

Note

The permissions policy does not include an MFA condition. It is important to understand that the MFA authentication is used only to determine whether a user can assume the role. Once the user has assumed the role, no further MFA checks are made.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "TableActions",
            "Effect": "Allow",
            "Action": "dynamodb:*",
            "Resource": "arn:aws:dynamodb:*:ACCOUNT-A-ID:table/Books"
        },
        {
            "Sid": "ListTable",
            "Effect": "Allow",
            "Action": "dynamodb>ListTable",
            "Resource": "*"
        }
    ]
}
```

3. In trusted account B, the administrator makes sure that IAM user Richard is configured with an AWS MFA device and that he knows the ID of the device. The device ID is the serial number if it's a hardware MFA device, or the device's ARN if it's a virtual MFA device.
4. In account B, the administrator attaches the following policy to user Richard (or a group that he's a member of) that allows him to call the `AssumeRole` action. The resource is set to the ARN of the role that Anaya created in step 1. Notice that this policy does not contain an MFA condition.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["sts:AssumeRole"],
            "Resource": ["arn:aws:iam::ACCOUNT-A-ID:role/CrossAccountRole"]
        }
    ]
}
```

5. In account B, Richard (or an application that Richard is running) calls `AssumeRole`. The API call includes the ARN of the role to assume (`arn:aws:iam::ACCOUNT-A-ID:role/CrossAccountRole`), the ID of the MFA device, and the current TOTP that Richard gets from his device.

When Richard calls `AssumeRole`, AWS determines whether he has valid credentials, including the requirement for MFA. If so, Richard successfully assumes the role and can perform any DynamoDB action on the table named `Books` in account A while using the role's temporary credentials.

For an example of a program that calls `AssumeRole`, see [Calling AssumeRole with MFA Authentication \(Python\) \(p. 153\)](#).

Scenario: MFA Protection for Access to API Operations in the Current Account

In this scenario, you should ensure that a user in your AWS account can access sensitive API operations only when the user is authenticated using an AWS MFA device.

Imagine that you have account A that contains a group of developers who need to work with EC2 instances. Ordinary developers can work with the instances, but they are not granted permissions for the `ec2:StopInstances` or `ec2:TerminateInstances` actions. You want to limit those "destructive"

privileged actions to just a few trusted users, so you add MFA protection to the policy that allows these sensitive Amazon EC2 actions.

In this scenario, one of those trusted users is user Sofía. User Anaya is an administrator in account A.

1. Anaya makes sure that Sofía is configured with an AWS MFA device and that Sofía knows the ID of the device. The device ID is the serial number if it's a hardware MFA device, or the device's ARN if it's a virtual MFA device.
2. Anaya creates a group named `EC2-Admins` and adds user Sofía to the group.
3. Anaya attaches the following policy to the `EC2-Admins` group. This policy grants users permission to call the Amazon EC2 `StopInstances` and `TerminateInstances` actions only if the user has authenticated using MFA.

```
{  
    "Version": "2012-10-17",  
    "Statement": [{  
        "Effect": "Allow",  
        "Action": [  
            "ec2:StopInstances",  
            "ec2:TerminateInstances"  
        ],  
        "Resource": ["*"],  
        "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}  
    }]  
}
```

4. **Note**

For this policy to take effect, users must first sign out and then sign in again.

If user Sofía needs to stop or terminate an Amazon EC2 instance, she (or an application that she is running) calls `GetSessionToken`. This API operation passes the ID of the MFA device and the current TOTP that Sofía gets from her device.

5. User Sofía (or an application that Sofía is using) uses the temporary credentials provided by `GetSessionToken` to call the Amazon EC2 `StopInstances` or `TerminateInstances` action.

For an example of a program that calls `GetSessionToken`, see [Calling `GetSessionToken` with MFA Authentication \(Python and C#\) \(p. 151\)](#) later in this document.

Scenario: MFA Protection for Resources That Have Resource-based Policies

In this scenario, you are the owner of an S3 bucket, an SQS queue, or an SNS topic. You want to make sure that any user from any AWS account who accesses the resource is authenticated by an AWS MFA device.

This scenario illustrates a way to provide cross-account MFA protection without requiring users to assume a role first. In this case, the user can access the resource if three conditions are met: The user must be authenticated by MFA, be able to get temporary security credentials from `GetSessionToken`, and be in an account that is trusted by the resource's policy.

Imagine that you are in account A and you create an S3 bucket. You want to grant access to this bucket to users who are in several different AWS accounts, but only if those users are authenticated with MFA.

In this scenario, user Anaya is an administrator in account A. User Nikhil is an IAM user in account C.

1. In account A, Anaya creates a bucket named `Account-A-bucket`.
2. Anaya adds the bucket policy to the bucket. The policy allows any user in account A, account B, or account C to perform the Amazon S3 `PutObject` and `DeleteObject` actions in the bucket. The policy includes an MFA condition.

```
{  
    "Version": "2012-10-17",  
    "Statement": [{  
        "Effect": "Allow",  
        "Principal": {"AWS": [  
            "ACCOUNT-A-ID",  
            "ACCOUNT-B-ID",  
            "ACCOUNT-C-ID"  
        ]},  
        "Action": [  
            "s3:PutObject",  
            "s3:DeleteObject"  
        ],  
        "Resource": ["arn:aws:s3:::ACCOUNT-A-BUCKET-NAME/*"],  
        "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}  
    }]  
}
```

Note

Amazon S3 offers an MFA Delete feature for *root* account access (only). You can enable Amazon S3 MFA Delete when you set the versioning state of the bucket. Amazon S3 MFA Delete cannot be applied to an IAM user, and is managed independently from MFA-protected API access. An IAM user with permissions to delete a bucket cannot delete a bucket with Amazon S3 MFA Delete enabled. For more information on Amazon S3 MFA Delete, see [MFA Delete](#).

3. In account C, an administrator makes sure that user Nikhil is configured with an AWS MFA device and that he knows the ID of the device. The device ID is the serial number if it's a hardware MFA device, or the device's ARN if it's a virtual MFA device.
4. In account C, Nikhil (or an application that he is running) calls `GetSessionToken`. The call includes the ID or ARN of the MFA device and the current TOTP that Nikhil gets from his device.
5. Nikhil (or an application that he is using) uses the temporary credentials returned by `GetSessionToken` to call the Amazon S3 `PutObject` action to upload a file to Account-A-bucket.

For an example of a program that calls `GetSessionToken`, see [Calling GetSessionToken with MFA Authentication \(Python and C#\) \(p. 151\)](#) later in this document.

Note

The temporary credentials that `AssumeRole` returns won't work in this case. Although the user can provide MFA information to assume a role, the temporary credentials returned by `AssumeRole` don't include the MFA information. That information is required in order to meet the MFA condition in the policy.

Sample Code: Requesting Credentials with Multi-factor Authentication

The following examples show how to call `GetSessionToken` and `AssumeRole` operations and pass MFA authentication parameters. No permissions are required to call `GetSessionToken`, but you must have a policy that allows you to call `AssumeRole`. The credentials returned are then used to list all S3 buckets in the account.

Calling `GetSessionToken` with MFA Authentication (Python and C#)

The following examples, written using the [AWS SDK for Python \(Boto\)](#) and [AWS SDK for .NET](#), show how to call `GetSessionToken` and pass MFA authentication information. The temporary security credentials returned by the `GetSessionToken` operation are then used to list all S3 buckets in the account.

The policy attached to the user who runs this code (or to a group that the user is in) provides the permissions for the returned temporary credentials. For this example code, the policy must grant the user permission to request the Amazon S3 ListBuckets operation.

Using Python

```

import boto
from boto.s3.connection import S3Connection
from boto.sts import STSConnection

# Prompt for MFA time-based one-time password (TOTP)
mfa_TOTP = raw_input("Enter the MFA code: ")

# The calls to AWS STS GetSessionToken must be signed with the access key ID and secret
# access key of an IAM user. The credentials can be in environment variables or in
# a configuration file and will be discovered automatically
# by the STSConnection() function. For more information, see the Python SDK
# documentation: http://boto.readthedocs.org/en/latest/boto_config_tut.html

sts_connection = STSConnection()

# Use the appropriate device ID (serial number for hardware device or ARN for virtual
# device).
# Replace ACCOUNT-NUMBER-WITHOUT-HYPHENS and MFA-DEVICE-ID with appropriate values.

tempCredentials = sts_connection.get_session_token(
    duration=3600,
    mfa_serial_number="&region-arn;iam::ACCOUNT-NUMBER-WITHOUT-HYPHENS:mfa/MFA-DEVICE-ID",
    mfa_token=mfa_TOTP
)

# Use the temporary credentials to list the contents of an S3 bucket
s3_connection = S3Connection(
    aws_access_key_id=tempCredentials.access_key,
    aws_secret_access_key=tempCredentials.secret_key,
    security_token=tempCredentials.session_token
)

# Replace BUCKET-NAME with an appropriate value.
bucket = s3_connection.get_bucket(bucket_name="BUCKET-NAME")
objectlist = bucket.list()
for obj in objectlist:
    print obj.name

```

Using C#

```

Console.Write("Enter MFA code: ");
string mfatOTP = Console.ReadLine(); // Get string from user

/* The calls to AWS STS GetSessionToken must be signed using the access key ID and secret
access key of an IAM user. The credentials can be in environment variables or in
a configuration file and will be discovered automatically
by the AmazonSecurityTokenServiceClient constructor. For more information, see
https://docs.aws.amazon.com/sdk-for-net/v2/developer-guide/net-dg-config-creds.html
*/
AmazonSecurityTokenServiceClient stsClient =
    new AmazonSecurityTokenServiceClient();
GetSessionTokenRequest getSessionTokenRequest = new GetSessionTokenRequest();
getSessionTokenRequest.DurationSeconds = 3600;

// Replace ACCOUNT-NUMBER-WITHOUT-HYPHENS and MFA-DEVICE-ID with appropriate values
getSessionTokenRequest.SerialNumber = "arn:aws:iam::ACCOUNT-NUMBER-WITHOUT-HYPHENS:mfa/MFA-
DEVICE-ID";

```

```

getSessionTokenRequest.TokenCode = mfaTOTP;

GetSessionTokenResponse getSessionTokenResponse =
    stsClient.GetSessionToken(getSessionTokenRequest);

// Extract temporary credentials from result of GetSessionToken call
GetSessionTokenResult getSessionTokenResult =
    getSessionTokenResponse.GetSessionTokenResult;
string tempAccessKeyId = getSessionTokenResult.Credentials.AccessKeyId;
string tempSessionToken = getSessionTokenResult.Credentials.SessionToken;
string tempSecretAccessKey = getSessionTokenResult.Credentials.SecretAccessKey;
SessionAWSCredentials tempCredentials = new SessionAWSCredentials(tempAccessKeyId,
    tempSecretAccessKey, tempSessionToken);

// Use the temporary credentials to list the contents of an S3 bucket
// Replace BUCKET-NAME with an appropriate value
ListObjectsRequest S3ListObjectsRequest = new ListObjectsRequest();
S3ListObjectsRequest.BucketName = "BUCKET-NAME";
S3Client = AWSClientFactory.CreateAmazonS3Client(tempCredentials);
ListObjectsResponse S3ListObjectsResponse =
    S3Client.ListObjects(S3ListObjectsRequest);
foreach (S3Object s3Object in S3ListObjectsResponse.S3Objects)
{
    Console.WriteLine(s3Object.Key);
}

```

Calling AssumeRole with MFA Authentication (Python)

The following example, written using the [AWS SDK for Python \(Boto\)](#), shows how to call `AssumeRole` and pass MFA authentication information. The temporary security credentials returned by `AssumeRole` are then used to list all Amazon S3 buckets in the account.

For more information about this scenario, see [Scenario: MFA Protection for Cross-Account Delegation \(p. 148\)](#).

```

import boto
from boto.s3.connection import S3Connection
from boto.sts import STSConnection

# Prompt for MFA time-based one-time password (TOTP)
mfa_TOTP = raw_input("Enter the MFA code: ")

# The calls to AWS STS AssumeRole must be signed with the access key ID and secret
# access key of an IAM user. (The AssumeRole API operation can also be called using
# temporary
# credentials, but this example does not show that scenario.)
# The IAM user credentials can be in environment variables or in
# a configuration file and will be discovered automatically
# by the STSConnection() function. For more information, see the Python SDK
# documentation: http://boto.readthedocs.org/en/latest/boto_config_tut.html

sts_connection = STSConnection()

# Use appropriate device ID (serial number for hardware device or ARN for virtual device)
# Replace ACCOUNT-NUMBER-WITHOUT-HYPHENS, ROLE-NAME, and MFA-DEVICE-ID with appropriate
# values
tempCredentials = sts_connection.assume_role(
    role_arn="arn:aws:iam::ACCOUNT-NUMBER-WITHOUT-HYPHENS:role/ROLE-NAME",
    role_session_name="AssumeRoleSession1",
    mfa_serial_number="arn:aws:iam::ACCOUNT-NUMBER-WITHOUT-HYPHENS:mfa/MFA-DEVICE-ID",
    mfa_token=mfa_TOTP
)

# Use the temporary credentials to list the contents of an S3 bucket

```

```
s3_connection = S3Connection(  
    aws_access_key_id=tempCredentials.credentials.access_key,  
    aws_secret_access_key=tempCredentials.credentials.secret_key,  
    security_token=tempCredentials.credentials.session_token  
)  
  
# Replace BUCKET-NAME with a real bucket name  
bucket = s3_connection.get_bucket(bucket_name="BUCKET-NAME")  
objectlist = bucket.list()  
for obj in objectlist:  
    print obj.name
```

Finding Unused Credentials

To increase the security of your AWS account, remove IAM user credentials (that is, passwords and access keys) that are not needed. For example, when users leave your organization or no longer need AWS access, find the credentials that they were using and ensure that they are no longer operational. Ideally, you delete credentials if they are no longer needed. You can always recreate them at a later date if the need arises. At the very least, you should change the password or deactivate the access keys so that the former users no longer have access.

Of course, the definition of *unused* can vary and usually means a credential that has not been used within a specified period of time.

Finding Unused Passwords

You can use the AWS Management Console to view password usage information for your users. If you have a large number of users, you can use the console to download a credential report with information about when each user last used their console password. You can also access the information from the AWS CLI or the IAM API.

To find unused passwords (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Users**.
3. If necessary, add the **Console last sign-in** column to the users table:
 - a. Above the table on the far right, choose the settings icon ().
 - b. In **Manage Columns**, select **Console last sign-in**.
 - c. Choose **Close** to return to the list of users.
4. The **Console last sign-in** column shows the number of days since the user last signed in to AWS through the console. You can use this information to find users with passwords who have not signed in for more than a specified period of time. The column displays **Never** for users with passwords that have never signed in. **None** indicates users with no passwords. Passwords that have not been used recently might be good candidates for removal.

Important

Due to a service issue, password last used data does not include password use from May 3rd 2018 22:50 PDT to May 23rd 2018 14:08 PDT. This affects **last sign-in** dates shown in the IAM console and password last used dates in the [IAM credential report](#), and returned by the [GetUser API operation](#). If users signed in during the affected time, the password last used date that is returned is the date the user last signed in before May 3rd 2018. For users that signed in after May 23rd 2018 14:08 PDT, the returned password last used date is accurate. If you use password last used information to identify unused credentials for deletion, such as deleting users who did not sign in to AWS in the last 90 days, we recommend that you

adjust your evaluation window to include dates after May 23rd 2018. Alternatively, if your users use access keys to access AWS programmatically you can refer to access key last used information because it is accurate for all dates.

To find unused passwords by downloading the credentials report (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Credential report**.
3. Choose **Download Report** to download a comma-separated value (CSV) file named `status_reports_<date>T<time>.csv`. The fifth column contains the `password_last_used` column with the dates or one of the following:
 - **N/A** – Users that do not have a password assigned at all.
 - **no_information** – Users that have not used their password since IAM began tracking password age on October 20, 2014.

To find unused passwords (AWS CLI)

Run the following command to find unused passwords:

- `aws iam list-users` returns a list of users, each with a `PasswordLastUsed` value. If the value is missing, then the user either has no password or the password has not been used since IAM began tracking password age on October 20, 2014.

To find unused passwords (AWS API)

Call the following operation to find unused passwords:

- `ListUsers` returns a collection of users, each of which has a `<PasswordLastUsed>` value. If the value is missing, then the user either has no password or the password has not been used since IAM began tracking password age on October 20, 2014.

For information about the commands to download the credentials report, see [Getting Credential Reports \(AWS CLI\) \(p. 160\)](#).

Finding Unused Access Keys

You can use the AWS Management Console to view access key usage information for your users. If you have a large number of users, you can use the console to download a credentials report to find when each user last used their access keys. You can also access the information from the AWS CLI or the IAM API.

To find unused access keys (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Users**.
3. If necessary, add the **Access key last used** column to the users table:
 - a. Above the table on the far right, choose the settings icon ().
 - b. In **Manage Columns**, select **Access key last used**.
 - c. Choose **Close** to return to the list of users.

4. The **Access key last used** column shows the number of days since the user last accessed AWS programmatically. You can use this information to find users with access keys that have not been used for more than a specified period of time. The column displays **None** for users with no access keys. Access keys that have not been used recently might be good candidates for removal.

To find unused access keys by downloading the credentials report (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Credential Report**.
3. Choose **Download Report** to download a comma-separated value (CSV) file named `status_reports_<date>T<time>.csv`. Columns 11 through 13 contain the last used date, Region, and service information for access key 1. Columns 16 through 18 contain the same information for access key 2. The value is **N/A** if the user does not have an access key or the user has not used the access key since IAM began tracking access key age on April 22, 2015.

To find unused access keys (AWS CLI)

Run the following commands to find unused access keys:

- `aws iam list-access-keys` returns information about the access keys for a user, including the `AccessKeyId`.
- `aws iam get-access-key-last-used` takes an access key ID and returns output that includes the `LastUsedDate`, the Region in which the access key was last used, and the `ServiceName` of the last service requested. If `LastUsedDate` is missing, then the access key has not been used since IAM began tracking access key age on April 22, 2015.

To find unused access keys (AWS API)

Call the following operations to find unused access keys:

- `ListAccessKeys` returns a list of `AccessKeyId` values for access keys that are associated with the specified user.
- `GetAccessKeyLastUsed` takes an access key ID and returns a collection of values. Included are the `LastUsedDate`, the Region in which the access key was last used, and the `ServiceName` of the last service requested. If the value is missing, then either the user has no access key or the access key has not been used since IAM began tracking access key age on April 22, 2015.

For information about the commands to download the credentials report, see [Getting Credential Reports \(AWS CLI\) \(p. 160\)](#).

Getting Credential Reports for Your AWS Account

You can generate and download a *credential report* that lists all users in your account and the status of their various credentials, including passwords, access keys, and MFA devices. You can get a credential report from the AWS Management Console, the [AWS SDKs](#) and [Command Line Tools](#), or the IAM API.

You can use credential reports to assist in your auditing and compliance efforts. You can use the report to audit the effects of credential lifecycle requirements, such as password and access key rotation. You can provide the report to an external auditor, or grant permissions to an auditor so that he or she can download the report directly.

You can generate a credential report as often as once every four hours. When you request a report, IAM first checks whether a report for the AWS account has been generated within the past four hours. If so,

the most recent report is downloaded. If the most recent report for the account is older than four hours, or if there are no previous reports for the account, IAM generates and downloads a new report.

Topics

- [Required Permissions \(p. 157\)](#)
- [Understanding the Report Format \(p. 157\)](#)
- [Getting Credential Reports \(Console\) \(p. 160\)](#)
- [Getting Credential Reports \(AWS CLI\) \(p. 160\)](#)
- [Getting Credential Reports \(AWS API\) \(p. 160\)](#)

Required Permissions

The following permissions are needed to create and download reports:

- To create a credential report: `GenerateCredentialReport`
- To download the report: `GetCredentialReport`

Understanding the Report Format

Credential reports are formatted as comma-separated values (CSV) files. You can open CSV files with common spreadsheet software to perform analysis, or you can build an application that consumes the CSV files programmatically and performs custom analysis.

The CSV file contains the following columns:

user

The friendly name of the user.

arn

The Amazon Resource Name (ARN) of the user. For more information about ARNs, see [IAM ARNs \(p. 576\)](#).

user_creation_time

The date and time when the user was created, in [ISO 8601 date-time format](#).

password_enabled

When the user has a password, this value is `TRUE`. Otherwise it is `FALSE`. The value for the AWS account root user is always `not_supported`.

password_last_used

The date and time when the AWS account root user or IAM user's password was last used to sign in to an AWS website, in [ISO 8601 date-time format](#). AWS websites that capture a user's last sign-in time are the AWS Management Console, the AWS Discussion Forums, and the AWS Marketplace. When a password is used more than once in a 5-minute span, only the first use is recorded in this field.

- The value in this field is `no_information` in these cases:
 - The user's password has never been used.
 - There is no sign-in data associated with the password, such as when user's password has not been used after IAM started tracking this information on October 20, 2014.
- The value in this field is `N/A` (not applicable) when the user does not have a password.

Important

Due to a service issue, password last used data does not include password use from May 3rd 2018 22:50 PDT to May 23rd 2018 14:08 PDT. This affects [last sign-in](#) dates shown in the IAM console and password last used dates in the [IAM credential report](#), and returned by the [GetUser API operation](#). If users signed in during the affected time, the password last used date that is returned is the date the user last signed in before May 3rd 2018. For users that signed in after May 23rd 2018 14:08 PDT, the returned password last used date is accurate.

If you use password last used information to identify unused credentials for deletion, such as deleting users who did not sign in to AWS in the last 90 days, we recommend that you adjust your evaluation window to include dates after May 23rd 2018. Alternatively, if your users use access keys to access AWS programmatically you can refer to access key last used information because it is accurate for all dates.

password_last_changed

The date and time when the user's password was last set, in [ISO 8601 date-time format](#). If the user does not have a password, the value in this field is `N/A` (not applicable). The value for the AWS account (root) is always `not_supported`.

password_next_rotation

When the account has a [password policy](#) that requires password rotation, this field contains the date and time, in [ISO 8601 date-time format](#), when the user is required to set a new password. The value for the AWS account (root) is always `not_supported`.

mfa_active

When a [multi-factor authentication \(p. 119\)](#) (MFA) device has been enabled for the user, this value is `TRUE`. Otherwise it is `FALSE`.

access_key_1_active

When the user has an access key and the access key's status is `Active`, this value is `TRUE`. Otherwise it is `FALSE`.

access_key_1_last_rotated

The date and time, in [ISO 8601 date-time format](#), when the user's access key was created or last changed. If the user does not have an active access key, the value in this field is `N/A` (not applicable).

access_key_1_last_used_date

The date and time, in [ISO 8601 date-time format](#), when the user's access key was most recently used to sign an AWS API request. When an access key is used more than once in a 15-minute span, only the first use is recorded in this field.

The value in this field is `N/A` (not applicable) in these cases:

- The user does not have an access key.
- The access key has never been used.
- The access key has not been used after IAM started tracking this information on April 22, 2015.

access_key_1_last_used_region

The [AWS Region](#) in which the access key was most recently used. When an access key is used more than once in a 15-minute span, only the first use is recorded in this field.

The value in this field is `N/A` (not applicable) in these cases:

- The user does not have an access key.
- The access key has never been used.

- The access key was last used before IAM started tracking this information on April 22, 2015.
- The last used service is not Region-specific, such as Amazon S3.

access_key_1_last_used_service

The AWS service that was most recently accessed with the access key. The value in this field uses the service's [namespace](#)—for example, s3 for Amazon S3 and ec2 for Amazon EC2. When an access key is used more than once in a 15-minute span, only the first use is recorded in this field.

The value in this field is **N/A** (not applicable) in these cases:

- The user does not have an access key.
- The access key has never been used.
- The access key was last used before IAM started tracking this information on April 22, 2015.

access_key_2_active

When the user has a second access key and the second key's status is **Active**, this value is **TRUE**. Otherwise it is **FALSE**.

Note

Users can have up to two access keys, to make rotation easier. For more information about rotating access keys, see [Rotating Access Keys \(p. 115\)](#).

access_key_2_last_rotated

The date and time, in [ISO 8601 date-time format](#), when the user's second access key was created or last changed. If the user does not have a second active access key, the value in this field is **N/A** (not applicable).

access_key_2_last_used_date

The date and time, in [ISO 8601 date-time format](#), when the user's second access key was most recently used to sign an AWS API request. When an access key is used more than once in a 15-minute span, only the first use is recorded in this field.

The value in this field is **N/A** (not applicable) in these cases:

- The user does not have a second access key.
- The user's second access key has never been used.
- The user's second access key was last used before IAM started tracking this information on April 22, 2015.

access_key_2_last_used_region

The [AWS Region](#) in which the user's second access key was most recently used. When an access key is used more than once in a 15-minute span, only the first use is recorded in this field. The value in this field is **N/A** (not applicable) in these cases:

- The user does not have a second access key.
- The user's second access key has never been used.
- The user's second access key was last used before IAM started tracking this information on April 22, 2015.
- The last used service is not Region-specific, such as Amazon S3.

access_key_2_last_used_service

The AWS service that was most recently accessed with the user's second access key. The value in this field uses the service's [namespace](#)—for example, s3 for Amazon S3 and ec2 for Amazon EC2. When an access key is used more than once in a 15-minute span, only the first use is recorded in this field. The value in this field is **N/A** (not applicable) in these cases:

- The user does not have a second access key.
- The user's second access key has never been used.
- The user's second access key was last used before IAM started tracking this information on April 22, 2015.

cert_1_active

When the user has an X.509 signing certificate and that certificate's status is **Active**, this value is **TRUE**. Otherwise it is **FALSE**.

cert_1_last_rotated

The date and time, in [ISO 8601 date-time format](#), when the user's signing certificate was created or last changed. If the user does not have an active signing certificate, the value in this field is **N/A** (not applicable).

cert_2_active

When the user has a second X.509 signing certificate and that certificate's status is **Active**, this value is **TRUE**. Otherwise it is **FALSE**.

Note

Users can have up to two X.509 signing certificates, to make certificate rotation easier.

cert_2_last_rotated

The date and time, in [ISO 8601 date-time format](#), when the user's second signing certificate was created or last changed. If the user does not have a second active signing certificate, the value in this field is **N/A** (not applicable).

Getting Credential Reports (Console)

You can use the AWS Management Console to download a credential report as a comma-separated values (CSV) file.

To download a credential report (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Credential report**.
3. Choose **Download Report**.

Getting Credential Reports (AWS CLI)

Run the following commands:

- To generate a credential report: `aws iam generate-credential-report`
- To retrieve a credential report: `aws iam get-credential-report`

Getting Credential Reports (AWS API)

Call the following operations:

- To generate a credential report: [GenerateCredentialReport](#)
- To retrieve a credential report: [GetCredentialReport](#)

Using IAM with CodeCommit: Git Credentials, SSH Keys, and AWS Access Keys

CodeCommit is a managed version control service that hosts private Git repositories in the AWS cloud. To use CodeCommit, you configure your Git client to communicate with CodeCommit repositories. As part of this configuration, you provide IAM credentials that CodeCommit can use to authenticate you. IAM supports CodeCommit with three types of credentials:

- Git credentials, an IAM -generated user name and password pair you can use to communicate with CodeCommit repositories over HTTPS.
- SSH keys, a locally generated public-private key pair that you can associate with your IAM user to communicate with CodeCommit repositories over SSH.
- [AWS access keys \(p. 110\)](#), which you can use with the credential helper included with the AWS CLI to communicate with CodeCommit repositories over HTTPS.

See the following sections for more information about each option.

Use Git Credentials and HTTPS with CodeCommit (Recommended)

With Git credentials, you generate a static user name and password pair for your IAM user, and then use those credentials for HTTPS connections. You can also use these credentials with any third-party tool or integrated development environment (IDE) that supports static Git credentials.

Because these credentials are universal for all supported operating systems and compatible with most credential management systems, development environments, and other software development tools, this is the recommended method. You can reset the password for Git credentials at any time. You can also make the credentials inactive or delete them if you no longer need them.

Note

You cannot choose your own user name or password for Git credentials. IAM generates these credentials for you to help ensure they meet the security standards for AWS and secure repositories in CodeCommit. You can download the credentials only once, at the time they are generated. Make sure that you save the credentials in a secure location. If necessary, you can reset the password at any time, but doing so invalidates any connections configured with the old password. You must reconfigure connections to use the new password before you can connect.

See the following topics for more information:

- To create an IAM user, see [Creating an IAM User in Your AWS Account \(p. 86\)](#).
- To generate and use Git credentials with CodeCommit, see [For HTTPS Users Using Git Credentials](#) in the [AWS CodeCommit User Guide](#).

Note

Changing the name of an IAM user after generating Git credentials does not change the user name of the Git credentials. The user name and password remain the same and are still valid.

To rotate service specific credentials

1. Create a second service-specific credential set in addition to the set currently in use.
2. Update all of your applications to use the new set of credentials and validate that the applications are working.
3. Change the state of the original credentials to "Inactive".

4. Ensure that all of your applications are still working.
5. Delete the inactive service-specific credentials.

Use SSH Keys and SSH with CodeCommit

With SSH connections, you create public and private key files on your local machine that Git and CodeCommit use for SSH authentication. You associate the public key with your IAM user and store the private key on your local machine. See the following topics for more information:

- To create an IAM user, see [Creating an IAM User in Your AWS Account \(p. 86\)](#).
- To create an SSH public key and associate it with an IAM user, see [For SSH Connections on Linux, macOS, or Unix](#) or see [For SSH Connections on Windows](#) in the *AWS CodeCommit User Guide*.

Note

The public key must be encoded in ssh-rsa format or PEM format. The minimum bit-length of the public key is 2048 bits, and the maximum length is 16384 bits. This is separate from the size of the file you upload. For example, you can generate a 2048-bit key, and the resulting PEM file is 1679 bytes long. If you provide your public key in another format or size, you will see an error message stating that the key format is not valid.

Use HTTPS with the AWS CLI Credential Helper and CodeCommit

As an alternative to HTTPS connections with Git credentials, you can allow Git to use a cryptographically signed version of your IAM user credentials or Amazon EC2 instance role whenever Git needs to authenticate with AWS to interact with CodeCommit repositories. This is the only connection method for CodeCommit repositories that does not require an IAM user. This is also the only method that works with federated access and temporary credentials. Unless your business needs require federated access or the use of temporary credentials, creating and using IAM users for access is strongly recommended. See the following topics for more information:

- To learn more about federated access, see [Identity Providers and Federation \(p. 184\)](#) and [Providing Access to Externally Authenticated Users \(Identity Federation\) \(p. 182\)](#).
- To learn more about temporary credentials, see [Temporary Security Credentials \(p. 307\)](#) and [Temporary Access to CodeCommit Repositories](#).

The AWS CLI credential helper is not compatible with other credential helper systems, such as Keychain Access or Windows Credential Management. There are additional configuration considerations when you configure HTTPS connections with the credential helper. For more information, see [For HTTPS Connections on Linux, macOS, or Unix with the AWS CLI Credential Helper](#) or [HTTPS Connections on Windows with the AWS CLI Credential Helper](#) in the *AWS CodeCommit User Guide*.

Using IAM with Amazon Managed Apache Cassandra Service

Amazon Managed Apache Cassandra Service (MCS) is a scalable, highly available, and managed Apache Cassandra-compatible database service. You can access MCS using the AWS Management Console, by running a `cqlsh` client, or using an Apache 2.0 licensed Cassandra driver.

Note

If you plan to interact with MCS only through the console, you don't need to generate service-specific credentials. For more information, see [Accessing Amazon Managed Apache Cassandra Service Using the Console](#) in the *Amazon Managed Apache Cassandra Service Developer Guide*.

You must generate service-specific credentials to allow your users to access MCS using `cqlsh` or an Apache 2.0 licensed Cassandra driver. *Service-specific credentials* allow an IAM user to interact with one AWS service, but no others.

For more information about the permissions required to access MCS, see [Generating Service-Specific Credentials](#) in the *Amazon Managed Apache Cassandra Service Developer Guide*.

Generating MCS Credentials (Console)

You can use the AWS Management Console to generate Amazon Managed Apache Cassandra Service (MCS) credentials for your IAM users.

To generate MCS service-specific credentials (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Users** and then choose the name of the user that requires the credentials.
3. On the **Security Credentials** tab beneath **Credentials for Amazon Managed Apache Cassandra Service (MCS)**, choose **Generate credentials**.
4. Your service-specific credentials are now available. This is the only time that the password can be viewed or downloaded. You cannot recover it later. However, you can reset your password at any time. Save the user and password in a secure location, because you'll need them later.

Generating MCS Credentials (AWS CLI)

You can use the AWS CLI to generate Amazon Managed Apache Cassandra Service (MCS) credentials for your IAM users.

To generate MCS service-specific credentials (AWS CLI)

- Use the following command:
 - `aws iam create-service-specific-credential`

Generating MCS Credentials (AWS API)

You can use the AWS API to generate Amazon Managed Apache Cassandra Service (MCS) credentials for your IAM users.

To generate MCS service-specific credentials (AWS API)

- Complete the following operation:
 - [CreateServiceSpecificCredential](#)

Working with Server Certificates

To enable HTTPS connections to your website or application in AWS, you need an *SSL/TLS server certificate*. For certificates in a Region supported by AWS Certificate Manager (ACM), we recommend that you use ACM to provision, manage, and deploy your server certificates. In unsupported Regions, you must use IAM as a certificate manager. To learn which Regions ACM supports, see [AWS Certificate Manager Certificate Manager Regions and Endpoints](#) in the *AWS General Reference*.

ACM is the preferred tool to provision, manage, and deploy your server certificates. With ACM you can request a certificate or deploy an existing ACM or external certificate to AWS resources. Certificates provided by ACM are free and automatically renew. In a [supported Region](#), you can use ACM to manage server certificates from the console or programmatically. For more information about using ACM, see the [AWS Certificate Manager User Guide](#). For more information about requesting an ACM certificate, see [Request a Public Certificate or Request a Private Certificate](#) in the [AWS Certificate Manager User Guide](#). For more information about importing third party certificates into ACM, see [Importing Certificates](#) in the [AWS Certificate Manager User Guide](#).

Use IAM as a certificate manager only when you must support HTTPS connections in a Region that is not [supported by ACM](#). IAM securely encrypts your private keys and stores the encrypted version in IAM SSL certificate storage. IAM supports deploying server certificates in all Regions, but you must obtain your certificate from an external provider for use with AWS. You cannot upload an ACM certificate to IAM. Additionally, you cannot manage your certificates from the IAM Console.

For more information about uploading third party certificates to IAM, see the following topics.

Topics

- [Uploading a Server Certificate \(AWS API\) \(p. 164\)](#)
- [Retrieving a Server Certificate \(AWS API\) \(p. 165\)](#)
- [Listing Server Certificates \(AWS API\) \(p. 165\)](#)
- [Renaming a Server Certificate or Updating its Path \(AWS API\) \(p. 165\)](#)
- [Deleting a Server Certificate \(AWS API\) \(p. 166\)](#)
- [Troubleshooting \(p. 166\)](#)

Uploading a Server Certificate (AWS API)

To upload a server certificate to IAM, you must provide the certificate and its matching private key. When the certificate is not self-signed, you must also provide a certificate chain. (You don't need a certificate chain when uploading a self-signed certificate.) Before you upload a certificate, ensure that you have all these items and that they meet the following criteria:

- The certificate must be valid at the time of upload. You cannot upload a certificate before its validity period begins (the certificate's `NotBefore` date) or after it expires (the certificate's `NotAfter` date).
- The private key must be unencrypted. You cannot upload a private key that is protected by a password or passphrase. For help decrypting an encrypted private key, see [Troubleshooting \(p. 166\)](#).
- The certificate, private key, and certificate chain must all be PEM-encoded. For help converting these items to PEM format, see [Troubleshooting \(p. 166\)](#).

To use the [IAM API](#) to upload a certificate, send an `UploadServerCertificate` request. The following example shows how to do this with the [AWS Command Line Interface \(AWS CLI\)](#). The example assumes the following:

- The PEM-encoded certificate is stored in a file named `Certificate.pem`.
- The PEM-encoded certificate chain is stored in a file named `CertificateChain.pem`.
- The PEM-encoded, unencrypted private key is stored in a file named `PrivateKey.pem`.

To use the following example command, replace these file names with your own and replace `ExampleCertificate` with a name for your uploaded certificate. Type the command on one continuous line. The following example includes line breaks and extra spaces to make it easier to read.

```
$ aws iam upload-server-certificate --server-certificate-name ExampleCertificate
```

```
--certificate-body file://Certificate.pem  
--certificate-chain file://CertificateChain.pem  
--private-key file://PrivateKey.pem
```

When the preceding command is successful, it returns metadata about the uploaded certificate, including its [Amazon Resource Name \(ARN\)](#), its friendly name, its identifier (ID), its expiration date, and more.

Note

If you are uploading a server certificate to use with Amazon CloudFront, you must specify a path using the --path option. The path must begin with /cloudfront and must include a trailing slash (for example, /cloudfront/test/).

To use the AWS Tools for Windows PowerShell to upload a certificate, use [Publish-IAMServerCertificate](#).

Retrieving a Server Certificate (AWS API)

To use the IAM API to retrieve a certificate, send a [GetServerCertificate](#) request. The following example shows how to do this with the AWS CLI. Replace *ExampleCertificate* with the name of the certificate to retrieve.

```
$ aws iam get-server-certificate --server-certificate-name ExampleCertificate
```

When the preceding command is successful, it returns the certificate, the certificate chain (if one was uploaded), and metadata about the certificate.

Note

You cannot download or retrieve a private key from IAM after you upload it.

To use the AWS Tools for Windows PowerShell to retrieve a certificate, use [Get-IAMServerCertificate](#).

Listing Server Certificates (AWS API)

To use the IAM API to list your uploaded server certificates, send a [ListServerCertificates](#) request. The following example shows how to do this with the AWS CLI.

```
$ aws iam list-server-certificates
```

When the preceding command is successful, it returns a list that contains metadata about each certificate.

To use the AWS Tools for Windows PowerShell to list your uploaded server certificates, use [Get-IAMServerCertificates](#).

Renaming a Server Certificate or Updating its Path (AWS API)

To use the IAM API to rename a server certificate or update its path, send an [UpdateServerCertificate](#) request. The following example shows how to do this with the AWS CLI.

To use the following example command, replace the old and new certificate names and the certificate path, and type the command on one continuous line. The following example includes line breaks and extra spaces to make it easier to read.

```
$ aws iam update-server-certificate --server-certificate-name ExampleCertificate  
--new-server-certificate-name CloudFrontCertificate
```

```
--new-path /cloudfront/
```

When the preceding command is successful, it does not return any output.

To use the AWS Tools for Windows PowerShell to rename a server certificate or update its path, use [Update-IAMServerCertificate](#).

Deleting a Server Certificate (AWS API)

To use the IAM API to delete a server certificate, send a [DeleteServerCertificate](#) request. The following example shows how to do this with the AWS CLI.

To use the following example command, replace *ExampleCertificate* with the name of the certificate to delete.

```
$ aws iam delete-server-certificate --server-certificate-name ExampleCertificate
```

When the preceding command is successful, it does not return any output.

To use the AWS Tools for Windows PowerShell to delete a server certificate, use [Remove-IAMServerCertificate](#).

Troubleshooting

Before you can upload a certificate to IAM, you must make sure that the certificate, private key, and certificate chain are all PEM-encoded. You must also ensure that the private key is unencrypted. See the following examples.

Example PEM-encoded certificate

```
-----BEGIN CERTIFICATE-----  
Base64-encoded certificate  
-----END CERTIFICATE-----
```

Example PEM-encoded, unencrypted private key

```
-----BEGIN RSA PRIVATE KEY-----  
Base64-encoded private key  
-----END RSA PRIVATE KEY-----
```

Example PEM-encoded certificate chain

A certificate chain contains one or more certificates. You can use a text editor, the copy command in Windows, or the Linux cat command to concatenate your certificate files into a chain. When you include multiple certificates, each certificate must certify the preceding certificate. You accomplish this by concatenating the certificates, including the root CA certificate last.

The following example contains three certificates, but your certificate chain might contain more or fewer certificates.

```
-----BEGIN CERTIFICATE-----  
Base64-encoded certificate  
-----END CERTIFICATE-----  
-----BEGIN CERTIFICATE-----
```

```
-----  
-----BEGIN CERTIFICATE-----  
-----  
-----  
-----  
-----END CERTIFICATE-----  
-----  
-----  
-----END CERTIFICATE-----
```

If these items are not in the right format for uploading to IAM, you can use [OpenSSL](#) to convert them to the right format.

To convert a certificate or certificate chain from DER to PEM

Use the [OpenSSL x509 command](#), as in the following example. In the following example command, replace `Certificate.der` with the name of the file that contains your DER-encoded certificate. Replace `Certificate.pem` with the preferred name of the output file to contain the PEM-encoded certificate.

```
$ openssl x509 -inform DER -in Certificate.der -outform PEM -out Certificate.pem
```

To convert a private key from DER to PEM

Use the [OpenSSL rsa command](#), as in the following example. In the following example command, replace `PrivateKey.der` with the name of the file that contains your DER-encoded private key. Replace `PrivateKey.pem` with the preferred name of the output file to contain the PEM-encoded private key.

```
$ openssl rsa -inform DER -in PrivateKey.der -outform PEM -out PrivateKey.pem
```

To decrypt an encrypted private key (remove the password or passphrase)

Use the [OpenSSL rsa command](#), as in the following example. To use the following example command, replace `EncryptedPrivateKey.pem` with the name of the file that contains your encrypted private key. Replace `PrivateKey.pem` with the preferred name of the output file to contain the PEM-encoded unencrypted private key.

```
$ openssl rsa -in EncryptedPrivateKey.pem -out PrivateKey.pem
```

To convert a certificate bundle from PKCS#12 (PFX) to PEM

Use the [OpenSSL pkcs12 command](#), as in the following example. In the following example command, replace `CertificateBundle.p12` with the name of the file that contains your PKCS#12-encoded certificate bundle. Replace `CertificateBundle.pem` with the preferred name of the output file to contain the PEM-encoded certificate bundle.

```
$ openssl pkcs12 -in CertificateBundle.p12 -out CertificateBundle.pem -nodes
```

To convert a certificate bundle from PKCS#7 to PEM

Use the [OpenSSL pkcs7 command](#), as in the following example. In the following example command, replace `CertificateBundle.p7b` with the name of the file that contains your PKCS#7-encoded

certificate bundle. Replace `CertificateBundle.pem` with the preferred name of the output file to contain the PEM-encoded certificate bundle.

```
$ openssl pkcs7 -in CertificateBundle.p7b -print_certs -out CertificateBundle.pem
```

IAM Groups

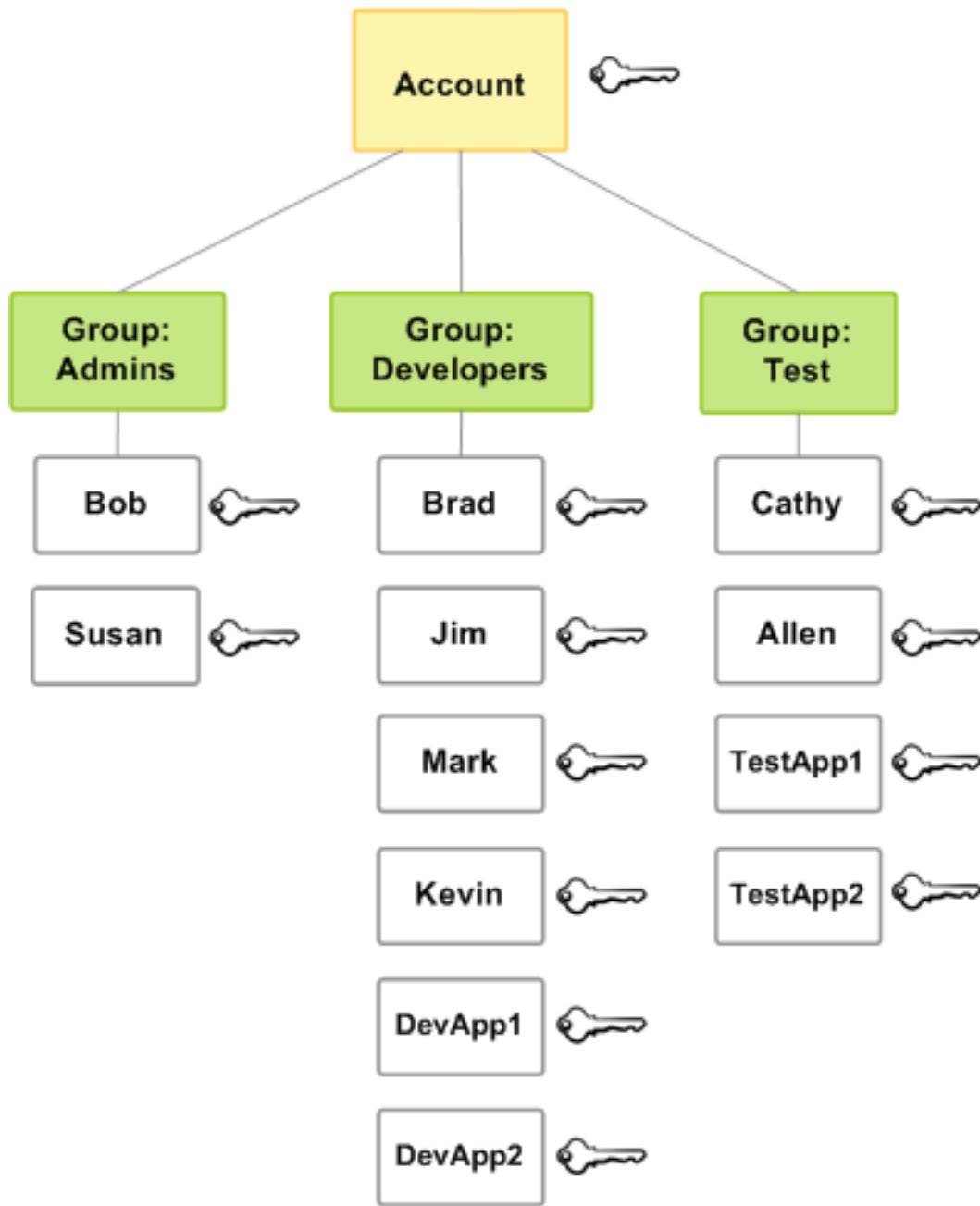
An IAM [group](#) (p. 168) is a collection of IAM users. Groups let you specify permissions for multiple users, which can make it easier to manage the permissions for those users. For example, you could have a group called `Admins` and give that group the types of permissions that administrators typically need. Any user in that group automatically has the permissions that are assigned to the group. If a new user joins your organization and needs administrator privileges, you can assign the appropriate permissions by adding the user to that group. Similarly, if a person changes jobs in your organization, instead of editing that user's permissions, you can remove him or her from the old groups and add him or her to the appropriate new groups.

Note that a group is not truly an "identity" in IAM because it cannot be identified as a `Principal` in a permission policy. It is simply a way to attach policies to multiple users at one time.

Following are some important characteristics of groups:

- A group can contain many users, and a user can belong to multiple groups.
- Groups can't be nested; they can contain only users, not other groups.
- There's no default group that automatically includes all users in the AWS account. If you want to have a group like that, you need to create it and assign each new user to it.
- There's a limit to the number of groups you can have, and a limit to how many groups a user can be in. For more information, see [IAM and STS Limits](#) (p. 581).

The following diagram shows a simple example of a small company. The company owner creates an `Admins` group for users to create and manage other users as the company grows. The `Admins` group creates a `Developers` group and a `Test` group. Each of these groups consists of users (humans and applications) that interact with AWS (Jim, Brad, DevApp1, and so on). Each user has an individual set of security credentials. In this example, each user belongs to a single group. However, users can belong to multiple groups.



Creating IAM Groups

To set up a group, you need to create the group. Then give the group permissions based on the type of work that you expect the users in the group to do. Finally, add users to the group.

For information about the permissions that you need in order to create a group, see [Permissions Required to Access IAM Resources \(p. 517\)](#).

To create an IAM group and attach policies (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, click **Groups** and then click **Create New Group**.
3. In the **Group Name** box, type the name of the group and then click **Next Step**.

Note

Group names can be a combination of up to 64 letters, digits, and these characters: plus (+), equal (=), comma (,), period (.), at sign (@), underscore (_), and hyphen (-). Names must be unique within an account. They are not distinguished by case. For example, you cannot create groups named both **ADMINS** and **admins**. For more information about limitations on IAM entities, see [IAM and STS Limits \(p. 581\)](#).

4. In the list of policies, select the check box for each policy that you want to apply to all members of the group. Then click **Next Step**.
5. Click **Create Group**.

For an example of how to set up an Administrators group, see [Creating Your First IAM Admin User and Group \(p. 20\)](#).

To create IAM groups (AWS CLI or AWS API)

Use one of the following:

- AWS CLI: `aws iam create-group`
- AWS API: `CreateGroup`

Managing IAM Groups

Amazon Web Services offers multiple tools for managing IAM groups. For information about the permissions that you need in order to add and remove users in a group, see [Permissions Required to Access IAM Resources \(p. 517\)](#).

Topics

- [Listing IAM Groups \(p. 170\)](#)
- [Adding and Removing Users in an IAM Group \(p. 171\)](#)
- [Attaching a Policy to an IAM Group \(p. 172\)](#)
- [Renaming an IAM Group \(p. 173\)](#)
- [Deleting an IAM Group \(p. 173\)](#)

Listing IAM Groups

You can list all the groups in your account, list the users in a group, and list the groups a user belongs to. If you use the AWS CLI or AWS API, you can list all the groups with a particular path prefix.

To list all the groups in your account

Do any of the following:

- [AWS Management Console](#): In the navigation pane, choose **Groups**.
- AWS CLI: `aws iam list-groups`

- AWS API: [ListGroup](#)

To list the users in a specific group

Do any of the following:

- [AWS Management Console](#): In the navigation pane, choose **Groups**, choose the name of the group, and then choose the **Users** tab.
- AWS CLI: `aws iam get-group`
- AWS API: `GetGroup`

To list all the groups that a user is in

Do any of the following:

- [AWS Management Console](#): In the navigation pane, choose **Users**, choose the user name, and then choose the **Groups** tab.
- AWS CLI: `aws iam list-groups-for-user`
- AWS API: `ListGroupsForUser`

Adding and Removing Users in an IAM Group

Use groups to apply the same permissions policies across multiple users at once. You can then add users to or remove users from an IAM group. This is useful as people enter and leave your organization.

[View Policy Access](#)

Before you change the permissions for a policy, you should review its recent service-level activity. This is important because you don't want to remove access from a principal (person or application) who is using it. For more information about viewing service last accessed data, see [Refining Permissions Using Service Last Accessed Data \(p. 475\)](#).

[Add or Remove a User in a Group \(Console\)](#)

You can use the AWS Management Console to add or remove a user from a group.

To add a user to an IAM group (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Groups** and then choose the name of the group.
3. Choose the **Users** tab and then choose **Add Users to Group**. Select the check box next to the users you want to add.
4. Choose **Add Users**.

To remove a user from an IAM group (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Groups** and then choose the name of the group.
3. Choose the **Users** tab and then choose **Remove Users from Group**. Select the check box next to the users you want to remove.

4. Choose **Remove Users**.

Add or Remove a User in a Group (AWS CLI)

You can use the AWS CLI to add or remove a user from a group.

To add a user to an IAM group (AWS CLI)

- Use the following command:
 - [aws iam add-user-to-group](#)

To remove a user from an IAM group (AWS CLI)

- Use the following command:
 - [aws iam remove-user-from-group](#)

Add or Remove a User in a Group (AWS API)

You can use the AWS API to add or remove a user in a group.

To add a user to an IAM group (AWS API)

- Complete the following operation:
 - [AddUserToGroup](#)

To remove a user from an IAM group (AWS API)

- Complete the following operation:
 - [RemoveUserFromGroup](#)

Attaching a Policy to an IAM Group

You can attach an [AWS managed policy \(p. 365\)](#)—that is, a prewritten policy provided by AWS—to a group, as explained in the following steps. To attach a customer managed policy—that is, a policy with custom permissions that you create—you must first create the policy. For information about creating customer managed policies, see [Creating IAM Policies \(p. 443\)](#).

For more information about permissions and policies, see [Access Management \(p. 355\)](#).

To attach a policy to a group (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, select **Policies**.
3. In the list of policies, select the check box next to the name of the policy to attach. You can use the **Filter** menu and the search box to filter the list of policies.
4. Click **Policy actions**, then click **Attach**.
5. For **Filter**, choose **All Types**, then click **Groups**.
6. Select the check box next to the name of the group to attach the policy to, then click **Attach policy**.

To attach a policy to a group (AWS CLI or AWS API)

Do either of the following:

- AWS CLI: [aws iam attach-group-policy](#)
- AWS API: [AttachGroupPolicy](#)

Renaming an IAM Group

When you change a group's name or path, the following happens:

- Any policies attached to the group stay with the group under the new name.
- The group retains all its users under the new name.
- The unique ID for the group remains the same. For more information about unique IDs, see [Unique Identifiers \(p. 579\)](#).

Because IAM does not automatically update policies that refer to the group as a resource to use the new name; you must be careful when you rename a group. Before you rename your group, you must manually check all of your policies to find any policies where that group is mentioned by name. For example, let's say Bob is the manager of the testing part of the organization. Bob has a policy attached to his IAM user entity that lets him add and remove users from the Test group. If an administrator changes the name of the group (or changes the group path), the administrator must also update the policy attached to Bob to use the new name or path. Otherwise Bob won't be able to add and remove users from the group.

To find policies that refer to a group as a resource:

1. From the navigation pane of the IAM console, choose **Policies**.
2. From the **Policy type** drop-down list, choose **Customer managed** to filter the policies to show only your custom policies.
3. Choose the arrow next to each policy name to expand the policy summary.
4. Choose **IAM** from the list of services, if it exists.
5. Look for the name of your group in the **Resource** column.
6. Choose **Edit policy** to change the name of your group in the policy.

To change the name of an IAM group

Do any of the following:

- **AWS Management Console:** In the navigation pane, choose **Groups** and then select the check box next to the group name. From the **Group Actions** list at the top of the page, choose **Edit Group Name**. Type the new group name and then choose **Yes, Edit**.
- AWS CLI: [aws iam update-group](#)
- AWS API: [UpdateGroup](#)

Deleting an IAM Group

When you delete a group in the AWS Management Console, the console automatically removes all group members, detaches all attached managed policies, and deletes all inline policies. However, because IAM does not automatically delete policies that refer to the group as a resource, you must be careful when you delete a group. Before you delete your group, you must manually check all of your policies to find

any policies where that group is mentioned by name. For example, let's say John is the manager of the testing part of the organization. John has a policy attached to his IAM user entity that lets him add and remove users from the Test group. If an administrator deletes the group, the administrator must also delete the policy attached to John.

To find policies that refer to a group as a resource

1. From the navigation pane of the IAM console, choose **Policies**.
2. From the **Policy type** drop-down list, choose **Customer managed** to filter the policies to show only your custom policies.
3. Choose the arrow next to each policy name to expand the policy summary.
4. Choose **IAM** from the list of services, if it exists.
5. Look for the name of your group in the **Resource** column.
6. Choose **Delete policy** to delete the policy.

In contrast, when you use the AWS CLI, Tools for Windows PowerShell, or AWS API to delete a group, you must first remove the users in the group. Then delete any inline policies embedded in the group. Next, detach any managed policies that are attached to the group. Only then can you delete the group itself.

Deleting an IAM Group (Console)

You can delete an IAM group from the AWS Management Console.

To delete an IAM group (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Groups**.
3. In the list of groups, select the check box next to the name of the group to delete. You can use the **Filter** menu and the search box to filter the list of policies.
4. Click **Group Actions**, then click **Delete Group**.
5. In the confirmation box, click **Yes, Delete**.

Deleting an IAM Group (AWS CLI)

You can delete an IAM group from the AWS CLI.

To delete an IAM group (AWS CLI)

1. Remove all users from the group.
 - [aws iam get-group](#) (to get the list of users in the group), and [aws iam remove-user-from-group](#) (to remove a user from the group)
2. Delete all inline policies embedded in the group.
 - [aws iam list-group-policies](#) (to get a list of the group's inline policies), and [aws iam delete-group-policy](#) (to delete the group's inline policies)
3. Detach all managed policies attached to the group.
 - [aws iam list-attached-group-policies](#) (to get a list of the managed policies attached to the group), and [aws iam detach-group-policy](#) (to detach a managed policy from the group)
4. Delete the group.

- [aws iam delete-group](#)

Deleting an IAM Group (AWS API)

You can use the AWS API to delete an IAM group.

To delete an IAM group (AWS API)

1. Remove all users from the group.
 - [GetGroup](#) (to get the list of users in the group) and [RemoveUserFromGroup](#) (to remove a user from the group)
2. Delete all inline policies embedded in the group.
 - [ListGroupPolicies](#) (to get a list of the group's inline policies) and [DeleteGroupPolicy](#) (to delete the group's inline policies)
3. Detach all managed policies attached to the group.
 - [ListAttachedGroupPolicies](#) (to get a list of the managed policies attached to the group) and [DetachGroupPolicy](#) (to detach a managed policy from the group)
4. Delete the group.
 - [DeleteGroup](#)

IAM Roles

An IAM *role* is an IAM identity that you can create in your account that has specific permissions. An IAM role is similar to an IAM user, in that it is an AWS identity with permission policies that determine what the identity can and cannot do in AWS. However, instead of being uniquely associated with one person, a role is intended to be assumable by anyone who needs it. Also, a role does not have standard long-term credentials such as a password or access keys associated with it. Instead, when you assume a role, it provides you with temporary security credentials for your role session.

You can use roles to delegate access to users, applications, or services that don't normally have access to your AWS resources. For example, you might want to grant users in your AWS account access to resources they don't usually have, or grant users in one AWS account access to resources in another account. Or you might want to allow a mobile app to use AWS resources, but not want to embed AWS keys within the app (where they can be difficult to rotate and where users can potentially extract them). Sometimes you want to give AWS access to users who already have identities defined outside of AWS, such as in your corporate directory. Or, you might want to grant access to your account to third parties so that they can perform an audit on your resources.

For these scenarios, you can delegate access to AWS resources using an *IAM role*. This section introduces roles and the different ways you can use them, when and how to choose among approaches, and how to create, manage, switch to (or assume), and delete roles.

Topics

- [Roles Terms and Concepts \(p. 176\)](#)
- [Common Scenarios for Roles: Users, Applications, and Services \(p. 178\)](#)
- [Identity Providers and Federation \(p. 184\)](#)
- [Using Service-Linked Roles \(p. 221\)](#)
- [Creating IAM Roles \(p. 228\)](#)

- [Using IAM Roles \(p. 253\)](#)
- [Managing IAM Roles \(p. 278\)](#)
- [How IAM Roles Differ from Resource-based Policies \(p. 292\)](#)

Roles Terms and Concepts

Here are some basic terms to help you get started with roles.

Role

An IAM identity that you can create in your account that has specific permissions. An IAM role has some similarities to an IAM user. Roles and users are both AWS identities with permissions policies that determine what the identity can and cannot do in AWS. However, instead of being uniquely associated with one person, a role is intended to be assumable by anyone who needs it. Also, a role does not have standard long-term credentials such as a password or access keys associated with it. Instead, when you assume a role, it provides you with temporary security credentials for your role session.

Roles can be used by the following:

- An IAM user in the same AWS account as the role
- An IAM user in a different AWS account than the role
- A web service offered by AWS such as Amazon Elastic Compute Cloud (Amazon EC2)
- An external user authenticated by an external identity provider (IdP) service that is compatible with SAML 2.0 or OpenID Connect, or a custom-built identity broker.

AWS service role

A role that a service assumes to perform actions in your account on your behalf. When you set up some AWS service environments, you must define a role for the service to assume. This service role must include all the permissions required for the service to access the AWS resources that it needs. Service roles vary from service to service, but many allow you to choose your permissions, as long as you meet the documented requirements for that service. Service roles provide access only within your account and cannot be used to grant access to services in other accounts. You can create, modify, and delete a service role from within IAM.

AWS service role for an EC2 instance

A special type of service role that an application running on an Amazon EC2 instance can assume to perform actions in your account. This role is assigned to the EC2 instance when it is launched. Applications running on that instance can retrieve temporary security credentials and perform actions that the role allows. For details about using a service role for an EC2 instance, see [Using an IAM Role to Grant Permissions to Applications Running on Amazon EC2 Instances \(p. 269\)](#).

AWS service-linked role

A unique type of service role that is linked directly to an AWS service. Service-linked roles are predefined by the service and include all the permissions that the service requires to call other AWS services on your behalf. The linked service also defines how you create, modify, and delete a service-linked role. A service might automatically create or delete the role. It might allow you to create, modify, or delete the role as part of a wizard or process in the service. Or it might require that you use IAM to create or delete the role. Regardless of the method, service-linked roles make setting up a service easier because you don't have to manually add the necessary permissions.

Note

If you are already using a service when it begins supporting service-linked roles, you might receive an email announcing a new role in your account. In this case, the service automatically created the service-linked role in your account. You don't need to take any

action to support this role, and you should not manually delete it. For more information, see [A New Role Appeared in My AWS Account \(p. 564\)](#).

For information about which services support using service-linked roles, see [AWS Services That Work with IAM \(p. 586\)](#) and look for the services that have **Yes** in the **Service-Linked Role** column. Choose a **Yes** with a link to view the service-linked role documentation for that service. If the service does not include documentation for creating, modifying, or deleting the service-linked role, then you can use the IAM console, AWS CLI, or API. For more information, see [Using Service-Linked Roles \(p. 221\)](#).

Role chaining

Role chaining occurs when you use a role to assume a second role through the AWS CLI or API. For example, assume that `User1` has permission to assume `RoleA` and `RoleB`. Additionally, `RoleA` has permission to assume `RoleB`. You can assume `RoleA` by using `User1`'s long-term user credentials in the `AssumeRole` API operation. This operation returns `RoleA`'s short-term credentials. To engage in role chaining, you can use `RoleA`'s short-term credentials to assume `RoleB`.

When you assume a role, you can pass a session tag and set the tag as transitive. Transitive session tags are passed to all subsequent sessions in a role chain. To learn more about session tags, see [Passing Session Tags in AWS STS \(p. 299\)](#).

Role chaining limits your AWS CLI or AWS API role session to a maximum of one hour. When you use the `AssumeRole` API operation to assume a role, you can specify the duration of your role session with the `DurationSeconds` parameter. You can specify a parameter value of up to 43200 seconds (12 hours), depending on the [maximum session duration setting \(p. 255\)](#) for your role. However, if you assume a role using role chaining and provide a `DurationSeconds` parameter value greater than one hour, the operation fails.

Delegation

The granting of permissions to someone to allow access to resources that you control. Delegation involves setting up a trust between two accounts. The first is the account that owns the resource (the trusting account). The second is the account that contains the users that need to access the resource (the trusted account). The trusted and trusting accounts can be any of the following:

- The same account.
- Separate accounts that are both under your organization's control.
- Two accounts owned by different organizations.

To delegate permission to access a resource, you [create an IAM role \(p. 229\)](#) in the trusting account that has two [policies \(p. 178\)](#) attached. The *permissions policy* grants the user of the role the needed permissions to carry out the intended tasks on the resource. The *trust policy* specifies which trusted account members are allowed to assume the role.

When you create a trust policy, you cannot specify a wildcard (*) as a principal. The trust policy is attached to the role in the trusting account, and is one-half of the permissions. The other half is a permissions policy attached to the user in the trusted account that [allows that user to switch to, or assume the role \(p. 255\)](#). A user who assumes a role temporarily gives up his or her own permissions and instead takes on the permissions of the role. When the user exits, or stops using the role, the original user permissions are restored. An additional parameter called [external ID \(p. 233\)](#) helps ensure secure use of roles between accounts that are not controlled by the same organization.

Federation

The creation of a trust relationship between an external identity provider and AWS. Users can sign in to a web identity provider, such as [Login with Amazon](#), [Facebook](#), [Google](#), or any IdP that is compatible with [OpenID Connect \(OIDC\)](#). Users can also sign in to an enterprise identity system that is compatible with Security Assertion Markup Language (SAML) 2.0, such as Microsoft Active Directory Federation Services. When you use OIDC and SAML 2.0 to configure a trust relationship

between these external identity providers and AWS, the user is assigned to an IAM role. The user also receives temporary credentials that allow the user to access your AWS resources.

Federated user

Instead of creating an IAM user, you can use existing identities from AWS Directory Service, your enterprise user directory, or a web identity provider. These are known as *federated users*. AWS assigns a role to a federated user when access is requested through an [identity provider \(p. 184\)](#). For more information about federated users, see [Federated Users and Roles \(p. 12\)](#) in the *IAM User Guide*.

Trust policy

A [JSON policy document \(p. 651\)](#) in which you define the principals that you *trust* to assume the role. A role trust policy is a required [resource-based policy \(p. 357\)](#) that is attached to a role in IAM. The [principals \(p. 601\)](#) that you can specify in the trust policy include users, roles, accounts, and services.

Permissions policy

A permissions document in [JSON](#) format in which you define what actions and resources the role can use. The document is written according to the rules of the [IAM policy language \(p. 597\)](#).

Permissions boundary

An advanced feature in which you use policies to limit the maximum permissions that an identity-based policy can grant to a role. You cannot apply a permissions boundary to a service-linked role. For more information, see [Permissions Boundaries for IAM Entities \(p. 370\)](#).

Principal

An entity in AWS that can perform actions and access resources. A principal can be an AWS account root user, an IAM user, or a role. You can grant permissions to access a resource in one of two ways:

- You can attach a permissions policy to a user (directly, or indirectly through a group) or to a role.
- For those services that support [resource-based policies \(p. 12\)](#), you can identify the principal in the `Principal` element of a policy attached to the resource.

If you reference an AWS account as principal, it generally means any principal defined within that account.

Note

You cannot use a wildcard (*) in the `Principal` element in a role's trust policy.

Role for cross-account access

A role that grants access to resources in one account to a trusted principal in a different account. Roles are the primary way to grant cross-account access. However, some AWS services allow you to attach a policy directly to a resource (instead of using a role as a proxy). These are called resource-based policies, and you can use them to grant principals in another AWS account access to the resource. Some of these resources include Amazon Simple Storage Service (S3) buckets, S3 Glacier vaults, Amazon Simple Notification Service (SNS) topics, and Amazon Simple Queue Service (SQS) queues. To learn which services support resource-based policies, see [AWS Services That Work with IAM \(p. 586\)](#). For more information about resource-based policies, see [How IAM Roles Differ from Resource-based Policies \(p. 292\)](#).

Common Scenarios for Roles: Users, Applications, and Services

As with most AWS features, you generally have two ways to use a role: interactively in the IAM console, or programmatically with the AWS CLI, Tools for Windows PowerShell, or API.

- IAM users in your account using the IAM console can *switch to* a role to temporarily use the permissions of the role in the console. The users give up their original permissions and take on the permissions assigned to the role. When the users exit the role, their original permissions are restored.
- An application or a service offered by AWS (like Amazon EC2) can *assume* a role by requesting temporary security credentials for a role with which to make programmatic requests to AWS. You use a role this way so that you don't have to share or maintain long-term security credentials (for example, by creating an IAM user) for each entity that requires access to a resource.

Note

This guide uses the phrases *switch to a role* and *assume a role* interchangeably.

The simplest way to use roles is to grant your IAM users permissions to switch to roles that you create within your own or another AWS account. They can switch roles easily using the IAM console to use permissions that you don't ordinarily want them to have, and then exit the role to surrender those permissions. This can help prevent *accidental* access to or modification of sensitive resources.

For more complex uses of roles, such as granting access to applications and services, or federated external users, you can call the `AssumeRole` API. This API call returns a set of temporary credentials that the application can use in subsequent API calls. Actions attempted with the temporary credentials have only the permissions granted by the associated role. An application doesn't have to "exit" the role the way a user in the console does; rather the application simply stops using the temporary credentials and resumes making calls with the original credentials.

Federated users sign in by using credentials from an identity provider (IdP). AWS then provides temporary credentials to the trusted IdP to pass on to the user for including in subsequent AWS resource requests. Those credentials provide the permissions granted to the assigned role.

This section provides overviews of the following scenarios:

- [Provide access for an IAM user in one AWS account that you own to access resources in another account that you own \(p. 179\)](#)
- [Provide access to IAM users in AWS accounts owned by third parties \(p. 181\)](#)
- [Provide access for services offered by AWS to AWS resources \(p. 182\)](#)
- [Provide access for externally authenticated users \(identity federation\) \(p. 182\)](#)

Providing Access to an IAM User in Another AWS Account That You Own

You can grant your IAM users permission to switch to roles within your AWS account or to roles defined in other AWS accounts that you own.

Note

If you want to grant access to an account that you do not own or control, see [Providing Access to AWS Accounts Owned by Third Parties \(p. 181\)](#) later in this topic.

Imagine that you have Amazon EC2 instances that are critical to your organization. Instead of directly granting your users permission to terminate the instances, you can create a role with those privileges. Then allow administrators to switch to the role when they need to terminate an instance. Doing this adds the following layers of protection to the instances:

- You must explicitly grant your users permission to assume the role.
- Your users must actively switch to the role using the AWS Management Console or assume the role using the AWS CLI or AWS API..
- You can add multi-factor authentication (MFA) protection to the role so that only users who sign in with an MFA device can assume the role. To learn how to configure a role so that users who assume

the role must first be authenticated using multi-factor authentication (MFA), see [Configuring MFA-Protected API Access \(p. 145\)](#).

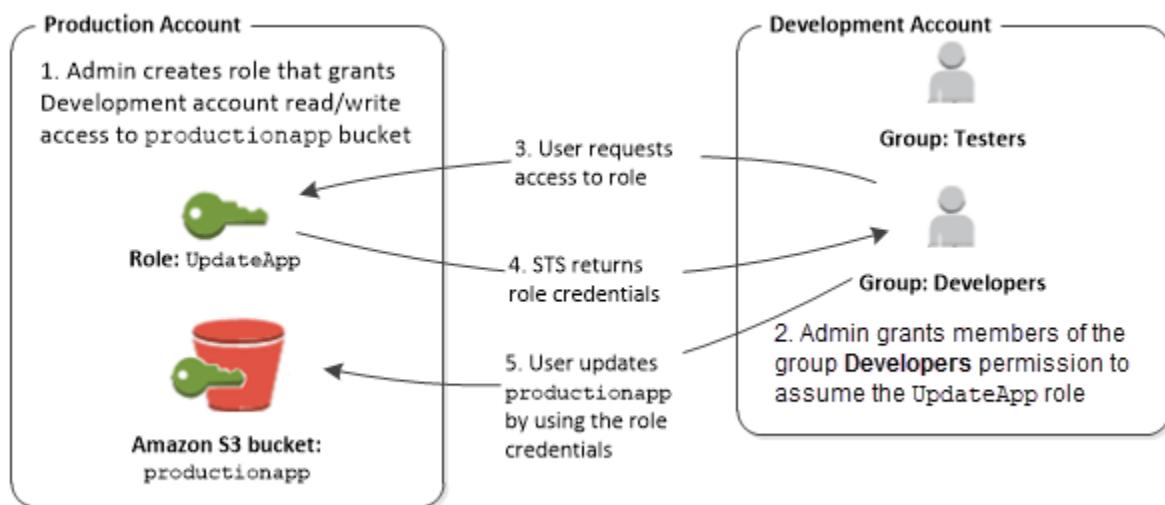
We recommend using this approach to enforce the *principle of least privilege*. That means restricting the use of elevated permissions to only those times when they are needed for specific tasks. With roles you can help prevent accidental changes to sensitive environments, especially if you combine them with [auditing \(p. 341\)](#) to help ensure that roles are only used when needed.

When you create a role for this purpose, you specify the accounts by ID whose users need access in the `Principal` element of the role's trust policy. You can then grant specific users in those other accounts permissions to switch to the role. To learn whether principals in accounts outside of your zone of trust (trusted organization, OU, or account) have access to assume your roles, see [What is IAM Access Analyzer?](#).

A user in one account can switch to a role in the same or a different account. While using the role, the user can perform only the actions and access only the resources permitted by the role; their original user permissions are suspended. When the user exits the role, the original user permissions are restored.

Example Scenario Using Separate Development and Production Accounts

Imagine that your organization has multiple AWS accounts to isolate a development environment from a production environment. Users in the development account might occasionally need to access resources in the production account. For example, you might need cross-account access when you are promoting an update from the development environment to the production environment. Although you could create separate identities (and passwords) for users who work in both accounts, managing credentials for multiple accounts makes identity management difficult. In the following figure, all users are managed in the development account, but some developers require limited access to the production account. The development account has two groups: Testers and Developers, and each group has its own policy.



1. In the production account, an administrator uses IAM to create the `UpdateApp` role in that account. In the role, the administrator defines a trust policy that specifies the development account as a `Principal`, meaning that authorized users from the development account can use the `UpdateApp` role. The administrator also defines a permissions policy for the role that specifies which role users have read and write permissions to the Amazon S3 bucket named `productionapp`.

The administrator then shares the appropriate information with anyone who needs to assume the role. That information is the account number and name of the role (for AWS console users) or

the Amazon Resource Name (ARN) (for AWS CLI or AWS API access). The role ARN might look like `arn:aws:iam::123456789012:role/UpdateApp`, where the role is named `UpdateApp` and the role was created in account number 123456789012.

Note

The administrator can optionally configure the role so that users who assume the role must first be authenticated using multi-factor authentication (MFA). For more information, see [Configuring MFA-Protected API Access \(p. 145\)](#).

2. In the development account, an administrator grants members of the Developers group permission to switch to the role. This is done by granting the Developers group permission to call the AWS Security Token Service (AWS STS) `AssumeRole` API for the `UpdateApp` role. Any IAM user that belongs to the Developers group in the development account can now switch to the `UpdateApp` role in the production account. Other users who are not in the developer group do not have permission to switch to the role and therefore cannot access the S3 bucket in the production account.
3. The user requests switches to the role:
 - AWS console: The user chooses the account name on the navigation bar and chooses **Switch Role**. The user specifies the account ID (or alias) and role name. Alternatively, the user can click on a link sent in email by the administrator. The link takes the user to the **Switch Role** page with the details already filled in.
 - AWS API/AWS CLI: A user in the Developers group of the development account calls the `AssumeRole` function to obtain credentials for the `UpdateApp` role. The user specifies the ARN of the `UpdateApp` role as part of the call. If a user in the Testers group makes the same request, the request fails because Testers do not have permission to call `AssumeRole` for the `UpdateApp` role ARN.
4. AWS STS returns temporary credentials:
 - AWS console: AWS STS verifies the request with the role's trust policy to ensure that the request is from a trusted entity (which it is: the development account). After verification, AWS STS returns **temporary security credentials** to the AWS console.
 - API/CLI: AWS STS verifies the request against the role's trust policy to ensure that the request is from a trusted entity (which it is: the Development account). After verification, AWS STS returns **temporary security credentials** to the application.
5. The temporary credentials allow access to the AWS resource:
 - AWS console: The AWS console uses the temporary credentials on behalf of the user for all subsequent console actions, in this case, to read and write to the `productionapp` bucket. The console cannot access any other resource in the production account. When the user exits the role, the user's permissions revert to the original permissions held before switching to the role.
 - API/CLI: The application uses the temporary security credentials to update the `productionapp` bucket. With the temporary security credentials, the application can only read from and write to the `productionapp` bucket and cannot access any other resource in the Production account. The application does not have to exit the role, but instead stops using the temporary credentials and uses the original credentials in subsequent API calls.

Providing Access to AWS Accounts Owned by Third Parties

When third parties require access to your organization's AWS resources, you can use roles to delegate access to them. For example, a third party might provide a service for managing your AWS resources. With IAM roles, you can grant these third parties access to your AWS resources without sharing your AWS security credentials. Instead, the third party can access your AWS resources by assuming a role that you create in your AWS account. To learn whether principals in accounts outside of your zone of trust (trusted organization, OU, or account) have access to assume your roles, see [What is IAM Access Analyzer?](#).

Third parties must provide you with the following information for you to create a role that they can assume:

- The third party's AWS account ID. You specify their AWS account ID as the principal when you define the trust policy for the role.
- An external ID to uniquely associate with the role. The external ID can be any secret identifier that is known by you and the third party. For example, you can use an invoice ID between you and the third party, but do not use something that can be guessed, like the name or phone number of the third party. You must specify this ID when you define the trust policy for the role. The third party must provide this ID when they assume the role. For more information about the external ID, see [How to Use an External ID When Granting Access to Your AWS Resources to a Third Party \(p. 233\)](#).
- The permissions that the third party requires to work with your AWS resources. You must specify these permissions when defining the role's permission policy. This policy defines what actions they can take and what resources they can access.

After you create the role, you must provide the role's Amazon Resource Name (ARN) to the third party. They require your role's ARN in order to assume the role.

For details about creating a role to delegate access to a third party, see [How to Use an External ID When Granting Access to Your AWS Resources to a Third Party \(p. 233\)](#).

Important

When you grant third parties access to your AWS resources, they can access any resource that you specify in the policy. Their use of your resources is billed to you. Ensure that you limit their use of your resources appropriately.

Providing Access to an AWS Service

Many AWS services require that you use roles to control what that service can access. A role that a service assumes to perform actions on your behalf is called a [service role \(p. 176\)](#). When a role serves a specialized purpose for a service, it can be categorized as a [service role for EC2 instances \(p. 176\)](#), or a [service-linked role \(p. 176\)](#). See the [AWS documentation](#) for each service to see if it uses roles and to learn how to assign a role for the service to use.

For details about creating a role to delegate access to a service offered by AWS, see [Creating a Role to Delegate Permissions to an AWS Service \(p. 236\)](#).

Providing Access to Externally Authenticated Users (Identity Federation)

Your users might already have identities outside of AWS, such as in your corporate directory. If those users need to work with AWS resources (or work with applications that access those resources), then those users also need AWS security credentials. You can use an IAM role to specify permissions for users whose identity is federated from your organization or a third-party identity provider (IdP).

Federating Users of a Mobile or Web-based App with Amazon Cognito

If you create a mobile or web-based app that accesses AWS resources, the app needs security credentials in order to make programmatic requests to AWS. For most mobile application scenarios, we recommend that you use [Amazon Cognito](#). You can use this service with the [AWS Mobile SDK for iOS](#) and the [AWS Mobile SDK for Android and Fire OS](#) to create unique identities for users and authenticate them for secure access to your AWS resources. Amazon Cognito supports the same identity providers as those listed in the next section, and it also supports [developer authenticated identities](#) and unauthenticated (guest) access. Amazon Cognito also provides API operations for synchronizing user data so that it is preserved as users move between devices. For more information, see [Using Amazon Cognito for Mobile Apps \(p. 185\)](#).

Federating Users with Public Identity Service Providers or OpenID Connect

Whenever possible, use Amazon Cognito for mobile and web-based application scenarios. Amazon Cognito does most of the behind-the-scenes work with public identity provider services for you. It works with the same third-party services and also supports anonymous sign-ins. However, for more advanced scenarios, you can work directly with a third-party service like Login with Amazon, Facebook, Google, or any IdP that is compatible with OpenID Connect (OIDC). For more information about using web identity federation using one of these services, see [About Web Identity Federation \(p. 185\)](#).

Federating users with SAML 2.0

If your organization already uses an identity provider software package that supports SAML 2.0 (Security Assertion Markup Language 2.0), you can create trust between your organization as an identity provider (IdP) and AWS as the service provider. You can then use SAML to provide your users with federated single-sign on (SSO) to the AWS Management Console or federated access to call AWS API operations. For example, if your company uses Microsoft Active Directory and Active Directory Federation Services, then you can federate using SAML 2.0. For more information about federating users with SAML 2.0, see [About SAML 2.0-based Federation \(p. 190\)](#).

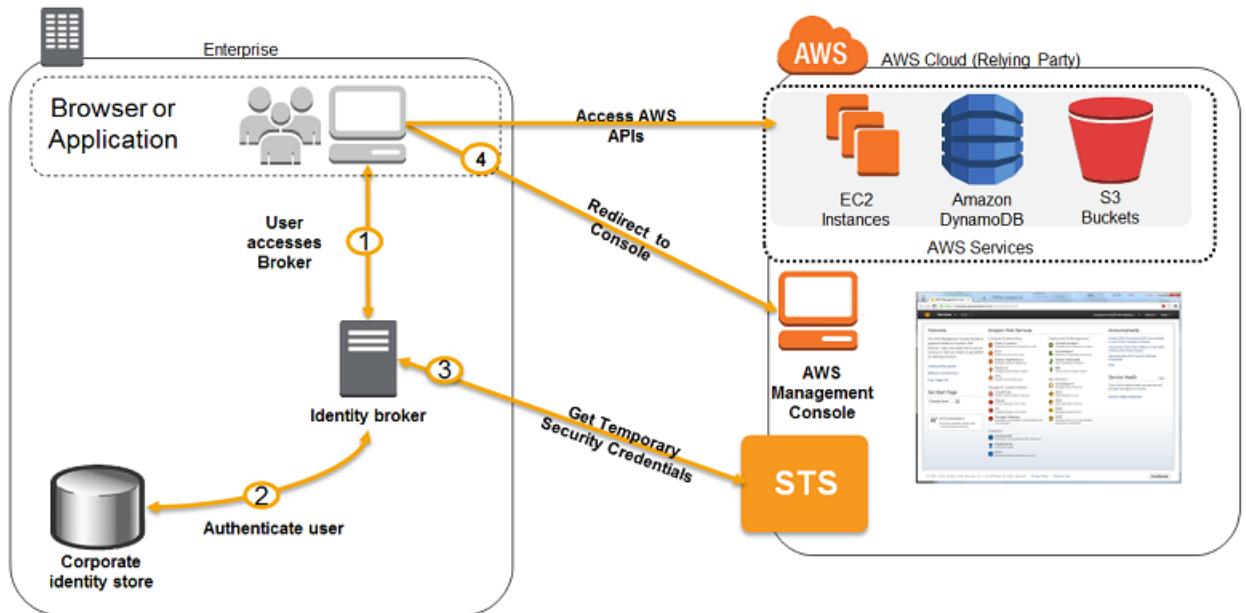
Federating users by creating a custom identity broker application

If your identity store is not compatible with SAML 2.0, then you can build a custom identity broker application to perform a similar function. The broker application authenticates users, requests temporary credentials for users from AWS, and then provides them to the user to access AWS resources.

For example, Example Corp. has many employees who need to run internal applications that access the company's AWS resources. The employees already have identities in the company identity and authentication system, and Example Corp. doesn't want to create a separate IAM user for each company employee.

Bob is a developer at Example Corp. To enable Example Corp. internal applications to access the company's AWS resources, Bob develops a custom identity broker application. The application verifies that employees are signed into the existing Example Corp. identity and authentication system, which might use LDAP, Active Directory, or another system. The identity broker application then obtains temporary security credentials for the employees. This scenario is similar to the previous one (a mobile app that uses a custom authentication system), except that the applications that need access to AWS resources all run within the corporate network, and the company has an existing authentication system.

To get temporary security credentials, the identity broker application calls either `AssumeRole` or `GetFederationToken` to obtain temporary security credentials, depending on how Bob wants to manage the policies for users and when the temporary credentials should expire. (For more information about the differences between these API operations, see [Temporary Security Credentials \(p. 307\)](#) and [Controlling Permissions for Temporary Security Credentials \(p. 322\)](#).) The call returns temporary security credentials consisting of an AWS access key ID, a secret access key, and a session token. The identity broker application makes these temporary security credentials available to the internal company application. The app can then use the temporary credentials to make calls to AWS directly. The app caches the credentials until they expire, and then requests a new set of temporary credentials. The following figure illustrates this scenario.



This scenario has the following attributes:

- The identity broker application has permissions to access IAM's token service (STS) API to create temporary security credentials.
- The identity broker application is able to verify that employees are authenticated within the existing authentication system.
- Users are able to get a temporary URL that gives them access to the AWS Management Console (which is referred to as single sign-on).

To see a sample application similar to the identity broker application that is described in this scenario, go to [Identity Federation Sample Application for an Active Directory Use Case](#) at [AWS Sample Code & Libraries](#). For information about creating temporary security credentials, see [Requesting Temporary Security Credentials \(p. 309\)](#). For more information about federated users getting access to the AWS Management Console, see [Enabling SAML 2.0 Federated Users to Access the AWS Management Console \(p. 211\)](#).

Identity Providers and Federation

If you already manage user identities outside of AWS, you can use IAM *identity providers* instead of creating IAM users in your AWS account. With an identity provider (IdP), you can manage your user identities outside of AWS and give these external user identities permissions to use AWS resources in your account. This is useful if your organization already has its own identity system, such as a corporate user directory. It is also useful if you are creating a mobile app or web application that requires access to AWS resources.

When you use an IAM identity provider, you don't have to create custom sign-in code or manage your own user identities. The IdP provides that for you. Your external users sign in through a well-known IdP, such as Login with Amazon, Facebook, or Google. You can give those external identities permissions to use AWS resources in your account. IAM identity providers help keep your AWS account secure because you don't have to distribute or embed long-term security credentials, such as access keys, in your application.

To use an IdP, you create an IAM identity provider entity to establish a trust relationship between your AWS account and the IdP. IAM supports IdPs that are compatible with [OpenID Connect \(OIDC\)](#) or [SAML 2.0 \(Security Assertion Markup Language 2.0\)](#). For more information about using one of these IdPs with AWS, see the following sections:

- [About Web Identity Federation \(p. 185\)](#)
- [About SAML 2.0-based Federation \(p. 190\)](#)

For details about creating the IAM identity provider entity to establish a trust relationship between a compatible IdP and AWS, see [Creating IAM Identity Providers \(p. 194\)](#)

About Web Identity Federation

Imagine that you are creating a mobile app that accesses AWS resources, such as a game that runs on a mobile device and stores player and score information using Amazon S3 and DynamoDB.

When you write such an app, you'll make requests to AWS services that must be signed with an AWS access key. However, we **strongly recommend** that you do **not** embed or distribute long-term AWS credentials with apps that a user downloads to a device, even in an encrypted store. Instead, build your app so that it requests temporary AWS security credentials dynamically when needed using *web identity federation*. The supplied temporary credentials map to an AWS role that has only the permissions needed to perform the tasks required by the mobile app.

With web identity federation, you don't need to create custom sign-in code or manage your own user identities. Instead, users of your app can sign in using a well-known external identity provider (IdP), such as Login with Amazon, Facebook, Google, or any other [OpenID Connect \(OIDC\)](#)-compatible IdP. They can receive an authentication token, and then exchange that token for temporary security credentials in AWS that map to an IAM role with permissions to use the resources in your AWS account. Using an IdP helps you keep your AWS account secure, because you don't have to embed and distribute long-term security credentials with your application.

For most scenarios, we recommend that you use [Amazon Cognito](#) because it acts as an identity broker and does much of the federation work for you. For details, see the following section, [Using Amazon Cognito for Mobile Apps \(p. 185\)](#).

If you don't use Amazon Cognito, then you must write code that interacts with a web IdP, such as Facebook, and then calls the `AssumeRoleWithWebIdentity` API to trade the authentication token you get from those IdPs for AWS temporary security credentials. If you have already used this approach for existing apps, you can continue to use it.

Topics

- [Using Amazon Cognito for Mobile Apps \(p. 185\)](#)
- [Using Web Identity Federation API Operations for Mobile Apps \(p. 187\)](#)
- [Identifying Users with Web Identity Federation \(p. 188\)](#)
- [Additional Resources for Web Identity Federation \(p. 190\)](#)

Using Amazon Cognito for Mobile Apps

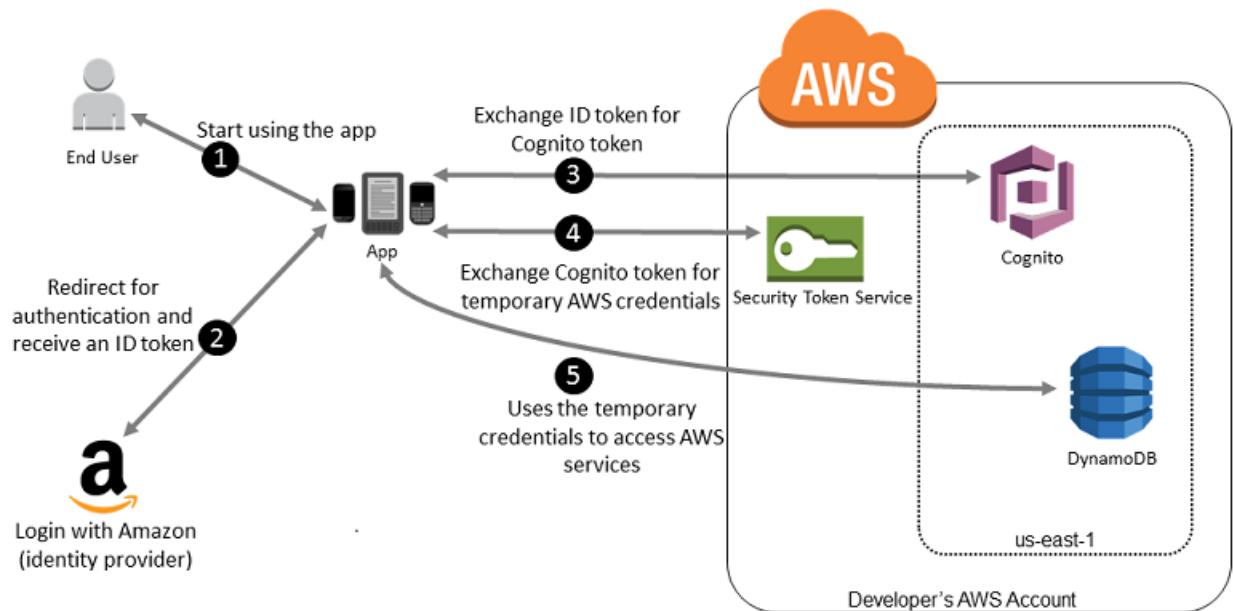
The preferred way to use web identity federation is to use [Amazon Cognito](#). For example, Adele the developer is building a game for a mobile device where user data such as scores and profiles is stored in Amazon S3 and Amazon DynamoDB. Adele could also store this data locally on the device and use Amazon Cognito to keep it synchronized across devices. She knows that for security and maintenance reasons, long-term AWS security credentials should not be distributed with the game. She also knows that the game might have a large number of users. For all of these reasons, she does not want to create new user identities in IAM for each player. Instead, she builds the game so that users can sign in using an

identity that they've already established with a well-known external identity provider (IdP), such as **Login with Amazon**, **Facebook**, **Google**, or any **OpenID Connect** (OIDC)-compatible IdP. Her game can take advantage of the authentication mechanism from one of these providers to validate the user's identity.

To enable the mobile app to access her AWS resources, Adele first registers for a developer ID with her chosen IdPs. She also configures the application with each of these providers. In her AWS account that contains the Amazon S3 bucket and DynamoDB table for the game, Adele uses Amazon Cognito to create IAM roles that precisely define permissions that the game needs. If she is using an OIDC IdP, she also creates an IAM OIDC identity provider entity to establish trust between her AWS account and the IdP.

In the app's code, Adele calls the sign-in interface for the IdP that she configured previously. The IdP handles all the details of letting the user sign in, and the app gets an OAuth access token or OIDC ID token from the provider. Adele's app can trade this authentication information for a set of temporary security credentials that consist of an AWS access key ID, a secret access key, and a session token. The app can then use these credentials to access web services offered by AWS. The app is limited to the permissions that are defined in the role that it assumes.

The following figure shows a simplified flow for how this might work, using Login with Amazon as the IdP. For Step 2, the app can also use Facebook, Google, or any OIDC-compatible IdP, but that's not shown here.



1. A customer starts your app on a mobile device. The app asks the user to sign in.
2. The app uses Login with Amazon resources to accept the user's credentials.
3. The app uses Cognito API operations to exchange the Login with Amazon ID token for a Cognito token.
4. The app requests temporary security credentials from AWS STS, passing the Cognito token.
5. The temporary security credentials can be used by the app to access any AWS resources required by the app to operate. The role associated with the temporary security credentials and its assigned policies determines what can be accessed.

Use the following process to configure your app to use Amazon Cognito to authenticate users and give your app access to AWS resources. For specific steps to accomplish this scenario, consult the documentation for Amazon Cognito.

1. (Optional) Sign up as a developer with Login with Amazon, Facebook, Google, or any other OpenID Connect (OIDC)-compatible IdP and configure one or more apps with the provider. This step is optional because Amazon Cognito also supports unauthenticated (guest) access for your users.
2. Go to [Amazon Cognito in the AWS Management Console](#). Use the Amazon Cognito wizard to create an identity pool, which is a container that Amazon Cognito uses to keep end user identities organized for your apps. You can share identity pools between apps. When you set up an identity pool, Amazon Cognito creates one or two IAM roles (one for authenticated identities, and one for unauthenticated "guest" identities) that define permissions for Amazon Cognito users.
3. Download and integrate the [AWS SDK for iOS](#) or the [AWS SDK for Android](#) with your app, and import the files required to use Amazon Cognito.
4. Create an instance of the Amazon Cognito credentials provider, passing the identity pool ID, your AWS account number, and the Amazon Resource Name (ARN) of the roles that you associated with the identity pool. The Amazon Cognito wizard in the AWS Management Console provides sample code to help you get started.
5. When your app accesses an AWS resource, pass the credentials provider instance to the client object, which passes temporary security credentials to the client. The permissions for the credentials are based on the role or roles that you defined earlier.

For more information, see the following:

- [Amazon Cognito Identity in the AWS Mobile SDK for Android Developer Guide](#).
- [Amazon Cognito Identity in the AWS Mobile SDK for iOS Developer Guide](#).

Using Web Identity Federation API Operations for Mobile Apps

For best results, use Amazon Cognito as your identity broker for almost all web identity federation scenarios. Amazon Cognito is easy to use and provides additional capabilities like anonymous (unauthenticated) access, and synchronizing user data across devices and providers. However, if you have already created an app that uses web identity federation by manually calling the `AssumeRoleWithWebIdentity` API, you can continue to use it and your apps will still work fine.

Note

To help understand how web identity federation works, you can use the [Web Identity Federation Playground](#). This interactive website lets you walk through the process of authenticating via Login with Amazon, Facebook, or Google, getting temporary security credentials, and then using those credentials to make a request to AWS.

The process for using web identity federation **without** Amazon Cognito follows this general outline:

1. Sign up as a developer with the external identity provider (IdP) and configure your app with the IdP, who gives you a unique ID for your app. (Different IdPs use different terminology for this process. This outline uses the term *configure* for the process of identifying your app with the IdP.) Each IdP gives you an app ID that's unique to that IdP, so if you configure the same app with multiple IdPs, your app will have multiple app IDs. You can configure multiple apps with each provider.

The following external links provide information about using some of the commonly used identity providers (IdPs):

- [Login with Amazon Developer Center](#)
- [Add Facebook Login to Your App or Website](#) on the Facebook developers site.
- [Using OAuth 2.0 for Login \(OpenID Connect\)](#) on the Google developers site.

Note

Although Amazon Cognito and Google are based on OIDC technology, you don't have to create an IAM identity provider entity to use them. Support for Amazon Cognito and Google are built-in to AWS.

2. If you use an IdP that is compatible with OIDC, then create an IAM identity provider entity for it.
3. In IAM, [create one or more roles \(p. 242\)](#). For each role, define who can assume the role (the trust policy) and what permissions the app's users are to have (the permissions policy). Typically, you create one role for each IdP that an app supports. For example, you might create a role that is assumed by an app when the user signs in through Login with Amazon, a second role for the same app where the user signs in through Facebook, and a third role for the app where the user signs in through Google. For the trust relationship, specify the IdP (like Amazon.com) as the Principal (the trusted entity), and include a Condition that matches the IdP assigned app ID. Examples of roles for different providers are described later in this topic.
4. In your application, authenticate your users with the IdP. The specifics of how to do this vary both according to which IdP you're using (Login with Amazon, Facebook, or Google) and on which platform your app runs. For example, an Android app's method of authentication can differ from that of an iOS app or a JavaScript-based web app.

Typically, if the user is not already signed in, the IdP takes care of displaying a sign-in page. After the IdP authenticates the user, the IdP returns an authentication token with information about the user to your app. The information included depends on what the IdP exposes and what information the user is willing to share. You can use this information in your app.

5. In your app, make an *unsigned* call to the `AssumeRoleWithWebIdentity` action to request temporary security credentials. In the request, you pass the IdP's authentication token and specify the Amazon Resource Name (ARN) for the IAM role that you created for that IdP. AWS verifies that the token is trusted and valid and if so, returns temporary security credentials to your app that have the permissions for the role that you name in the request. The response also includes metadata about the user from the IdP, such as the unique user ID that the IdP associates with the user.
6. Using the temporary security credentials from the `AssumeRoleWithWebIdentity` response, your app makes signed requests to AWS API operations. The user ID information from the IdP can distinguish users in your app—for example, you can put objects into Amazon S3 folders that include the user ID as prefixes or suffixes. This lets you create access control policies that lock the folder so only the user with that ID can access it. For more information, see [Identifying Users with Web Identity Federation \(p. 188\)](#) later in this topic.
7. Your app should cache the temporary security credentials so that you do not have to get new ones each time the app needs to make a request to AWS. By default, the credentials are good for one hour. When the credentials expire (or before then), you make another call to `AssumeRoleWithWebIdentity` to obtain a new set of temporary security credentials. Depending on the IdP and how they manage their tokens, you might have to refresh the IdP's token before you make a new call to `AssumeRoleWithWebIdentity`, since the IdP's tokens also usually expire after a fixed time. If you use the AWS SDK for iOS or the AWS SDK for Android, you can use the `AmazonSTSCredentialsProvider` action, which manages the IAM temporary credentials, including refreshing them as required.

Identifying Users with Web Identity Federation

When you create access policies in IAM, it's often useful to be able to specify permissions based on configured apps and on the ID of users who have authenticated using an external identity provider (IdP). For example, your mobile app that's using web identity federation might keep information in Amazon S3 using a structure like this:

```
myBucket/app1/user1
myBucket/app1/user2
myBucket/app1/user3
...
myBucket/app2/user1
myBucket/app2/user2
myBucket/app2/user3
...
```

You might also want to additionally distinguish these paths by provider. In that case, the structure might look like the following (only two providers are listed to save space):

```
myBucket/Amazon/app1/user1
myBucket/Amazon/app1/user2
myBucket/Amazon/app1/user3
...
myBucket/Amazon/app2/user1
myBucket/Amazon/app2/user2
myBucket/Amazon/app2/user3

myBucket/Facebook/app1/user1
myBucket/Facebook/app1/user2
myBucket/Facebook/app1/user3
...
myBucket/Facebook/app2/user1
myBucket/Facebook/app2/user2
myBucket/Facebook/app2/user3
...
```

For these structures, app1 and app2 represent different apps, such as different games, and each user of the app has a distinct folder. The values for app1 and app2 might be friendly names that you assign (for example, mynumbersgame) or they might be the app IDs that the providers assign when you configure your app. If you decide to include provider names in the path, those can also be friendly names like Cognito, Amazon, Facebook, and Google.

You can typically create the folders for app1 and app2 through the AWS Management Console, since the application names are static values. That's true also if you include the provider name in the path, since the provider name is also a static value. In contrast, the user-specific folders (**user1**, **user2**, **user3**, etc.) have to be created at run time from the app, using the user ID that's available in the `SubjectFromWebIdentityToken` value that is returned by the request to `AssumeRoleWithWebIdentity`.

To write policies that allow exclusive access to resources for individual users, you can match the complete folder name, including the app name and provider name, if you're using that. You can then include the following provider-specific context keys that reference the user ID that the provider returns:

- `cognito-identity.amazonaws.com:sub`
- `www.amazon.com:user_id`
- `graph.facebook.com:id`
- `accounts.google.com:sub`

For OIDC providers, use the fully qualified URL of the OIDC provider with the subcontext key, like the following example:

- `server.example.com:sub`

The following example shows a permission policy that grants access to a bucket in Amazon S3 only if the prefix for the bucket matches the string:

```
myBucket/Amazon/mynumbersgame/user1
```

The example assumes that the user is signed in using Login with Amazon, and that the user is using an app called mynumbersgame. The user's unique ID is presented as an attribute called `user_id`.

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

```
"Statement": [
    {
        "Effect": "Allow",
        "Action": ["s3>ListBucket"],
        "Resource": ["arn:aws:s3:::myBucket"],
        "Condition": {"StringLike": {"s3:prefix": ["Amazon/mynumbersgame/${www.amazon.com:user_id}/*"]}}
    },
    {
        "Effect": "Allow",
        "Action": [
            "s3:GetObject",
            "s3:PutObject",
            "s3>DeleteObject"
        ],
        "Resource": [
            "arn:aws:s3:::myBucket/amazon/mynumbersgame/${www.amazon.com:user_id}",
            "arn:aws:s3:::myBucket/amazon/mynumbersgame/${www.amazon.com:user_id}/*"
        ]
    }
]
```

You would create similar policies for users who sign in using Amazon Cognito, Facebook, Google, or another OpenID Connect-compatible IdP. Those policies would use a different provider name as part of the path as well as different app IDs.

For more information about the web identity federation keys available for condition checks in policies, see [Available Keys for AWS Web Identity Federation \(p. 678\)](#).

Additional Resources for Web Identity Federation

The following resources can help you learn more about web identity federation:

- [Amazon Cognito Identity](#) in the *AWS Mobile SDK for Android Developer Guide* and [Amazon Cognito Identity](#) in the *AWS Mobile SDK for iOS Developer Guide*.
- The [Web Identity Federation Playground](#) is an interactive website that lets you walk through the process of authenticating via Login with Amazon, Facebook, or Google, getting temporary security credentials, and then using those credentials to make a request to AWS.
- The entry [Web Identity Federation using the AWS SDK for .NET](#) on the AWS .NET Development blog walks through how to use web identity federation with Facebook and includes code snippets in C# that show how to call `AssumeRoleWithWebIdentity` and how to use the temporary security credentials from that API call to access an S3 bucket.
- The [AWS SDK for iOS](#) and the [AWS SDK for Android](#) contain sample apps. These apps include code that shows how to invoke the identity providers and then how to use the information from these providers to get and use temporary security credentials.
- The article [Web Identity Federation with Mobile Applications](#) discusses web identity federation and shows an example of how to use web identity federation to get access to content in Amazon S3.

About SAML 2.0-based Federation

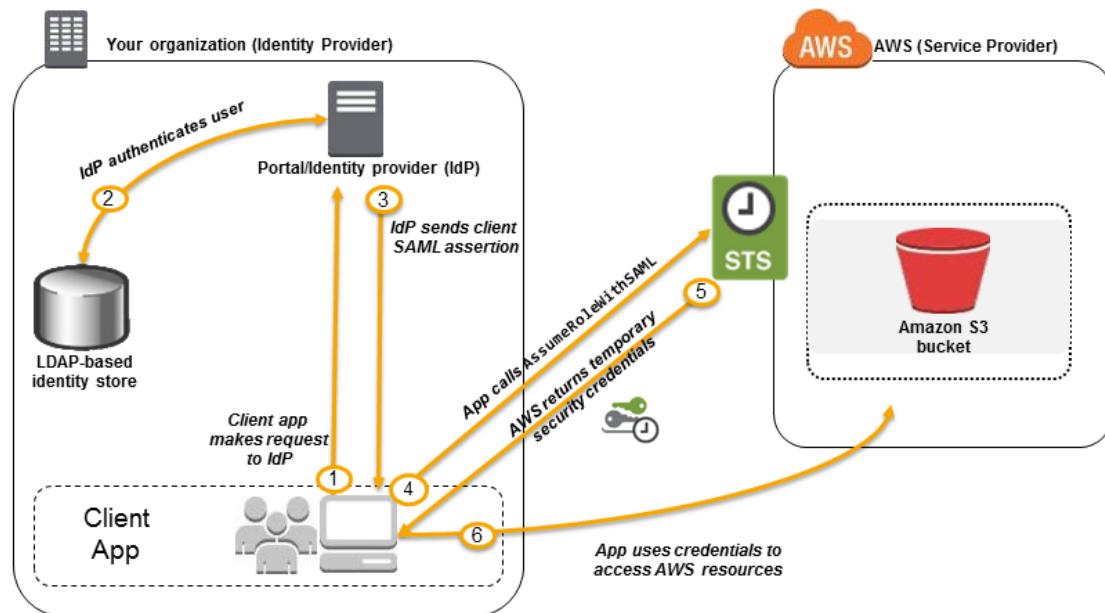
AWS supports identity federation with [SAML 2.0 \(Security Assertion Markup Language 2.0\)](#), an open standard that many identity providers (IdPs) use. This feature enables federated single sign-on (SSO), so users can log into the AWS Management Console or call the AWS API operations without you having to create an IAM user for everyone in your organization. By using SAML, you can simplify the process of configuring federation with AWS, because you can use the IdP's service instead of [writing custom identity proxy code](#).

IAM federation supports these use cases:

- **Federated access to allow a user or application in your organization to call AWS API operations (p. 191)**. You use a SAML assertion (as part of the authentication response) that is generated in your organization to get temporary security credentials. This scenario is similar to other federation scenarios that IAM supports, like those described in [Requesting Temporary Security Credentials \(p. 309\)](#) and [About Web Identity Federation \(p. 185\)](#). However, SAML 2.0-based IdPs in your organization handle many of the details at run time for performing authentication and authorization checking. This is the scenario discussed in this topic.
- **Web-based single sign-on (SSO) to the AWS Management Console from your organization (p. 211)**. Users can sign in to a portal in your organization hosted by a SAML 2.0-compatible IdP, select an option to go to AWS, and be redirected to the console without having to provide additional sign-in information. You can use a third-party SAML IdP to establish SSO access to the console or you can create a custom IdP to enable console access for your external users. For more information about building a custom IdP, see [Enabling Custom Identity Broker Access to the AWS Console \(p. 213\)](#).

Using SAML-Based Federation for API Access to AWS

Assume that you want to provide a way for employees to copy data from their computers to a backup folder. You build an application that users can run on their computers. On the back end, the application reads and writes objects in an S3 bucket. Users don't have direct access to AWS. Instead, the following process is used:



1. A user in your organization uses a client app to request authentication from your organization's IdP.
2. The IdP authenticates the user against your organization's identity store.
3. The IdP constructs a SAML assertion with information about the user and sends the assertion to the client app.
4. The client app calls the AWS STS [AssumeRoleWithSAML](#) API, passing the ARN of the SAML provider, the ARN of the role to assume, and the SAML assertion from IdP.
5. The API response to the client app includes temporary security credentials.
6. The client app uses the temporary security credentials to call Amazon S3 API operations.

Overview of Configuring SAML 2.0-Based Federation

Before you can use SAML 2.0-based federation as described in the preceding scenario and diagram, you must configure your organization's IdP and your AWS account to trust each other. The general process for configuring this trust is described in the following steps. Inside your organization, you must have an [IdP that supports SAML 2.0 \(p. 204\)](#), like Microsoft Active Directory Federation Service (AD FS, part of Windows Server), Shibboleth, or another compatible SAML 2.0 provider.

To configure your organization's IdP and AWS to trust each other

1. You begin by registering AWS with your IdP. In your organization's IdP you register AWS as a service provider (SP) by using the SAML metadata document that you get from the following URL:

`https://signin.aws.amazon.com/static/saml-metadata.xml`
2. Using your organization's IdP, you generate an equivalent metadata XML file that can describe your IdP as an IAM identity provider in AWS. It must include the issuer name, a creation date, an expiration date, and keys that AWS can use to validate authentication responses (assertions) from your organization.
3. In the IAM console, you create a SAML identity provider entity. As part of this process, you upload the SAML metadata document that was produced by the IdP in your organization in [Step 2](#). For more information, see [Creating IAM SAML Identity Providers \(p. 201\)](#).
4. In IAM, you create one or more IAM roles. In the role's trust policy, you set the SAML provider as the principal, which establishes a trust relationship between your organization and AWS. The role's permission policy establishes what users from your organization are allowed to do in AWS. For more information, see [Creating a Role for a Third-Party Identity Provider \(Federation\) \(p. 242\)](#).
5. In your organization's IdP, you define assertions that map users or groups in your organization to the IAM roles. Note that different users and groups in your organization might map to different IAM roles. The exact steps for performing the mapping depend on what IdP you're using. In the [earlier scenario \(p. 191\)](#) of an Amazon S3 folder for users, it's possible that all users will map to the same role that provides Amazon S3 permissions. For more information, see [Configuring SAML Assertions for the Authentication Response \(p. 206\)](#).

If your IdP enables SSO to the AWS console, then you can configure the maximum duration of the console sessions. For more information, see [Enabling SAML 2.0 Federated Users to Access the AWS Management Console \(p. 211\)](#).

Note

The AWS implementation of SAML 2.0 federation does not support encrypted SAML assertions between the IAM identity provider and AWS. However, the traffic between the customer's systems and AWS is transmitted over an encrypted (TLS) channel.

6. In the application that you're creating, you call the AWS Security Token Service `AssumeRoleWithSAML` API, passing it the ARN of the SAML provider you created in [Step 3](#), the ARN of the role to assume that you created in [Step 4](#), and the SAML assertion about the current user that you get from your IdP. AWS makes sure that the request to assume the role comes from the IdP referenced in the SAML provider.

For more information, see [AssumeRoleWithSAML](#) in the [AWS Security Token Service API Reference](#).

7. If the request is successful, the API returns a set of temporary security credentials, which your application can use to make signed requests to AWS. Your application has information about the current user and can access user-specific folders in Amazon S3, as described in the previous scenario.

Overview of the Role to Allow SAML-Federated Access to Your AWS Resources

The role or roles that you create in IAM define what federated users from your organization are allowed to do in AWS. When you create the trust policy for the role, you specify the SAML provider that you created earlier as the `Principal`. You can additionally scope the trust policy with a `Condition` to allow

only users that match certain SAML attributes to access the role. For example, you can specify that only users whose SAML affiliation is `staff` (as asserted by `https://openidp.feide.no`) are allowed to access the role, as illustrated by the following sample policy:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {"Federated": "arn:aws:iam::ACCOUNT-ID-WITHOUT-HYPHENS:saml-provider/ExampleOrgSSOProvider"},
            "Action": "sts:AssumeRoleWithSAML",
            "Condition": {
                "StringEquals": {
                    "saml:aud": "https://signin.aws.amazon.com/saml",
                    "saml:iss": "https://openidp.feide.no"
                },
                "ForAllValues:StringLike": {"saml:edupersonaffiliation": ["staff"]}
            }
        }
    ]
}
```

For more information about the SAML keys that you can check in a policy, see [Available Keys for SAML-Based AWS STS Federation \(p. 681\)](#).

For the permission policy in the role, you specify permissions as you would for any role. For example, if users from your organization are allowed to administer Amazon Elastic Compute Cloud instances, you must explicitly allow Amazon EC2 actions in the permissions policy, such as those in the [AmazonEC2FullAccess](#) managed policy.

Uniquely Identifying Users in SAML-Based Federation

When you create access policies in IAM, it's often useful to be able to specify permissions based on the identity of users. For example, for users who have been federated using SAML, an application might want to keep information in Amazon S3 using a structure like this:

```
myBucket/app1/user1
myBucket/app1/user2
myBucket/app1/user3
```

You can create the bucket (`myBucket`) and folder (`app1`) through the Amazon S3 console or the AWS CLI, since those are static values. However, the user-specific folders (`user1`, `user2`, `user3`, etc.) have to be created at run time using code, since the value that identifies the user isn't known until the first time the user signs in through the federation process.

To write policies that reference user-specific details as part of a resource name, the user identity has to be available in SAML keys that can be used in policy conditions. The following keys are available for SAML 2.0-based federation for use in IAM policies. You can use the values returned by the following keys to create unique user identifiers for resources like Amazon S3 folders.

- `saml:namequalifier`. A hash value based on the concatenation of the `Issuer` response value (`saml:iss`) and a string with the AWS account ID and the friendly name (the last part of the ARN) of the SAML provider in IAM. The concatenation of the account ID and friendly name of the SAML provider is available to IAM policies as the key `saml:doc`. The account ID and provider name must be separated by a '/' as in "123456789012/provider_name". For more information, see the `saml:doc` key at [Available Keys for SAML-Based AWS STS Federation \(p. 681\)](#).

The combination of `NameQualifier` and `Subject` can be used to uniquely identify a federated user. The following pseudocode shows how this value is calculated. In this pseudocode `+ indicates concatenation`, `SHA1` represents a function that produces a message digest using SHA-1, and `Base64` represents a function that produces Base-64 encoded version of the hash output.

```
Base64 ( SHA1 ( "https://example.com/saml" + "123456789012" + "/" + "MySAMLIdP" ) )
```

For more information about the policy keys that are available for SAML-based federation, see [Available Keys for SAML-Based AWS STS Federation \(p. 681\)](#).

- `saml:sub` (string). This is the subject of the claim, which includes a value that uniquely identifies an individual user within an organization (for example, `_cbb8bf52c2510eabe00c1642d4643f41430fe25e3`).
- `saml:sub_type` (string). This key can be `persistent`, `transient`, or the full Format URI from the `Subject` and `NameID` elements used in your SAML assertion. A value of `persistent` indicates that the value in `saml:sub` is the same for a user across all sessions. If the value is `transient`, the user has a different `saml:sub` value for each session. For information about the `NameID` element's `Format` attribute, see [Configuring SAML Assertions for the Authentication Response \(p. 206\)](#).

The following example shows a permission policy that uses the preceding keys to grant permissions to a user-specific folder in Amazon S3. The policy assumes that the Amazon S3 objects are identified using a prefix that includes both `saml:namequalifier` and `saml:sub`. Notice that the `Condition` element includes a test to be sure that `saml:sub_type` is set to `persistent`. If it is set to `transient`, the `saml:sub` value for the user can be different for each session, and the combination of values should not be used to identify user-specific folders.

```
>{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:PutObject",
                "s3:DeleteObject"
            ],
            "Resource": [
                "arn:aws:s3:::exampleorgBucket/backup/${saml:namequalifier}/${saml:sub}",
                "arn:aws:s3:::exampleorgBucket/backup/${saml:namequalifier}/${saml:sub}/*"
            ],
            "Condition": {"StringEquals": {"saml:sub_type": "persistent"}}
        }
    ]
}
```

For more information about mapping assertions from the IdP to policy keys, see [Configuring SAML Assertions for the Authentication Response \(p. 206\)](#).

Creating IAM Identity Providers

When you want to configure federation with an external identity provider (IdP) service, you create an IAM *identity provider* to inform AWS about the IdP and its configuration. This establishes "trust" between your AWS account and the IdP. The following topics include details about how to create an IAM identity provider for each of the IdP types.

Topics

- [Creating OpenID Connect \(OIDC\) Identity Providers \(p. 194\)](#)
- [Creating IAM SAML Identity Providers \(p. 201\)](#)

Creating OpenID Connect (OIDC) Identity Providers

IAM *OIDC identity providers* are entities in IAM that describe an external identity provider (IdP) service that supports the [OpenID Connect](#) (OIDC) standard, such as Google or Salesforce. You use an IAM OIDC

identity provider when you want to establish trust between an OIDC-compatible IdP and your AWS account. This is useful when creating a mobile app or web application that requires access to AWS resources, but you don't want to create custom sign-in code or manage your own user identities. For more information about this scenario, see the section called "About Web Identity Federation" (p. 185).

You can create and manage an IAM OIDC identity provider using the AWS Management Console, the AWS Command Line Interface, the Tools for Windows PowerShell, or the IAM API.

Topics

- [Creating and Managing an OIDC Provider \(Console\) \(p. 195\)](#)
- [Creating and Managing an IAM OIDC Identity Provider \(AWS CLI\) \(p. 196\)](#)
- [Creating and Managing an OIDC Identity Provider \(AWS API\) \(p. 197\)](#)
- [Obtaining the Root CA Thumbprint for an OpenID Connect Identity Provider \(p. 198\)](#)

[Creating and Managing an OIDC Provider \(Console\)](#)

Follow these instructions to create and manage an IAM OIDC identity provider in the AWS Management Console.

To create an IAM OIDC identity provider (console)

1. Before you create an IAM OIDC identity provider, you must register your application with the IdP to receive a *client ID*. The client ID (also known as *audience*) is a unique identifier for your app that is issued to you when you register your app with the IdP. For more information about obtaining a client ID, see the documentation for your IdP.
2. Open the IAM console at <https://console.aws.amazon.com/iam/>.
3. In the navigation pane, choose **Identity Providers**, and then choose **Create Provider**.
4. For **Provider Type**, choose **Choose a provider type**, and then choose **OpenID Connect**.
5. For **Provider URL**, type the URL of the IdP. The URL must comply with these restrictions:
 - The URL is case-sensitive.
 - The URL must begin with **https://**.
 - The URL cannot include a colon (:) character, and therefore cannot specify a port number. This means that the server must be listening on the default port 443.
 - Within your AWS account, each IAM OIDC identity provider must use a unique URL.
6. For **Audience**, type the client ID of the application that you registered with the IdP and received in Step 1, and that will make requests to AWS. If you have additional client IDs (also known as *audiences*) for this IdP, you can add them later on the provider detail page. Choose **Next Step**.
7. Use the **Thumbprint** to verify the server certificate of your IdP. To learn how, see [Obtaining the Root CA Thumbprint for an OpenID Connect Identity Provider \(p. 198\)](#). Choose **Create**.
8. In the confirmation message at the top of the screen, choose **Do this now** to go to the **Roles** tab to create a role for this identity provider. For more information about creating a role for an OIDC identity provider, see [Creating a Role for a Third-Party Identity Provider \(Federation\) \(p. 242\)](#). OIDC identity providers must have a role in order to access your AWS account. To skip this step and create the role later, choose **Close**.

To add or remove a thumbprint or client ID (also known as audience) for an IAM OIDC identity provider (console)

1. Open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Identity Providers**, then choose the name of the IAM identity provider that you want to update.

3. To add a thumbprint or audience, choose **Add a Thumbprint** or **Add an Audience**. To remove a thumbprint or audience, choose **Remove** next to the item that you want to remove.

Note

An IAM OIDC identity provider must have at least one and can have a maximum of five thumbprints. An OIDC identity provider must have at least one and can have a maximum of 100 audiences.

When you are done, choose **Save Changes**.

To delete an IAM OIDC identity provider (console)

1. Open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Identity Providers**.
3. Select the check box next to the IAM identity provider that you want to delete.
4. Choose **Delete Providers**.

Creating and Managing an IAM OIDC Identity Provider (AWS CLI)

You can use the following AWS CLI commands to create and manage IAM OIDC identity providers.

To create an IAM OIDC identity provider (AWS CLI)

1. (Optional) To get a list of all the IAM OIDC identity providers in your AWS account, run the following command:
 - `aws iam list-open-id-connect-providers`
2. To create a new IAM OIDC identity provider, run the following command:
 - `aws iam create-open-id-connect-provider`

To update the list of server certificate thumbprints for an existing IAM OIDC identity provider (AWS CLI)

- To update the list of server certificate thumbprints for an IAM OIDC identity provider, run the following command:
 - `aws iam update-open-id-connect-provider-thumbprint`

To add or remove a client ID from an existing IAM OIDC identity provider (AWS CLI)

1. (Optional) To get a list of all the IAM OIDC identity provider in your AWS account, run the following command:
 - `aws iam list-open-id-connect-providers`
2. (Optional) To get detailed information about an IAM OIDC identity provider, run the following command:
 - `aws iam get-open-id-connect-provider`
3. To add a new client ID to an existing IAM OIDC identity provider, run the following command:
 - `aws iam add-client-id-to-open-id-connect-provider`
4. To remove a client from an existing IAM OIDC identity provider, run the following command:
 - `aws iam remove-client-id-from-open-id-connect-provider`

To delete an IAM OIDC identity provider (AWS CLI)

1. (Optional) To get a list of all the IAM OIDC identity provider in your AWS account, run the following command:
 - `aws iam list-open-id-connect-providers`
2. (Optional) To get detailed information about an IAM OIDC identity provider, run the following command:
 - `aws iam get-open-id-connect-provider`
3. To delete an IAM OIDC identity provider, run the following command:
 - `aws iam delete-open-id-connect-provider`

Creating and Managing an OIDC Identity Provider (AWS API)

You can use the following IAM API commands to create and manage OIDC providers.

To create an IAM OIDC identity provider (AWS API)

1. (Optional) To get a list of all the IAM OIDC identity provider in your AWS account, call the following operation:
 - `ListOpenIDConnectProviders`
2. To create a new IAM OIDC identity provider, call the following operation:
 - `CreateOpenIDConnectProvider`

To update the list of server certificate thumbprints for an existing IAM OIDC identity provider (AWS API)

- To update the list of server certificate thumbprints for an IAM OIDC identity provider, call the following operation:
 - `UpdateOpenIDConnectProviderThumbprint`

To add or remove a client ID from an existing IAM OIDC identity provider (AWS API)

1. (Optional) To get a list of all the IAM OIDC identity provider in your AWS account, call the following operation:
 - `ListOpenIDConnectProviders`
2. (Optional) To get detailed information about an IAM OIDC identity provider, call the following operation:
 - `GetOpenIDConnectProvider`
3. To add a new client ID to an existing IAM OIDC identity provider, call the following operation:
 - `AddClientIDToOpenIDConnectProvider`
4. To remove a client ID from an existing IAM OIDC identity provider, call the following operation:
 - `RemoveClientIDFromOpenIDConnectProvider`

To delete an IAM OIDC identity provider (AWS API)

1. (Optional) To get a list of all the IAM OIDC identity provider in your AWS account, call the following operation:
 - [ListOpenIDConnectProviders](#)
2. (Optional) To get detailed information about an IAM OIDC identity provider, call the following operation:
 - [GetOpenIDConnectProvider](#)
3. To delete an IAM OIDC identity provider, call the following operation:
 - [DeleteOpenIDConnectProvider](#)

Obtaining the Root CA Thumbprint for an OpenID Connect Identity Provider

When you [create an OpenID Connect \(OIDC\) identity provider \(p. 194\)](#) in IAM, you must supply a thumbprint. IAM requires the thumbprint for the root certificate authority (CA) that signed the certificate used by the external identity provider (IdP). The thumbprint is a signature for the CA's certificate that was used to issue the certificate for the OIDC-compatible IdP. When you create an IAM OIDC identity provider, you are trusting identities authenticated by that IdP to have access to your AWS account. By supplying the CA's certificate thumbprint, you trust any certificate issued by that CA with the same DNS name as the one registered. This eliminates the need to update trusts in each account when you renew the IdP's signing certificate.

Important

In most cases, the federation server uses two different certificates. The first establishes an HTTPS connection between the clients and the federation endpoint. This can be safely issued by a public root CA, such as AWS Certificate Manager. The second is used to sign tokens. We recommend that you issue this using a private CA.

You can create an IAM OIDC identity provider with [the AWS Command Line Interface, the Tools for Windows PowerShell, or the IAM API \(p. 196\)](#). When you use these methods, you must obtain the thumbprint manually and supply it to AWS. When you create an OIDC identity provider with [the IAM console \(p. 194\)](#), the console attempts to fetch the thumbprint for you. We recommend that you also obtain the thumbprint for your OIDC IdP manually and verify that the console fetched the correct thumbprint.

You use a web browser and the OpenSSL command line tool to obtain the thumbprint for an OIDC provider. For more information, see the following sections.

To obtain the thumbprint for an OIDC IdP

1. Before you can obtain the thumbprint for an OIDC IdP, you need to obtain the OpenSSL command-line tool. You use this tool to download the OIDC IdP's certificate chain and produce a thumbprint of the final certificate in the certificate chain. If you need to install and configure OpenSSL, follow the instructions at [Install OpenSSL \(p. 200\)](#) and [Configure OpenSSL \(p. 200\)](#).
2. Start with the OIDC IdP's URL (for example, `https://server.example.com`), and then add `/well-known/openid-configuration` to form the URL for the IdP's configuration document, such as the following:

`https://server.example.com/.well-known/openid-configuration`

Open this URL in a web browser, replacing `server.example.com` with your IdP's server name.

3. In the document displayed in your web browser, find "jwks_uri". (Use your web browser's **Find** feature to locate this text on the page.) Immediately following the text "jwks_uri" you will see a colon (:) followed by a URL. Copy the fully qualified domain name of the URL. Do not include the `https://` or any path that comes after the top-level domain.

4. Use the OpenSSL command line tool to execute the following command. Replace `keys.example.com` with the domain name you obtained in [Step 3](#).

```
openssl s_client -servername keys.example.com -showcerts -connect keys.example.com:443
```

5. In your command window, scroll up until you see a certificate similar to the following example. If you see more than one certificate, find the last certificate that is displayed (at the bottom of the command output). This will be the certificate of the root CA in the certificate authority chain.

```
-----BEGIN CERTIFICATE-----  
MIICiTCCAFICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC  
VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6  
b24xFDASBgNVBAsTC01BTSDb25zb2x1MRIwEAYDVQQDEwlUZXN0Q2lsYWmxHzAd  
BgkqhkiG9w0BCQEWEGBv251QGftXXpvb5jb20whRcNMTEwNDI1MjAONTIxWhCN  
MTIwNDI0MjAONTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAlDbMRawDgYD  
VQQHEwdTZWF0dGx1MQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC01BTSDb25z  
b2x1MRIwEAYDVQQDEwlUZXN0Q2lsYWmxHzAdBgkqhkiG9w0BCQEWEGBv251QGft  
YXpvb5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ  
21uUSfwfEvysWtC2XADZ4nB+BLYgVlk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T  
rDHudUZg3qX4waLG5M43q7Wgc/MbQITxOUSQv7c7ugFFDzQGBzzswY6786m86gpE  
Ibb3OhjZnzcvQAArHhd1QWI Mm2nrAgMBAAwDQYJKoZIhvcNAQEFBQADgYEAtCu4  
nUhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb  
FFBjvSfpJ11J00zbhNYS5f6GuoEDmFJ10ZxBHjJnyp3780D8uTs7fLvjx79LjStb  
NYiytVbZPQUQ5Yaxu2jXnimvv3rrszlaEXAMPLE=-----END CERTIFICATE-----
```

Copy the certificate (including the -----BEGIN CERTIFICATE----- and -----END CERTIFICATE----- lines) and paste it into a text file. Then save the file with the file name `certificate.crt`.

6. Use the OpenSSL command-line tool to execute the following command.

```
openssl x509 -in certificate.crt -fingerprint -noout
```

Your command window displays the certificate thumbprint, which looks similar to the following example:

```
SHA1 Fingerprint=99:0F:41:93:97:2F:2B:EC:F1:2D:DE:DA:52:37:F9:C9:52:F2:0D:9E
```

Remove the colon characters (:) from this string to produce the final thumbprint, like this:

```
990F4193972F2BECF12DDEDA5237F9C952F20D9E
```

7. If you are creating the IAM OIDC identity provider with the AWS CLI, Tools for Windows PowerShell, or the IAM API, supply this thumbprint when creating the provider.

If you are creating the IAM OIDC identity provider in the IAM console, compare this thumbprint to the thumbprint shown on the console [Verify Provider Information](#) page when you create an OIDC provider.

Important

If the thumbprint you obtained does not match the one you see in the console, you should not create the OIDC provider in the console. Instead, you should wait a while and then try again to create the OIDC provider, ensuring that the thumbprints match before you create the provider. If the thumbprints still do not match after a second attempt, use the [IAM Forum](#) to contact AWS.

Install OpenSSL

If you don't already have OpenSSL installed, follow the instructions in this section.

To install OpenSSL on Linux or Unix

1. Go to [OpenSSL: Source, Tarballs](https://openssl.org/source/) (<https://openssl.org/source/>).
2. Download the latest source and build the package.

To install OpenSSL on Windows

1. Go to [OpenSSL: Binary Distributions](https://wiki.openssl.org/index.php/Binaries) (<https://wiki.openssl.org/index.php/Binaries>) for a list of sites from which you can install the Windows version.
2. Follow the instructions on your selected site to start the installation.
3. If you are asked to install the **Microsoft Visual C++ 2008 Redistributables** and it is not already installed on your system, choose the download link appropriate for your environment. Follow the instructions provided by the **Microsoft Visual C++ 2008 Redistributable Setup Wizard**.

Note

If you are not sure whether the Microsoft Visual C++ 2008 Redistributables is already installed on your system, you can try installing OpenSSL first. The OpenSSL installer displays an alert if the Microsoft Visual C++ 2008 Redistributables is not yet installed. Make sure that you install the architecture (32-bit or 64-bit) that matches the version of OpenSSL that you install.

4. After you have installed the Microsoft Visual C++ 2008 Redistributables, select the appropriate version of the OpenSSL binaries for your environment and save the file locally. Start the **OpenSSL Setup Wizard**.
5. Follow the instructions described in the **OpenSSL Setup Wizard**.

Configure OpenSSL

Before you use OpenSSL commands, you must configure the operating system so that it has information about the location where OpenSSL is installed.

To configure OpenSSL on Linux or Unix

1. At the command line, set the `OpenSSL_HOME` variable to the location of the OpenSSL installation:

```
$ export OpenSSL_HOME=path_to_your_OpenSSL_installation
```

2. Set the path to include the OpenSSL installation:

```
$ export PATH=$PATH:$OpenSSL_HOME/bin
```

Note

Any changes you make to environment variables with the `export` command are valid only for the current session. You can make persistent changes to the environment variables by setting them in your shell configuration file. For more information, see the documentation for your operating system.

To configure OpenSSL on Windows

1. Open a **Command Prompt** window.
2. Set the `OpenSSL_HOME` variable to the location of the OpenSSL installation:

```
C:\> set OpenSSL_HOME=path_to_your_OpenSSL_installation
```

3. Set the OpenSSL_CONF variable to the location of the configuration file in your OpenSSL installation:

```
C:\> set OpenSSL_CONF=path_to_your_OpenSSL_installation\bin\openssl.cfg
```

4. Set the path to include the OpenSSL installation:

```
C:\> set Path=%Path%;%OpenSSL_HOME%\bin
```

Note

Any changes you make to Windows environment variables in a **Command Prompt** window are valid only for the current command line session. You can make persistent changes to the environment variables by setting them as system properties. The exact procedures depend on what version of Windows you're using. (For example, in Windows 7, open **Control Panel**, **System and Security**, **System**. Then choose **Advanced system settings**, **Advanced tab**, **Environment Variables**.) For more information, see the Windows documentation.

Creating IAM SAML Identity Providers

An IAM SAML 2.0 identity provider is an entity in IAM that describes an external identity provider (IdP) service that supports the [SAML 2.0 \(Security Assertion Markup Language 2.0\)](#) standard. You use an IAM identity provider when you want to establish trust between a SAML-compatible IdP such as Shibboleth or Active Directory Federation Services and AWS, so that users in your organization can access AWS resources. IAM SAML identity providers are used as principals in an IAM trust policy.

For more information about this scenario, see [About SAML 2.0-based Federation \(p. 190\)](#).

You can create and manage an IAM identity provider in the AWS Management Console or with AWS CLI, Tools for Windows PowerShell, or AWS API calls.

After you create a SAML provider, you must create one or more IAM roles. A role is an identity in AWS that doesn't have its own credentials (as a user does). But in this context, a role is dynamically assigned to a federated user that is authenticated by your organization's IdP. The role permits your organization's IdP to request temporary security credentials for access to AWS. The policies assigned to the role determine what the federated users are allowed to do in AWS. To create a role for SAML federation, see [Creating a Role for a Third-Party Identity Provider \(Federation\) \(p. 242\)](#).

Finally, after you create the role, you complete the SAML trust by configuring your IdP with information about AWS and the roles that you want your federated users to use. This is referred to as configuring relying party trust between your IdP and AWS. To configure relying party trust, see [Configuring your SAML 2.0 IdP with Relying Party Trust and Adding Claims \(p. 203\)](#).

Topics

- [Creating and Managing an IAM Identity Provider \(Console\) \(p. 202\)](#)
- [Creating and Managing an IAM SAML Identity Provider \(AWS CLI\) \(p. 202\)](#)
- [Creating and Managing an IAM SAML Identity Provider \(AWS API\) \(p. 203\)](#)
- [Configuring your SAML 2.0 IdP with Relying Party Trust and Adding Claims \(p. 203\)](#)
- [Integrating Third-Party SAML Solution Providers with AWS \(p. 204\)](#)
- [Configuring SAML Assertions for the Authentication Response \(p. 206\)](#)

Creating and Managing an IAM Identity Provider (Console)

You can use the AWS Management Console to create and delete IAM SAML identity providers.

To create an IAM identity provider (console)

1. Before you can create an IAM identity provider, you need the SAML metadata document that you get from the IdP. This document includes the issuer's name, expiration information, and keys that can be used to validate the SAML authentication response (assertions) that are received from the IdP. To generate the metadata document, use the identity management software your organization uses as its IdP. For instructions on how to configure many of the available IdPs to work with AWS, including how to generate the required SAML metadata document, see [Integrating Third-Party SAML Solution Providers with AWS \(p. 204\)](#).

Important

The metadata file must be encoded in UTF-8 format without a byte order mark (BOM). Also, the x.509 certificate that is included as part of the SAML metadata document must use a key size of at least 1024 bits. If the key size is smaller, the IdP creation fails with an "Unable to parse metadata" error. To remove the BOM, you can encode the file as UTF-8 using a text editing tool, such as Notepad++.

2. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
3. In the navigation pane, click **Identity Providers** and then click **Create Provider**.
4. For **Provider Type**, click **Choose a provider type** and click **SAML**.
5. Type a name for the identity provider.
6. For **Metadata Document**, click **Choose File**, specify the SAML metadata document that you downloaded in Step 1, and click **Open**. Click **Next Step**.
7. Verify the information that you have provided, and click **Create**.

To delete a SAML provider (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, click **Identity Providers**.
3. Select the check box next to the identity provider that you want to delete.
4. Click **Delete Providers**.

Creating and Managing an IAM SAML Identity Provider (AWS CLI)

You can use the AWS CLI to create and manage SAML providers.

To create an IAM identity provider and upload a metadata document (AWS CLI)

- Run this command: `aws iam create-saml-provider`

To upload a new metadata document for an IAM identity provider (AWS CLI)

- Run this command: `aws iam update-saml-provider`

To delete an IAM SAML identity provider (AWS CLI)

1. (Optional) To list information for all providers, such as the ARN, creation date, and expiration, run the following command:

- `aws iam list-saml-providers`
2. (Optional) To get information about a specific provider, such as the ARN, creation date, and expiration, run the following command:
 - `aws iam get-saml-provider`
 3. To delete an IAM identity provider, run the following command:
 - `aws iam delete-saml-provider`

Creating and Managing an IAM SAML Identity Provider (AWS API)

You can use the AWS API to create and manage SAML providers.

To create an IAM identity provider and upload a metadata document (AWS API)

- Call this operation: [CreateSAMLProvider](#)

To upload a new metadata document for an IAM identity provider (AWS API)

- Call this operation: [UpdateSAMLProvider](#)

To delete an IAM identity provider (AWS API)

1. (Optional) To list information for all IdPs, such as the ARN, creation date, and expiration, call the following operation:
 - [ListSAMLProviders](#)
2. (Optional) To get information about a specific provider, such as the ARN, creation date, and expiration, call the following operation:
 - [GetSAMLProvider](#)
3. To delete an IdP, call the following operation:
 - [DeleteSAMLProvider](#)

Configuring your SAML 2.0 IdP with Relying Party Trust and Adding Claims

When you create an IAM identity provider and role for SAML access, you are telling AWS about the external identity provider (IdP) and what its users are allowed to do. Your next step is to then tell the IdP about AWS as a service provider. This is called adding *relying party trust* between your IdP and AWS. The exact process for adding relying party trust depends on what IdP you're using. For details, see the documentation for your identity management software.

Many IdPs allow you to specify a URL from which the IdP can read an XML document that contains relying party information and certificates. For AWS, you can use <https://signin.aws.amazon.com/static/saml-metadata.xml>

If you can't specify a URL directly, then download the XML document from the preceding URL and import it into your IdP software.

You also need to create appropriate claim rules in your IdP that specify AWS as a relying party. When the IdP sends a SAML response to the AWS endpoint, it includes a SAML *assertion* that contains one or more *claims*. A claim is information about the user and its groups. A claim rule maps that information into SAML attributes. This lets you make sure that SAML authentication responses from your IdP contain

the necessary attributes that AWS uses in IAM policies to check permissions for federated users. For more information, see the following topics:

- [Overview of the Role to Allow SAML-Federated Access to Your AWS Resources \(p. 192\)](#). This topic discusses using SAML-specific keys in IAM policies and how to use them to restrict permissions for SAML-federated users.
- [Configuring SAML Assertions for the Authentication Response \(p. 206\)](#). This topic discusses how to configure SAML claims that include information about the user. The claims are bundled into a SAML assertion and included in the SAML response that is sent to AWS. You must ensure that the information needed by AWS policies is included in the SAML assertion in a form that AWS can recognize and use.
- [Integrating Third-Party SAML Solution Providers with AWS \(p. 204\)](#). This topic provides links to documentation provided by third-party organizations about how to integrate identity solutions with AWS.

Integrating Third-Party SAML Solution Providers with AWS

The following links help you configure third-party SAML 2.0 identity provider (IdP) solutions to work with AWS federation.

Note

AWS Support engineers can assist customers who have business and enterprise support plans with some integration tasks that involve third-party software. For a current list of supported platforms and applications, see [What third-party software is supported?](#) in the *AWS Support FAQs*.

Solution	More information
Auth0	AWS Integration in Auth0 – This page on the Auth0 documentation website describes how to set up single sign-on (SSO) with the AWS Management Console and includes a JavaScript example. You can configure Auth0 to pass session tags (p. 299) . For more information, see Auth0 Announces Partnership with AWS for IAM Session Tags .
Bitium	Configuring SAML for Amazon Web Services (AWS) – This article on the Bitium support site explains how to use Bitium to set up AWS with SAML SSO.
Centrify	Configure Centrify and Use SAML for SSO to AWS – This page on the Centrify website explains how to configure Centrify to use SAML for SSO to AWS.
Clearlogin	Amazon Web Services Setup – This article in the Clearlogin Help Center explains how to set up SSO functionality between Clearlogin and AWS.
ForgeRock	The ForgeRock Identity Platform integrates with AWS. You can configure ForgeRock to pass session tags (p. 299) . For more information, see Attribute Based Access Control for Amazon Web Services .
Google G Suite	Amazon Web Services cloud application – This article on the Google G Suite Administrator Help site describes how to configure G Suite as a SAML 2.0 IdP with AWS as the service provider.

Solution	More information
IBM	You can configure IBM to pass session tags (p. 299) . For more information, see IBM Cloud Identity IDaaS one of first to support AWS session tags .
Identacor	Configuring SSO (SAML) for AWS – This article on the Identacor website describes how to set up and enable SSO for AWS.
Matrix42	MyWorkspace Getting Started Guide – This guide describes how to integrate AWS identity services with Matrix42 MyWorkspace.
Microsoft Active Directory Federation Services (AD FS)	<p>Enabling Federation to AWS Using Windows Active Directory, AD FS, and SAML 2.0 – This post on the AWS Security Blog shows how to set up AD FS on an EC2 instance and enable SAML federation with AWS. You can configure AD FS to pass session tags (p. 299). For more information, see Use attribute-based access control with AD FS to simplify IAM permissions management.</p> <p>PowerShell Automation to Give AWS Console Access – This post on Sivaprasad Padisetty's blog describes how to use Windows PowerShell to automate the process of setting up Active Directory and AD FS. It also covers enabling SAML federation with AWS.</p>
miniOrange	SSO for AWS – This page on the miniOrange website describes how to establish secure access to AWS for enterprises and full control over access of AWS applications.
MIRACL	Setting up MIRACL Trust SSO as an Identity Provider within AWS – This page on the MIRACL website explains how to configure an AWS service provider for SSO authentication with MIRACL Trust SSO SAML.
Okta	Integrating the Amazon Web Services Command Line Interface Using Okta – From this page on the Okta support site you can learn how to configure Okta for use with AWS. You can configure Okta to pass session tags (p. 299) . For more information, see Okta and AWS Partner to Simplify Access Via Session Tags .
OneLogin	From the OneLogin Knowledgebase , search for SAML AWS for a list of articles that explain how to set up AWS SSO functionality between OneLogin and AWS for a single-role and multi-role scenarios. You can configure OneLogin to pass session tags (p. 299) . For more information, see OneLogin and Session Tags: Attribute-Based Access Control for AWS Resources .

Solution	More information
Ping Identity	PingFederate AWS Connector – View details about the PingFederate AWS Connector, a quick connection template to easily set up a single sign-on (SSO) and provisioning connection. Read documentation and download the latest PingFederate AWS Connector for integrations with AWS. You can configure Ping Identity to pass session tags (p. 299) . For more information, see Announcing Ping Identity Support for Attribute-Based Access Control in AWS .
RadiantLogic	Radiant Logic Technology Partners – Radiant Logic's RadiantOne Federated Identity Service integrates with AWS to provide an identity hub for SAML-based SSO.
RSA	RSA Link is an online community that facilitates information sharing and discussion. You can configure RSA to pass session tags (p. 299) . For more information, see Simplify Identity Access and Assurance Decisions on AWS with RSA SecurID and Session Tags .
Salesforce.com	How to configure SSO from Salesforce to AWS – This how-to article on the Salesforce.com developer site describes how to set up an identity provider (IdP) in Salesforce and configure AWS as a service provider.
SecureAuth	AWS - SecureAuth SAML SSO – This article on the SecureAuth website describes how to set up SAML integration with AWS for a SecureAuth appliance.
Shibboleth	How to Use Shibboleth for SSO to the AWS Management Console – This entry on the AWS Security Blog provides a step-by-step tutorial on how to set up Shibboleth and configure it as an identity provider for AWS. You can configure Shibboleth to pass session tags (p. 299) .

For more details, see the [IAM Partners](#) page on the AWS website.

Configuring SAML Assertions for the Authentication Response

In your organization, after a user's identity has been verified, the external identity provider (IdP) sends an authentication response to the AWS SAML endpoint at <https://signin.aws.amazon.com/saml>. This response is a POST request that includes a SAML token that adheres to the [HTTP POST Binding for SAML 2.0](#) standard and that contains the following elements, or *claims*. You configure these claims in your SAML-compatible IdP. Refer to the documentation for your IdP for instructions on how to enter these claims.

When the IdP sends the response containing the claims to AWS, many of the incoming claims map to AWS context keys. These context keys can be checked in IAM policies using the Condition element. A listing of the available mappings follows in the section [Mapping SAML Attributes to AWS Trust Policy Context Keys \(p. 209\)](#).

Subject and NameID

The following excerpt shows an example. Substitute your own values for the marked ones. There must be exactly one `SubjectConfirmation` element with a `SubjectConfirmationData` element that includes both the `NotOnOrAfter` attribute and a `Recipient` attribute. These attributes include a value that must match the AWS endpoint (<https://signin.aws.amazon.com/saml>), as shown in the

following example. For information about the name identifier formats supported for single sign-on interactions, see [Oracle Sun OpenSSO Enterprise Administration Reference](#).

```
<Subject>
  <NameID Format="urn:oasis:names:tc:SAML:2.0:nameid-
format:persistent">_cbb88bf52c2510eabe00c1642d4643f41430fe25e3</NameID>
  <SubjectConfirmation Method="urn:oasis:names:tc:SAML:2.0:cm:bearer">
    <SubjectConfirmationData NotOnOrAfter="2013-11-05T02:06:42.876Z" Recipient="https://
signin.&home-domain;/saml"/>
  </SubjectConfirmation>
</Subject>
```

AudienceRestriction and Audience

For security reasons, AWS should be included as an audience in the SAML assertion your IdP sends to AWS. For the value of the Audience element, specify either `https://signin.aws.amazon.com/saml` or `urn:amazon:webservices`. The following sample XML snippets from SAML assertions show how this key can be specified by the IdP. Include whichever sample applies to your use case.

```
<Conditions>
  <AudienceRestriction>
    <Audience>https://signin.&home-domain;/saml</Audience>
  </AudienceRestriction>
</Conditions>
```

```
<Conditions>
  <AudienceRestriction>
    <Audience>urn:amazon:webservices</Audience>
  </AudienceRestriction>
</Conditions>
```

Important

The SAML AudienceRestriction value in the SAML assertion from the IdP does *not* map to the `saml:aud` context key that you can test in an IAM policy. Instead, the `saml:aud` context key comes from the SAML `recipient` attribute because it is the SAML equivalent to the OIDC audience field, for example, by `accounts.google.com:aud`.

SAML Role Attribute

You can use an Attribute element with the Name attribute set to `https://aws.amazon.com/SAML/Attributes/Role`. This element contains one or more AttributeValue elements that list the IAM identity provider and role to which the user is mapped by your IdP. The IAM role and IAM identity provider are specified as a comma-delimited pair of ARNs in the same format as the `RoleArn` and `PrincipalArn` parameters that are passed to [AssumeRoleWithSAML](#). This element must contain at least one role-provider pair (AttributeValue element), and can contain multiple pairs. If the element contains multiple pairs, then the user is asked to select which role to assume when they use WebSSO to sign into the AWS Management Console.

Important

The value of the Name attribute in the Attribute tag is case-sensitive. It must be set to `https://aws.amazon.com/SAML/Attributes/Role` exactly.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/Role">
  <AttributeValue>arn:aws:iam::account-number:role/role-name1,arn:aws:iam::account-
number:saml-provider/provider-name</AttributeValue>
  <AttributeValue>arn:aws:iam::account-number:role/role-name2,arn:aws:iam::account-
number:saml-provider/provider-name</AttributeValue>
  <AttributeValue>arn:aws:iam::account-number:role/role-name3,arn:aws:iam::account-
number:saml-provider/provider-name</AttributeValue>
```

```
</Attribute>
```

SAML RoleSessionName Attribute

You can use an `Attribute` element with the `Name` attribute set to `https://aws.amazon.com/SAML/Attributes/RoleSessionName`. This element contains one `AttributeValue` element that provides an identifier for the AWS temporary credentials that are issued for SSO. This element is used to display user information in the AWS Management Console. The value in the `AttributeValue` element must be between 2 and 64 characters long, can contain only alphanumeric characters, underscores, and the following characters: + (plus sign), = (equals sign), , (comma), . (period), @ (at symbol), and - (hyphen). It cannot contain spaces. The value is typically a user ID (`johndoe`) or an email address (`johndoe@example.com`). It should not be a value that includes a space, like a user's display name (John Doe).

Important

The value of the `Name` attribute in the `Attribute` tag is case-sensitive. It must be set to `https://aws.amazon.com/SAML/Attributes/RoleSessionName` exactly.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/RoleSessionName">
  <AttributeValue>user-id-name</AttributeValue>
</Attribute>
```

SAML SessionDuration Attribute

(Optional) You can use an `Attribute` element with the `Name` attribute set to `https://aws.amazon.com/SAML/Attributes/SessionDuration`. This element contains one `AttributeValue` element that specifies how long the user can access the AWS Management Console before having to request new temporary credentials. The value is an integer representing the number of seconds for the session. The value can range from 900 seconds (15 minutes) to 43200 seconds (12 hours). If this attribute is not present, then the credential last for one hour (the default value of the `DurationSeconds` parameter of the `AssumeRoleWithSAML` API).

To use this attribute, you must configure the SAML provider to provide single sign-on access to the AWS Management Console through the console sign-in web endpoint at `https://signin.aws.amazon.com/saml`. Note that this attribute extends sessions only to the AWS Management Console. It cannot extend the lifetime of other credentials. However, if it is present in an `AssumeRoleWithSAML` API call, it can be used to *shorten* the duration of the session. The default lifetime of the credentials returned by the call is 60 minutes.

Note, too, that if a `SessionNotOnOrAfter` attribute is also defined, then the *lesser* value of the two attributes, `SessionDuration` or `SessionNotOnOrAfter`, establishes the maximum duration of the console session.

When you enable console sessions with an extended duration the risk of compromise of the credentials rises. To help you mitigate this risk, you can immediately disable the active console sessions for any role by choosing **Revoke Sessions** on the **Role Summary** page in the IAM console. For more information, see [Revoking IAM Role Temporary Security Credentials \(p. 277\)](#).

Important

The value of the `Name` attribute in the `Attribute` tag is case-sensitive. It must be set to `https://aws.amazon.com/SAML/Attributes/SessionDuration` exactly.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/SessionDuration">
  <AttributeValue>1800</AttributeValue>
</Attribute>
```

SAML PrincipalTag Attribute

(Optional) You can use an `Attribute` element with the `Name` attribute set to `https://aws.amazon.com/SAML/Attributes/PrincipalTag:{TagKey}`. This element allows you to pass

attributes as session tags in the SAML assertion. For more information about session tags, see [Passing Session Tags in AWS STS \(p. 299\)](#).

To pass attributes as session tags, include the `AttributeValue` element that specifies the value of the tag. For example, to pass the tag key-value pairs `Project = Marketing` and `CostCenter = 12345`, use the following attribute. Include a separate `Attribute` element for each tag.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:Project">
  <AttributeValue>Marketing</AttributeValue>
</Attribute>
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:CostCenter">
  <AttributeValue>12345</AttributeValue>
</Attribute>
```

To set the tags above as transitive, include another `Attribute` element with the `Name` attribute set to `https://aws.amazon.com/SAML/Attributes/TransitiveTagKeys`. This is an optional multivalued attribute that sets your session tags as transitive. Transitive tags persist when you use the SAML session to assume another role in AWS. This is known as [role chaining \(p. 177\)](#). For example, to set both the `Principal` and `CostCenter` tags as transitive, use the following attribute to specify the keys.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/TransitiveTagKeys">
  <AttributeValue>Project</AttributeValue>
  <AttributeValue>CostCenter</AttributeValue>
</Attribute>
```

Mapping SAML Attributes to AWS Trust Policy Context Keys

The tables in this section list commonly used SAML attributes and how they map to trust policy condition context keys in AWS. You can use these keys to control access to a role. To do that, compare the keys to the values that are included in the assertions that accompany a SAML access request.

Important

These keys are available only in IAM trust policies (policies that determine who can assume a role) and are not applicable to permissions policies.

In the `eduPerson` and `eduOrg` attributes table, values are typed either as strings or as lists of strings. For string values, you can test these values in IAM trust policies using `StringEquals` or `StringLike` conditions. For values that contain a list of strings, you can use the `ForAnyValue` and `ForAllValues` policy set operators (p. 621) to test the values in trust policies.

Note

You should include only one claim per AWS context key. If you include more than one, only one claim will be mapped.

eduPerson and eduOrg Attributes

eduPerson or eduOrg attribute (Name key)	Maps to this AWS Context Key (FriendlyName key)	Type
<code>urn:oid:1.3.6.1.4.1.5923.1.1.1.1</code>	<code>eduPersonAffiliation</code>	List of strings
<code>urn:oid:1.3.6.1.4.1.5923.1.1.1.2</code>	<code>eduPersonNickname</code>	List of strings
<code>urn:oid:1.3.6.1.4.1.5923.1.1.1.3</code>	<code>eduPersonOrgDN</code>	String
<code>urn:oid:1.3.6.1.4.1.5923.1.1.1.4</code>	<code>eduPersonOrgUnitDN</code>	List of strings
<code>urn:oid:1.3.6.1.4.1.5923.1.1.1.5</code>	<code>eduPersonPrimaryAffiliation</code>	String

eduPerson or eduOrg attribute (Name key)	Maps to this AWS Context Key (FriendlyName key)	Type
urn:oid:1.3.6.1.4.1.5923.1.1.1.6	eduPersonPrincipalName	String
urn:oid:1.3.6.1.4.1.5923.1.1.1.7	eduPersonEntitlement	List of strings
urn:oid:1.3.6.1.4.1.5923.1.1.1.8	eduPersonPrimaryOrgUnitDN	String
urn:oid:1.3.6.1.4.1.5923.1.1.1.9	eduPersonScopedAffiliation	List of strings
urn:oid:1.3.6.1.4.1.5923.1.1.1.10	eduPersonTargetedID	List of strings
urn:oid:1.3.6.1.4.1.5923.1.1.1.11	eduPersonAssurance	List of strings
urn:oid:1.3.6.1.4.1.5923.1.2.1.2	eduOrgHomePageURI	List of strings
urn:oid:1.3.6.1.4.1.5923.1.2.1.3	eduOrgIdentityAuthNPolicyURI	List of strings
urn:oid:1.3.6.1.4.1.5923.1.2.1.4	eduOrgLegalName	List of strings
urn:oid:1.3.6.1.4.1.5923.1.2.1.5	eduOrgSuperiorURI	List of strings
urn:oid:1.3.6.1.4.1.5923.1.2.1.6	eduOrgWhitePagesURI	List of strings
urn:oid:2.5.4.3	cn	List of strings

Active Directory Attributes

AD Attribute	Maps to this AWS Context Key	Type
http://schemas.xmlsoap.org/ws/2005/05/identity/claims/name	name	String
http://schemas.xmlsoap.org/claims/CommonName	commonName	String
http://schemas.xmlsoap.org/ws/2005/05/identity/claims/givenname	givenName	String
http://schemas.xmlsoap.org/ws/2005/05/identity/claims/surname	surname	String
http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress	mail	String
http://schemas.microsoft.com/ws/2008/06/identity/claims/primarygroupsid	uid	String

X.500 Attributes

X.500 Attribute	Maps to this AWS Context Key	Type
2.5.4.3	commonName	String
2.5.4.4	surname	String

X.500 Attribute	Maps to this AWS Context Key	Type
2.4.5.42	givenName	String
2.5.4.45	x500UniqueIdentifier	String
0.9.2342.19200300100.1.1	uid	String
0.9.2342.19200300100.1.3	mail	String
0.9.2342.19200300.100.1.45	organizationStatus	String

Enabling SAML 2.0 Federated Users to Access the AWS Management Console

You can use a role to configure your SAML 2.0-compliant identity provider (IdP) and AWS to permit your federated users to access the AWS Management Console. The role grants the user permissions to carry out tasks in the console. If you want to give SAML federated users other ways to access AWS, see one of these topics:

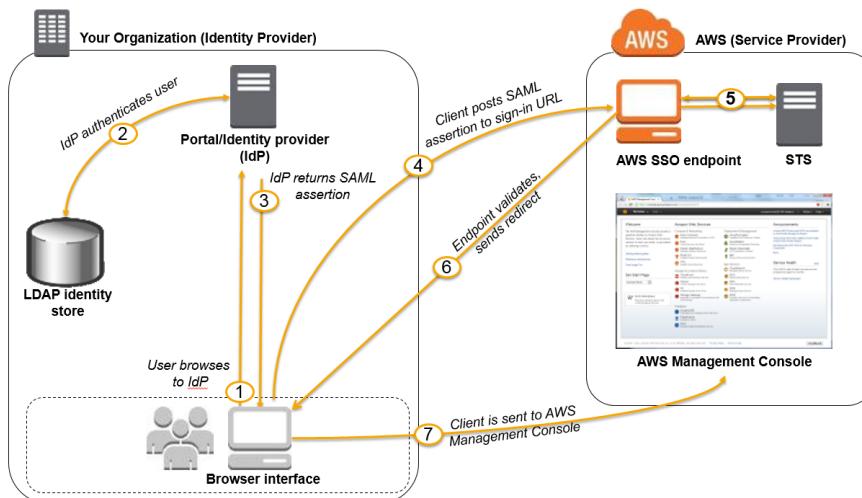
- AWS CLI: [Switching to an IAM Role \(AWS CLI\)](#) (p. 262)
- Tools for Windows PowerShell: [Switching to an IAM Role \(Tools for Windows PowerShell\)](#) (p. 266)
- AWS API: [Switching to an IAM Role \(AWS API\)](#) (p. 268)

Overview

The following diagram illustrates the flow for SAML-enabled single sign-on.

Note

This specific use of SAML differs from the more general one illustrated at [About SAML 2.0-based Federation \(p. 190\)](#) because this workflow opens the AWS Management Console on behalf of the user. This requires the use of the AWS SSO endpoint instead of directly calling the `AssumeRoleWithSAML` API. The endpoint calls the API for the user and returns a URL that automatically redirects the user's browser to the AWS Management Console.



The diagram illustrates the following steps:

1. The user browses to your organization's portal and selects the option to go to the AWS Management Console. In your organization, the portal is typically a function of your IdP that handles the exchange of trust between your organization and AWS. For example, in Active Directory Federation Services, the portal URL is: <https://ADFSServiceName/adfs/ls/IdpInitiatedSignOn.aspx>
2. The portal verifies the user's identity in your organization.
3. The portal generates a SAML authentication response that includes assertions that identify the user and include attributes about the user. You can also configure your IdP to include a SAML assertion attribute called `SessionDuration` that specifies how long the console session is valid. You can also configure the IdP to pass attributes as [session tags \(p. 299\)](#). The portal sends this response to the client browser.
4. The client browser is redirected to the AWS single sign-on endpoint and posts the SAML assertion.
5. The endpoint requests temporary security credentials on behalf of the user and creates a console sign-in URL that uses those credentials.
6. AWS sends the sign-in URL back to the client as a redirect.
7. The client browser is redirected to the AWS Management Console. If the SAML authentication response includes attributes that map to multiple IAM roles, the user is first prompted to select the role for accessing the console.

From the user's perspective, the process happens transparently: The user starts at your organization's internal portal and ends up at the AWS Management Console, without ever having to supply any AWS credentials.

Consult the following sections for an overview of how to configure this behavior along with links to detailed steps.

Configure your network as a SAML provider for AWS

Inside your organization's network, you configure your identity store (such as Windows Active Directory) to work with a SAML-based IdP like Windows Active Directory Federation Services, Shibboleth, etc. Using your IdP, you generate a metadata document that describes your organization as an IdP and includes authentication keys. You also configure your organization's portal to route user requests for the AWS Management Console to the AWS SAML endpoint for authentication using SAML assertions. How you configure your IdP to produce the `metadata.xml` file depends on your IdP. Refer to your IdP's documentation for instructions, or see [Integrating Third-Party SAML Solution Providers with AWS \(p. 204\)](#) for links to the web documentation for many of the SAML providers supported.

Create a SAML provider in IAM

Next, you sign in to the AWS Management Console and go to the IAM console. There you create a new SAML provider, which is an entity in IAM that holds information about your organization's IdP. As part of this process, you upload the metadata document produced by the IdP software in your organization in the previous section. For details, see [Creating IAM SAML Identity Providers \(p. 201\)](#).

Configure permissions in AWS for your federated users

The next step is to create an IAM role that establishes a trust relationship between IAM and your organization's IdP. This role must identify your IdP as a principal (trusted entity) for purposes of federation. The role also defines what users authenticated by your organization's IdP are allowed to do in AWS. You can use the IAM console to create this role. When you create the trust policy that indicates who can assume the role, you specify the SAML provider that you created earlier in IAM. You also specify one or more SAML attributes that a user must match to be allowed to assume the role. For example, you can specify that only users whose SAML `eduPersonOrgDN` value is `ExampleOrg` are allowed to sign in. The role wizard automatically adds a condition to test the `saml:aud` attribute to make sure that the role is assumed only for sign-in to the AWS Management Console. The trust policy for the role might look like this:

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {"Federated": "arn:aws:iam::ACCOUNT-ID-WITHOUT-HYPHENS:saml-provider/ExampleOrgSSOProvider"},
            "Action": "sts:AssumeRoleWithSAML",
            "Condition": {"StringEquals": {
                "saml:edupersonorgdn": "ExampleOrg",
                "saml:aud": "https://signin.aws.amazon.com/saml"
            }}
        }
    ]
}
```

For the [permission policy \(p. 356\)](#) in the role, you specify permissions as you would for any role, user, or group. For example, if users from your organization are allowed to administer Amazon EC2 instances, you explicitly allow Amazon EC2 actions in the permission policy. You can do this by assigning a [managed policy \(p. 457\)](#), such as the [Amazon EC2 Full Access](#) managed policy.

For details about creating a role for a SAML IdP, see [Creating a Role for SAML 2.0 Federation \(Console\) \(p. 248\)](#).

Finish Configuration and Create SAML Assertions

After you create the role, inform your SAML IdP about AWS as a service provider by installing the `saml-metadata.xml` file found at <https://signin.aws.amazon.com/static/saml-metadata.xml>. How you install that file depends on your IdP. Some providers give you the option to type the URL, whereupon the IdP gets and installs the file for you. Others require you to download the file from the URL and then provide it as a local file. Refer to your IdP documentation for details, or see [Integrating Third-Party SAML Solution Providers with AWS \(p. 204\)](#) for links to the web documentation for many of the supported SAML providers.

You also configure the information that you want the IdP to pass as SAML attributes to AWS as part of the authentication response. Most of this information appears in AWS as condition context keys that you can evaluate in your policies. These condition keys ensure that only authorized users in the right contexts are granted permissions to access your AWS resources. You can specify time windows that restrict when the console may be used. You can also specify the maximum time (up to 12 hours) that users can access the console before having to refresh their credentials. For details, see [Configuring SAML Assertions for the Authentication Response \(p. 206\)](#).

Enabling Custom Identity Broker Access to the AWS Console

You can write and run code to create a URL that lets users who sign in to your organization's network securely access the AWS Management Console. The URL includes a sign-in token that you get from AWS and that authenticates the user to AWS.

Note

If your organization uses an identity provider (IdP) that is compatible with SAML, you can set up access to the console without writing code. This works with providers like Microsoft's Active Directory Federation Services or open-source Shibboleth. For details, see [Enabling SAML 2.0 Federated Users to Access the AWS Management Console \(p. 211\)](#).

To enable your organization's users to access the AWS Management Console, you can create a custom *identity broker* that performs the following steps:

1. Verify that the user is authenticated by your local identity system.
2. Call the AWS Security Token Service (AWS STS) [AssumeRole](#) (recommended) or [GetFederationToken](#) API operations to obtain temporary security credentials for the user. To learn about the different methods that you can use to assume a role, see [Using IAM Roles \(p. 253\)](#). To learn how to pass optional session tags when you obtain your security credentials, see [Passing Session Tags in AWS STS \(p. 299\)](#).

- If you use one of the `AssumeRole*` API operations to get the temporary security credentials for a role, you can include the `DurationSeconds` parameter in your call. This parameter specifies the duration of your role session, from 900 seconds (15 minutes) up to the maximum session duration setting for the role. To learn how to view or change the maximum value for a role, see [View the Maximum Session Duration Setting for a Role \(p. 255\)](#). Additionally, if you use the `AssumeRole*` API operations, you must call them as an IAM user with long-term credentials. Otherwise, the call to the federation endpoint in step 3 fails.
 - If you use the `GetFederationToken` API operation to get the credentials, you can include the `DurationSeconds` parameter in your call. This parameter specifies the duration of your role session. The value can range from 900 seconds (15 minutes) to 129,600 seconds (36 hours). You can make this API call only by using the long-term AWS security credentials of an IAM user. You can also make these calls using AWS account root user credentials, but we do not recommend it. If you make this call as the root user, the default session lasts for one hour. Or you can specify a session from 900 seconds (15 minutes) up to 3,600 seconds (one hour).
3. Call the AWS federation endpoint and supply the temporary security credentials to request a sign-in token.
 4. Construct a URL for the console that includes the token:
 - If you use one of the `AssumeRole*` API operations in your URL, you can include the `SessionDuration` HTTP parameter. This parameter specifies the duration of the console session, from 900 seconds (15 minutes) to 43200 seconds (12 hours).
 - If you use the `GetFederationToken` API operation in your URL, you can include the `DurationSeconds` parameter. This parameter specifies the duration of the federated console session. The value can range from 900 seconds (15 minutes) to 129,600 seconds (36 hours).
 5. Give the URL to the user or invoke the URL on the user's behalf.

The URL that the federation endpoint provides is valid for 15 minutes after it is created. This differs from the duration (in seconds) of the temporary security credential session that is associated with the URL. Those credentials are valid for the duration you specified when you created them, starting from the time they were created.

Important

The URL grants access to your AWS resources through the AWS Management Console if you have enabled permissions in the associated temporary security credentials. For this reason, you should treat the URL as a secret. We recommend returning the URL through a secure redirect, for example, by using a 302 HTTP response status code over an SSL connection. For more information about the 302 HTTP response status code, go to [RFC 2616, section 10.3.3](#).

To view a sample application that shows you how you can implement a single sign-on solution, go to [AWS Management Console federation proxy sample use case](#) in the *AWS Sample Code & Libraries*.

To complete these tasks, you can use the [HTTPS Query API for AWS Identity and Access Management \(IAM\)](#) and the [AWS Security Token Service \(AWS STS\)](#). Or, you can use programming languages, such as Java, Ruby, or C#, along with the appropriate [AWS SDK](#). Each of these methods is described in the following sections.

Topics

- [Example Code Using IAM Query API Operations \(p. 215\)](#)
- [Example Code Using Python \(p. 217\)](#)
- [Example Code Using Java \(p. 218\)](#)
- [Example Showing How to Construct the URL \(Ruby\) \(p. 219\)](#)

Example Code Using IAM Query API Operations

You can construct a URL that gives your federated users direct access to the AWS Management Console. This task uses the IAM and AWS STS HTTPS Query API. For more information about making query requests, see [Making Query Requests](#).

Note

The following procedure contains examples of text strings. To enhance readability, line breaks have been added to some of the longer examples. When you create these strings for your own use, you should omit any line breaks.

To give a federated user access to your resources from the AWS Management Console

1. Authenticate the user in your identity and authorization system.
2. Obtain temporary security credentials for the user. The temporary credentials consist of an access key ID, a secret access key, and a security token. For more information about creating temporary credentials, see [Temporary Security Credentials \(p. 307\)](#).

To get temporary credentials, you call either the AWS STS [AssumeRole](#) API (recommended) or the [GetFederationToken](#) API. For more information about the differences between these API operations, see [Understanding the API Options for Securely Delegating Access to Your AWS Account](#) in the AWS Security Blog.

Important

When you use the [GetFederationToken](#) API to create temporary security credentials, you must specify the permissions that the credentials grant to the user who assumes the role. For any of the API operations that begin with `AssumeRole*`, you use an IAM role to assign permissions. For the other API operations, the mechanism varies with the API. For more details, see [Controlling Permissions for Temporary Security Credentials \(p. 322\)](#). Additionally, if you use the `AssumeRole*` API operations, you must call them as an IAM user with long-term credentials. Otherwise, the call to the federation endpoint in step 3 fails.

3. After you obtain the temporary security credentials, build them into a JSON session string to exchange them for a sign-in token. The following example shows how to encode the credentials. You replace the placeholder text with the appropriate values from the credentials that you receive in the previous step.

```
{"sessionId": "*** temporary access key ID ***",  
 "sessionKey": "*** temporary secret access key ***",  
 "sessionToken": "*** security token ***"}
```

4. [URL encode](#) the session string from the previous step. Because the information that you are encoding is sensitive, we recommend that you avoid using a web service for this encoding. Instead, use a locally installed function or feature in your development toolkit to securely encode this information. You can use the `urllib.quote_plus` function in Python, the `URLEncoder.encode` function in Java, or the `CGI.escape` function in Ruby. See the examples later in this topic.
5. Send your request to the AWS federation endpoint at the following address:

<https://signin.aws.amazon.com/federation>

The request must include the `Action` and `Session` parameters, and (optionally) if you used an `AssumeRole*` API operation, a `SessionDuration` HTTP parameter as shown in the following example.

```
Action = getSigninToken  
SessionDuration = time in seconds  
Session = *** the URL encoded JSON string created in steps 3 & 4 ***
```

The `SessionDuration` HTTP parameter specifies the duration of the console session. This is separate from the duration of the temporary credentials that you specify using the `DurationSeconds` parameter. You can specify a `SessionDuration` maximum value of 43,200 (12 hours). If the `SessionDuration` parameter is missing, then the session defaults to the duration of the credentials that you retrieved from AWS STS in step 2 (which defaults to one hour). See the [documentation for the `AssumeRole` API](#) for details about how to specify a duration using the `DurationSeconds` parameter. The ability to create a console session that is longer than one hour is intrinsic to the `getSigninToken` operation of the federation endpoint.

Note

Do not use the `SessionDuration` HTTP parameter if you got the temporary credentials with `GetFederationToken`. Doing so will cause the operation to fail.

When you enable console sessions with an extended duration, you increase the risk of credential exposure. To help you mitigate this risk, you can immediately disable the active console sessions for any role by choosing **Revoke Sessions** on the **Role Summary** IAM console page. For more information, see [Revoking IAM Role Temporary Security Credentials \(p. 277\)](#).

The following is an example of what your request might look like. The lines are wrapped here for readability, but you should submit it as a one-line string.

```
https://signin.aws.amazon.com/federation
?Action=getSigninToken
&SessionDuration=1800
&Session=%7B%22sessionId%22%3A+%22ASIAJUMHIZPTOKTBMK5A%22%2C+%22sessionKey%22
%3A+%22LSD7LWI%2FL%2FN%2BgYpan5QFz0XUpc8s7HYjRsgcsrsm%22%2C+%22sessionToken%22
%3A+%22FQoDYXdzEBQaDLbj3VWv2u50NN%2F3yyLSASwYtWhPnGPMNmzzFfzsL0Qd3vtYHw5A5dW
AjOsrankPkgHomIe3mJip5%2F0djDBbo7Sm0%2FENDEiCdpssQKodTpleKA8xQq0CwFg6a69xdEBQT8
FipATnLbKoyS4b%2FehbnstUjZZQWP0wXXqFF7gSm%2FMe2tXe0jzsdP0012obeZ9lijPSdF1k2b5
PfGhiuyAR9aD5%2BubM0pY86fKex1qsytyjvyTbZ9nXe6DvxVDcnCohOGETJ7XFksFdH0v%2FYR25C
UAhJ3nXIkB7G7Ucv9cOEpcf%2Fg23ijRgILIBQ%3D%3D%22%7D
```

The response from the federation endpoint is a JSON document with a `SigninToken` value. It will look similar to the following example.

```
{"SigninToken": "*** the SigninToken string ***"}
```

6. Finally, create the URL that your federated users can use to access the AWS Management Console. The URL is the same federation URL endpoint that you used in [Step 5 \(p. 215\)](#), plus the following parameters:

```
?Action = login
&Issuer = *** the form-urlencoded URL for your internal sign-in page ***
&Destination = *** the form-urlencoded URL to the desired AWS console page ***
&SigninToken = *** the value of SigninToken received in the previous step ***
```

The following example shows what the final URL might look like. The URL is valid for 15 minutes from the time it is created. The temporary security credentials and console session embedded within the URL are valid for the duration you specify in the `SessionDuration` HTTP parameter when you initially request them.

```
https://signin.aws.amazon.com/federation
?Action=login
&Issuer=https%3A%2F%2Fexample.com
&Destination=https%3A%2F%2Fconsole.aws.amazon.com%2Fs
&SigninToken=VCQgs5qZt3Q6fn8Tr5EXAMPLELnwB7JjUc-SHwnUUWabcRdnWsi4DBn-dvC
CZ85wrD0nmlldUcZEXAMPLE-vXYH4Q__mleuF_W2BE5HYexbe9y4Of-kje53SsjNNecATfjIzpW1
WibbnH6YcYRiBoffZBGExbEXAMPLE5aiKX4THWjQKC6gg6alHu6JFrnOJoK3dtP6I9a6hi6yPgm
```

```
iOkPZMmNGmhsVxetKzr8mx3pxhHbMEXAMPLEv1pij0rok3IyCR2YVcIjqwfWv32HU2Xl{j471u
3fU6uOfUComeKiqtGX974xzJ0ZbdmX_t_1LrhEXAMPLEDDIissSnyHGw2xaZZqudm4mo2uTDk9Pv
915K0ZCqIgEXAMPLEcA6tgLPykEWGUyH6BdSC6166n4M4JkXIQgac7_7821YqixsNxZ6rsrpzwf
nQoS1407R0eJCCJ684EXAMPLERdBNuLbUYpz2Iw3vIN0tQgOujwnwydPscM9F7foaEK3jwMkg
Apeb1-6L_OB12MZhuFxx5555EXAMPLEhyETEd4ZulKPdXHkg16T9ZkI1Hz2Uy1RUTUhUxNtSQ
nWc5xbBoEcXqposIEK7yhje9Vzhd61AEXAMPLElbWeouACEMG6-Vd3dAgFYd6i5FYoyFrZLWvm
0LSG7RyYKeYN5VIZUk3YWQpyjPORi5KUrsUi-NEXAMPLExMOMdoODBEgKQsk-iu2ozh6r8bxwC
RNhuwjg
```

Example Code Using Python

The following example shows how to use Python to programmatically construct a URL that gives federated users direct access to the AWS Management Console. The example uses the [AWS SDK for Python \(Boto 3\)](#).

The code uses the [AssumeRole](#) API to obtain temporary security credentials.

```
import urllib, json, sys
import requests # 'pip install requests'
import boto3 # AWS SDK for Python (Boto3) 'pip install boto3'

# Step 1: Authenticate user in your own identity system.

# Step 2: Using the access keys for an IAM user in your AWS account,
# call "AssumeRole" to get temporary access keys for the federated user

# Note: Calls to AWS STS AssumeRole must be signed using the access key ID
# and secret access key of an IAM user or using existing temporary credentials.
# The credentials can be in EC2 instance metadata, in environment variables,
# or in a configuration file, and will be discovered automatically by the
# client('sts') function. For more information, see the Python SDK docs:
# http://boto3.readthedocs.io/en/latest/reference/services/sts.html
# http://boto3.readthedocs.io/en/latest/reference/services/sts.html#STS.Client.assume_role
sts_connection = boto3.client('sts')

assumed_role_object = sts_connection.assume_role(
    RoleArn="arn:aws:iam::ACCOUNT-ID-WITHOUT-HYPHENS:role/ROLE-NAME",
    RoleSessionName="AssumeRoleSession",
)

# Step 3: Format resulting temporary credentials into JSON
url_credentials = {}
url_credentials['sessionId'] = assumed_role_object.get('Credentials').get('AccessKeyId')
url_credentials['sessionKey'] =
    assumed_role_object.get('Credentials').get('SecretAccessKey')
url_credentials['sessionToken'] =
    assumed_role_object.get('Credentials').get('SessionToken')
json_string_with_temp_credentials = json.dumps(url_credentials)

# Step 4. Make request to AWS federation endpoint to get sign-in token. Construct the
# parameter string with
# the sign-in action request, a 12-hour session duration, and the JSON document with
# temporary credentials
# as parameters.
request_parameters = "?Action=getSigninToken"
request_parameters += "&SessionDuration=43200"
if sys.version_info[0] < 3:
    def quote_plus_function(s):
        return urllib.quote_plus(s)
else:
    def quote_plus_function(s):
        return urllib.parse.quote_plus(s)
request_parameters += "&Session=" + quote_plus_function(json_string_with_temp_credentials)
```

```

request_url = "https://signin.aws.amazon.com/federation" + request_parameters
r = requests.get(request_url)
# Returns a JSON document with a single element named SigninToken.
signin_token = json.loads(r.text)

# Step 5: Create URL where users can use the sign-in token to sign in to
# the console. This URL must be used within 15 minutes after the
# sign-in token was issued.
request_parameters = "?Action=login"
request_parameters += "&Issuer=Example.org"
request_parameters += "&Destination=" + quote_plus_function("https://"
console.aws.amazon.com/")
request_parameters += "&SigninToken=" + signin_token["SigninToken"]
request_url = "https://signin.aws.amazon.com/federation" + request_parameters

# Send final URL to stdout
print (request_url)

```

Example Code Using Java

The following example shows how to use Java to programmatically construct a URL that gives federated users direct access to the AWS Management Console. The following code snippet uses the [AWS SDK for Java](#).

```

import java.net.URLDecoder;
import java.net.URL;
import java.netURLConnection;
import java.io.BufferedReader;
import java.io.InputStreamReader;
// Available at http://www.json.org/java/index.html
import org.json.JSONObject;
import com.amazonaws.auth.AWS Credentials;
import com.amazonaws.auth.BasicAWS Credentials;
import com.amazonaws.services.securitytoken.AWSSecurityTokenServiceClient;
import com.amazonaws.services.securitytoken.model.Credentials;
import com.amazonaws.services.securitytoken.model.GetFederationTokenRequest;
import com.amazonaws.services.securitytoken.model.GetFederationTokenResult;

/* Calls to AWS STS API operations must be signed using the access key ID
   and secret access key of an IAM user or using existing temporary
   credentials. The credentials should not be embedded in code. For
   this example, the code looks for the credentials in a
   standard configuration file.
*/
AWS Credentials credentials =
    new PropertiesCredentials(
        AwsConsoleApp.class.getResourceAsStream("AwsCredentials.properties"));

AWSSecurityTokenServiceClient stsClient =
    new AWSSecurityTokenServiceClient(credentials);

GetFederationTokenRequest getFederationTokenRequest =
    new GetFederationTokenRequest();
getFederationTokenRequest.setDurationSeconds(1800);
getFederationTokenRequest.setName("UserName");

// A sample policy for accessing Amazon Simple Notification Service (Amazon SNS) in the
// console.

String policy = "{\"Version\":\"2012-10-17\", \"Statement\":[{\"Action\":\"sns:*\", \""
    +"Effect\":\"Allow\", \"Resource\":\"*\"]}}";
getFederationTokenRequest.setPolicy(policy);

```

```

GetFederationTokenResult federationTokenResult =
    stsClient.getFederationToken(getFederationTokenRequest);

Credentials federatedCredentials = federationTokenResult.getCredentials();

// The issuer parameter specifies your internal sign-in
// page, for example https://mysignin.internal.mycompany.com/.
// The console parameter specifies the URL to the destination console of the
// AWS Management Console. This example goes to Amazon SNS.
// The signin parameter is the URL to send the request to.

String issuerURL = "https://mysignin.internal.mycompany.com/";
String consoleURL = "https://console.aws.amazon.com/sns";
String signInURL = "https://signin.aws.amazon.com/federation";

// Create the sign-in token using temporary credentials,
// including the access key ID, secret access key, and security token.
String sessionJson = String.format(
    "{\"%1$s\":\"%2$s\", \"%3$s\":\"%4$s\", \"%5$s\":\"%6$s\"}",
    "sessionId", federatedCredentials.getAccessKeyId(),
    "sessionKey", federatedCredentials.getSecretAccessKey(),
    "sessionToken", federatedCredentials.getSessionToken());

// Construct the sign-in request with the request sign-in token action, a
// 12-hour console session duration, and the JSON document with temporary
// credentials as parameters.

String getSigninTokenURL = signInURL +
    "?Action=getSigninToken" +
    "&DurationSeconds=43200" +
    "&SessionType=json&Session=" +
    URLEncoder.encode(sessionJson, "UTF-8");

URL url = new URL(getSigninTokenURL);

// Send the request to the AWS federation endpoint to get the sign-in token
URLConnection conn = url.openConnection ();

BufferedReader bufferReader = new BufferedReader(new
    InputStreamReader(conn.getInputStream()));
String returnContent = bufferReader.readLine();

String signinToken = new JSONObject(returnContent).getString("SigninToken");

String signinTokenParameter = "&SigninToken=" + URLEncoder.encode(signinToken, "UTF-8");

// The issuer parameter is optional, but recommended. Use it to direct users
// to your sign-in page when their session expires.

String issuerParameter = "&Issuer=" + URLEncoder.encode(issuerURL, "UTF-8");

// Finally, present the completed URL for the AWS console session to the user

String destinationParameter = "&Destination=" + URLEncoder.encode(consoleURL, "UTF-8");
String loginURL = signInURL + "?Action=login" +
    signinTokenParameter + issuerParameter + destinationParameter;

```

Example Showing How to Construct the URL (Ruby)

The following example shows how to use Ruby to programmatically construct a URL that gives federated users direct access to the AWS Management Console. This code snippet uses the [AWS SDK for Ruby](#).

```
require 'rubygems'
```

```

require 'json'
require 'open-uri'
require 'cgi'
require 'aws-sdk'

# Create a new STS instance
#
# Note: Calls to AWS STS API operations must be signed using an access key ID
# and secret access key. The credentials can be in EC2 instance metadata
# or in environment variables and will be automatically discovered by
# the default credentials provider in the AWS Ruby SDK.
sts = Aws::STS::Client.new()

# The following call creates a temporary session that returns
# temporary security credentials and a session token.
# The policy grants permissions to work
# in the AWS SNS console.

session = sts.get_federation_token({
  duration_seconds: 1800,
  name: "UserName",
  policy: "{\"Version\":\"2012-10-17\", \"Statement\":{\"Effect\":\"Allow\", \"Action\":\"sns:*\", \"Resource\":\"*\"}}",
})

# The issuer value is the URL where users are directed (such as
# to your internal sign-in page) when their session expires.
#
# The console value specifies the URL to the destination console.
# This example goes to the Amazon SNS console.
#
# The sign-in value is the URL of the AWS STS federation endpoint.
issuer_url = "https://mysignin.internal.mycompany.com/"
console_url = "https://console.aws.amazon.com/sns"
signin_url = "https://signin.aws.amazon.com/federation"

# Create a block of JSON that contains the temporary credentials
# (including the access key ID, secret access key, and session token).
session_json = {
  :sessionId => session.credentials[:access_key_id],
  :sessionKey => session.credentials[:secret_access_key],
  :sessionToken => session.credentials[:session_token]
}.to_json

# Call the federation endpoint, passing the parameters
# created earlier and the session information as a JSON block.
# The request returns a sign-in token that's valid for 15 minutes.
# Signing in to the console with the token creates a session
# that is valid for 12 hours.
get_signin_token_url = signin_url +
  "?Action=getSigninToken" +
  "&SessionType=json&Session=" +
  CGI.escape(session_json)

returned_content = URI.parse(get_signin_token_url).read

# Extract the sign-in token from the information returned
# by the federation endpoint.
signin_token = JSON.parse(returned_content)['SigninToken']
signin_token_param = "&SigninToken=" + CGI.escape(signin_token)

# Create the URL to give to the user, which includes the
# sign-in token and the URL of the console to open.
# The "issuer" parameter is optional but recommended.
issuer_param = "&Issuer=" + CGI.escape(issuer_url)
destination_param = "&Destination=" + CGI.escape(console_url)

```

```
login_url = signin_url + "?Action=login" + signin_token_param +
issuer_param + destination_param
```

Using Service-Linked Roles

A service-linked role is a unique type of IAM role that is linked directly to an AWS service. Service-linked roles are predefined by the service and include all the permissions that the service requires to call other AWS services on your behalf. The linked service also defines how you create, modify, and delete a service-linked role. A service might automatically create or delete the role. It might allow you to create, modify, or delete the role as part of a wizard or process in the service. Or it might require that you use IAM to create or delete the role. Regardless of the method, service-linked roles make setting up a service easier because you don't have to manually add the necessary permissions for the service to complete actions on your behalf.

The linked service defines the permissions of its service-linked roles, and unless defined otherwise, only that service can assume the roles. The defined permissions include the trust policy and the permissions policy, and that permissions policy cannot be attached to any other IAM entity.

You can delete the roles only after first deleting their related resources. This protects your resources because you can't inadvertently remove permission to access the resources.

Tip

For information about which services support using service-linked roles, see [AWS Services That Work with IAM \(p. 586\)](#) and look for the services that have **Yes** in the **Service-Linked Role** column. Choose a **Yes** with a link to view the service-linked role documentation for that service.

Service-Linked Role Permissions

You must configure permissions for an IAM entity (user, group, or role) to allow the user or role to create or edit the service-linked role.

Note

The ARN for a service-linked role includes a service principal, which is indicated in the policies below as `SERVICE-NAME.amazonaws.com`. Do not try to guess the service principal, because it is case sensitive and the format can vary across AWS services. To view the service principal for a service, see its service-linked role documentation.

To allow an IAM entity to create a specific service-linked role

Add the following policy to the IAM entity that needs to create the service-linked role.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::*:role/aws-service-role/SERVICE-
NAME.amazonaws.com/SERVICE-LINKED-ROLE-NAME-PREFIX*",
            "Condition": {"StringLike": {"iam:AWSServiceName": "SERVICE-
NAME.amazonaws.com"}}
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:AttachRolePolicy",
                "iam:PutRolePolicy"
            ],
            "Resource": "arn:aws:iam::*:role/aws-service-role/SERVICE-
NAME.amazonaws.com/SERVICE-LINKED-ROLE-NAME-PREFIX*"
        }
    ]
}
```

```
    ]  
}
```

To allow an IAM entity to create any service-linked role

Add the following statement to the permissions policy for the IAM entity that needs to create a service-linked role, or any service role that includes the needed policies. This policy statement does not allow the IAM entity to attach a policy to the role.

```
{  
    "Effect": "Allow",  
    "Action": "iam:CreateServiceLinkedRole",  
    "Resource": "arn:aws:iam::*:role/aws-service-role/*"  
}
```

To allow an IAM entity to edit the description of any service roles

Add the following statement to the permissions policy for the IAM entity that needs to edit the description of a service-linked role, or any service role.

```
{  
    "Effect": "Allow",  
    "Action": "iam:UpdateRoleDescription",  
    "Resource": "arn:aws:iam::*:role/aws-service-role/*"  
}
```

To allow an IAM entity to delete a specific service-linked role

Add the following statement to the permissions policy for the IAM entity that needs to delete the service-linked role.

```
{  
    "Effect": "Allow",  
    "Action": [  
        "iam>DeleteServiceLinkedRole",  
        "iam:GetServiceLinkedRoleDeletionStatus"  
    ],  
    "Resource": "arn:aws:iam::*:role/aws-service-role/SERVICE-NAME.amazonaws.com/SERVICE-LINKED-ROLE-NAME-PREFIX*"  
}
```

To allow an IAM entity to delete any service-linked role

Add the following statement to the permissions policy for the IAM entity that needs to delete a service-linked role, but not service role.

```
{  
    "Effect": "Allow",  
    "Action": [  
        "iam>DeleteServiceLinkedRole",  
        "iam:GetServiceLinkedRoleDeletionStatus"  
    ],  
    "Resource": "arn:aws:iam::*:role/aws-service-role/*"  
}
```

To allow an IAM entity to pass an existing role to the service

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

statement to the permissions policy for the IAM entity that needs to pass a role. This policy statement also allows the entity to view a list of roles from which they can choose the role to pass. For more information, see [Granting a User Permissions to Pass a Role to an AWS Service \(p. 258\)](#).

```
{  
    "Effect": "Allow",  
    "Action": [  
        "iam:ListRoles",  
        "iam:PassRole"  
    ],  
    "Resource": "arn:aws:iam::123456789012:role/my-role-for-XYZ"  
}
```

Transferring Service-Linked Role Permissions

The permissions granted by a service-linked role are indirectly transferable to other users and roles. When you allow a service to perform operations in other services, the service can use those permissions in the future. If another user or role has permission to perform actions in the service, the service can then assume the role and access resources in other services. This means that the other user or role can indirectly access the other services.

For example, when you create an Amazon RDS DB instance, [RDS creates the service-linked role](#) for you. This role allows RDS to call Amazon EC2, Amazon SNS, Amazon CloudWatch Logs, and Amazon Kinesis on your behalf whenever you edit the DB instance. If you create a policy to allow users and roles in your account or another account to access that Amazon RDS instance, then RDS can still use that role make changes to EC2, SNS, CloudWatch Logs, and Kinesis on their behalf. The new user or role can indirectly edit resources in those other services.

Creating a Service-Linked Role

The method that you use to create a service-linked role depends on the service. In some cases, you don't need to manually create a service-linked role. For example, when you complete a specific action (such as creating a resource) in the service, the service might create the service-linked role for you. Or if you were using a service before it began supporting service-linked roles, then the service might have automatically created the role in your account. To learn more, see [A New Role Appeared in My AWS Account \(p. 564\)](#).

In other cases, the service might support creating a service-linked role manually using the service console, API, or CLI. For information about which services support using service-linked roles, see [AWS Services That Work with IAM \(p. 586\)](#) and look for the services that have Yes in the **Service-Linked Role** column. To learn whether the service supports creating the service-linked role, choose the Yes link to view the service-linked role documentation for that service.

If the service does not support creating the role, then you can use IAM to create the service-linked role.

Important

Service-linked roles count toward your [IAM roles in an AWS account](#) limit, but if you have reached your limit, you can still create service-linked roles in your account. Only service-linked roles can exceed the limit.

Creating a Service-Linked Role (Console)

Before you create a service-linked role in IAM, find out whether the linked service automatically creates service-linked roles. In addition, learn whether you can create the role from the service's console, API, or CLI.

To create a service-linked role (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.

2. In the navigation pane of the IAM console, choose **Roles**. Then choose **Create role**.
3. Choose the **AWS Service** role type, and then choose the service that you want to allow to assume this role.
4. Choose the use case for your service. If the specified service has only one use case, it is selected for you. Use cases are defined by the service to include the trust policy required by the service. Then choose **Next: Permissions**.
5. Choose one or more permissions policies to attach to the role. Depending on the use case that you selected, the service might do any of the following:
 - Define the permissions used by the role
 - Allow you to choose from a limited set of permissions
 - Allow you to choose from any permissions
 - Allow you to select no policies at this time, create the policies later, and then attach them to the role.

Select the box next to the policy that assigns the permissions that you want the role to have, and then choose **Next: Tags**.

Note

The permissions that you specify are available to any entity that uses the role. By default, a role has no permissions.

6. Choose **Next: Review**. You cannot attach tags to service-linked roles during creation. For more information about using tags in IAM, see [Tagging IAM Users and Roles \(p. 295\)](#).
7. For **Role name**, the degree of role name customization is defined by the service. If the service defines the role's name, then this option is not editable. In other cases, the service might define a prefix for the role and allow you to type an optional suffix.

If possible, type a role name suffix to add to the default name. This suffix helps you identify the purpose of this role. Role names must be unique within your AWS account. They are not distinguished by case. For example, you cannot create roles named both `<service-linked-role-name>_SAMPLE` and `<service-linked-role-name>_sample`. Because various entities might reference the role, you cannot edit the name of the role after it has been created.

8. (Optional) For **Role description**, edit the description for the new service-linked role.
9. Review the role and then choose **Create role**.

Creating a Service-Linked Role (AWS CLI)

Before creating a service-linked role in IAM, find out whether the linked service automatically creates service-linked roles and whether you can create the role from the service's CLI. If the service CLI is not supported, you can use IAM commands to create a service-linked role with the trust policy and inline policies that the service needs to assume the role.

To create a service-linked role (AWS CLI)

Run the following command:

```
$ aws iam create-service-linked-role --aws-service-name SERVICE-NAME.amazonaws.com
```

Creating a Service-Linked Role (AWS API)

Before creating a service-linked role in IAM, find out whether the linked service automatically creates service-linked roles and whether you can create the role from the service's API. If the service API is not supported, you can use the AWS API to create a service-linked role with the trust policy and inline policies that the service needs to assume the role.

To create a service-linked role (AWS API)

Use the [CreateServiceLinkedRole](#) API call. In the request, specify a service name of *SERVICE_NAME_URL*.amazonaws.com.

For example, to create the **Lex Bots** service-linked role, use lex.amazonaws.com.

Editing a Service-Linked Role

The method that you use to edit a service-linked role depends on the service. Some services might allow you to edit the permissions for a service-linked role from the service console, API, or CLI. However, after you create a service-linked role, you cannot change the name of the role because various entities might reference the role. You can edit the description of any role from the IAM console, API, or CLI.

For information about which services support using service-linked roles, see [AWS Services That Work with IAM \(p. 586\)](#) and look for the services that have **Yes** in the **Service-Linked Role** column. To learn whether the service supports editing the service-linked role, choose the **Yes** link to view the service-linked role documentation for that service.

Editing a Service-Linked Role Description (Console)

You can use the IAM console to edit the description of a service-linked role.

To edit the description of a service-linked role (console)

1. In the navigation pane of the IAM console, choose **Roles**.
2. Choose the name of the role to modify.
3. To the far right of **Role description**, choose **Edit**.
4. Type a new description in the box and choose **Save**.

Editing a Service-Linked Role Description (AWS CLI)

You can use IAM commands from the AWS CLI to edit the description of a service-linked role.

To change the description of a service-linked role (AWS CLI)

1. (Optional) To view the current description for a role, run the following command:

```
$ aws iam get-role --role-name ROLE-NAME
```

Use the role name, not the ARN, to refer to roles with the CLI commands. For example, if a role has the following ARN: arn:aws:iam::123456789012:role/myrole, you refer to the role as **myrole**.

2. To update a service-linked role's description, run the following command:

```
$ aws iam update-role --role-name ROLE-NAME --description OPTIONAL-DESCRIPTION
```

Editing a Service-Linked Role Description (AWS API)

You can use the AWS API to edit the description of a service-linked role.

To change the description of a service-linked role (AWS API)

1. (Optional) To view the current description for a role, call the following operation, and specify the name of the role:

AWS API: [GetRole](#)

2. To update a role's description, call the following operation, and specify the name (and optional description) of the role:

AWS API: [UpdateRole](#)

Deleting a Service-Linked Role

The method that you use to create a service-linked role depends on the service. In some cases, you don't need to manually delete a service-linked role. For example, when you complete a specific action (such as removing a resource) in the service, the service might delete the service-linked role for you.

In other cases, the service might support deleting a service-linked role manually from the service console, API, or CLI.

For information about which services support using service-linked roles, see [AWS Services That Work with IAM \(p. 586\)](#) and look for the services that have **Yes** in the **Service-Linked Role** column. To learn whether the service supports deleting the service-linked role, choose the **Yes** link to view the service-linked role documentation for that service.

If the service does not support deleting the role, then you can delete the service-linked role from the IAM console, API, or CLI. If you no longer need to use a feature or service that requires a service-linked role, we recommend that you delete that role. That way you don't have an unused entity that is not actively monitored or maintained. However, you must clean up your service-linked role before you can delete it.

Cleaning Up a Service-Linked Role

Before you can use IAM to delete a service-linked role, you must first confirm that the role has no active sessions and remove any resources used by the role.

To check whether the service-linked role has an active session in the IAM console

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane of the IAM console, choose **Roles**. Then choose the name (not the check box) of the service-linked role.
3. On the **Summary** page for the selected role, choose the **Access Advisor** tab.
4. On the **Access Advisor** tab, review recent activity for the service-linked role.

Note

If you are unsure whether the service is using the service-linked role, you can try to delete the role. If the service is using the role, then the deletion fails and you can view the regions where the role is being used. If the role is being used, then you must wait for the session to end before you can delete the role. You cannot revoke the session for a service-linked role.

To remove resources used by a service-linked role

For information about which services support using service-linked roles, see [AWS Services That Work with IAM \(p. 586\)](#) and look for the services that have **Yes** in the **Service-Linked Role** column. To learn whether the service supports deleting the service-linked role, choose the **Yes** link to view the service-linked role documentation for that service. See the documentation for that service to learn how to remove resources used by your service-linked role.

Deleting a Service-Linked Role (Console)

You can use the IAM console to delete a service-linked role.

To delete a service-linked role (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane of the IAM console, choose **Roles**. Then select the check box next to the role name that you want to delete, not the name or row itself.
3. For **Role actions** at the top of the page, choose **Delete role**.
4. In the confirmation dialog box, review the service last accessed data, which shows when each of the selected roles last accessed an AWS service. This helps you to confirm whether the role is currently active. If you want to proceed, choose **Yes, Delete** to submit the service-linked role for deletion.
5. Watch the IAM console notifications to monitor the progress of the service-linked role deletion. Because the IAM service-linked role deletion is asynchronous, after you submit the role for deletion, the deletion task can succeed or fail.
 - If the task succeeds, then the role is removed from the list and a notification of success appears at the top of the page.
 - If the task fails, you can choose **View details** or **View Resources** from the notifications to learn why the deletion failed. If the deletion fails because the role is using the service's resources, then the notification includes a list of resources, if the service returns that information. You can then [clean up the resources \(p. 226\)](#) and submit the deletion again.

Note

You might have to repeat this process several times, depending on the information that the service returns. For example, your service-linked role might use six resources and your service might return information about five of them. If you clean up the five resources and submit the role for deletion again, the deletion fails and the service reports the one remaining resource. A service might return all of the resources, a few of them, or it might not report any resources.

- If the task fails and the notification does not include a list of resources, then the service might not return that information. To learn how to clean up the resources for that service, see [AWS Services That Work with IAM \(p. 586\)](#). Find your service in the table, and choose the **Yes** link to view the service-linked role documentation for that service.

Deleting a Service-Linked Role (AWS CLI)

You can use IAM commands from the AWS CLI to delete a service-linked role.

To delete a service-linked role (AWS CLI)

1. If you don't know the name of the service-linked role that you want to delete, type the following command to list the roles and their Amazon Resource Names (ARNs) in your account:

```
$ aws iam get-role --role-name role-name
```

Use the role name, not the ARN, to refer to roles with the CLI commands. For example, if a role has the following ARN: `arn:aws:iam::123456789012:role/myrole`, you refer to the role as **myrole**.

2. Because a service-linked role cannot be deleted if it is being used or has associated resources, you must submit a deletion request. That request can be denied if these conditions are not met. You must capture the `deletion-task-id` from the response to check the status of the deletion task. Type the following command to submit a service-linked role deletion request:

```
$ aws iam delete-service-linked-role --role-name role-name
```

3. Type the following command to check the status of the deletion task:

```
$ aws iam get-service-linked-role-deletion-status --deletion-task-id deletion-task-id
```

The status of the deletion task can be NOT_STARTED, IN_PROGRESS, SUCCEEDED, or FAILED. If the deletion fails, the call returns the reason that it failed so that you can troubleshoot. If the deletion fails because the role is using the service's resources, then the notification includes a list of resources, if the service returns that information. You can then [clean up the resources \(p. 226\)](#) and submit the deletion again.

Note

You might have to repeat this process several times, depending on the information that the service returns. For example, your service-linked role might use six resources and your service might return information about five of them. If you clean up the five resources and submit the role for deletion again, the deletion fails and the service reports the one remaining resource. A service might return all of the resources, a few of them, or it might not report any resources. To learn how to clean up the resources for a service that does not report any resources, see [AWS Services That Work with IAM \(p. 586\)](#). Find your service in the table, and choose the **Yes** link to view the service-linked role documentation for that service.

Deleting a Service-Linked Role (AWS API)

You can use the AWS API to delete a service-linked role.

To delete a service-linked role (AWS API)

1. To submit a deletion request for a service-linked role, call [DeleteServiceLinkedRole](#). In the request, specify a role name.

Because a service-linked role cannot be deleted if it is being used or has associated resources, you must submit a deletion request. That request can be denied if these conditions are not met. You must capture the `DeletionTaskId` from the response to check the status of the deletion task.

2. To check the status of the deletion, call [GetServiceLinkedRoleDeletionStatus](#). In the request, specify the `DeletionTaskId`.

The status of the deletion task can be NOT_STARTED, IN_PROGRESS, SUCCEEDED, or FAILED. If the deletion fails, the call returns the reason that it failed so that you can troubleshoot. If the deletion fails because the role is using the service's resources, then the notification includes a list of resources, if the service returns that information. You can then [clean up the resources \(p. 226\)](#) and submit the deletion again.

Note

You might have to repeat this process several times, depending on the information that the service returns. For example, your service-linked role might use six resources and your service might return information about five of them. If you clean up the five resources and submit the role for deletion again, the deletion fails and the service reports the one remaining resource. A service might return all of the resources, a few of them, or it might not report any resources. To learn how to clean up the resources for a service that does not report any resources, see [AWS Services That Work with IAM \(p. 586\)](#). Find your service in the table, and choose the **Yes** link to view the service-linked role documentation for that service.

Creating IAM Roles

To create a role, you can use the AWS Management Console, the AWS CLI, the Tools for Windows PowerShell, or the IAM API.

If you use the AWS Management Console, a wizard guides you through the steps for creating a role. The wizard has slightly different steps depending on whether you're creating a role for an AWS service, for an AWS account, or for a federated user.

Topics

- [Creating a Role to Delegate Permissions to an IAM User \(p. 229\)](#)
- [Creating a Role to Delegate Permissions to an AWS Service \(p. 236\)](#)
- [Creating a Role for a Third-Party Identity Provider \(Federation\) \(p. 242\)](#)
- [Examples of Policies for Delegating Access \(p. 250\)](#)

Creating a Role to Delegate Permissions to an IAM User

You can use IAM roles to delegate access to your AWS resources. With IAM roles, you can establish trust relationships between your *trusting* account and other AWS *trusted* accounts. The trusting account owns the resource to be accessed and the trusted account contains the users who need access to the resource. However, it is possible for another account to own a resource in your account. For example, the trusting account might allow the trusted account to create new resources, such as creating new objects in an Amazon S3 bucket. In that case, the account that creates the resource owns the resource and controls who can access that resource.

After you create the trust relationship, an IAM user or an application from the trusted account can use the AWS Security Token Service (AWS STS) [AssumeRole](#) API operation. This operation provides temporary security credentials that enable access to AWS resources in your account.

The accounts can both be controlled by you, or the account with the users can be controlled by a third party. If the other account with the users is in an AWS account that you do not control, then you can use the `externalId` attribute. The external ID can be any word or number that is agreed upon between you and the administrator of the third-party account. This option automatically adds a condition to the trust policy that allows the user to assume the role only if the request includes the correct `sts:ExternalID`. For more information, see [How to Use an External ID When Granting Access to Your AWS Resources to a Third Party \(p. 233\)](#).

For information about how to use roles to delegate permissions, see [Roles Terms and Concepts \(p. 176\)](#). For information about using a service role to allow services to access resources in your account, see [Creating a Role to Delegate Permissions to an AWS Service \(p. 236\)](#).

Creating an IAM Role (Console)

You can use the AWS Management Console to create a role that an IAM user can assume. For example, assume that your organization has multiple AWS accounts to isolate a development environment from a production environment. For a high-level description of the steps to set up and use a role that allows users in the development account to access resources in the production account, see [Example Scenario Using Separate Development and Production Accounts \(p. 180\)](#).

To create a role (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane of the console, choose **Roles** and then choose **Create role**.
3. Choose the **Another AWS account** role type.
4. For **Account ID**, type the AWS account ID to which you want to grant access to your resources.

The administrator of the specified account can grant permission to assume this role to any IAM user in that account. To do this, the administrator attaches a policy to the user or a group that

grants permission for the `sts:AssumeRole` action. That policy must specify the role's ARN as the Resource.

5. If you are granting permissions to users from an account that you do not control, and the users will assume this role programmatically, then select **Require external ID**. The external ID can be any word or number that is agreed upon between you and the administrator of the third-party account. This option automatically adds a condition to the trust policy that allows the user to assume the role only if the request includes the correct `sts:ExternalID`. For more information, see [How to Use an External ID When Granting Access to Your AWS Resources to a Third Party \(p. 233\)](#).

Important

Choosing this option restricts access to the role only through the AWS CLI, Tools for Windows PowerShell, or the AWS API. This is because you cannot use the AWS console to switch to a role that has an `externalId` condition in its trust policy. However, you can create this kind of access programmatically by writing a script or an application using the relevant SDK. For more information and a sample script, see [How to Enable Cross-Account Access to the AWS Management Console](#) in the [AWS Security Blog](#).

6. If you want to restrict the role to users who sign in with multi-factor authentication (MFA), select **Require MFA**. This adds a condition to the role's trust policy that checks for an MFA sign-in. A user who wants to assume the role must sign in with a temporary one-time password from a configured MFA device. Users without MFA authentication cannot assume the role. For more information about MFA, see [Using Multi-Factor Authentication \(MFA\) in AWS \(p. 119\)](#)
7. Choose **Next: Permissions**.
8. IAM includes a list of the AWS managed and customer managed policies in your account. Select the policy to use for the permissions policy or choose **Create policy** to open a new browser tab and create a new policy from scratch. For more information, see step 4 in the procedure [Creating IAM Policies \(Console\) \(p. 443\)](#). After you create the policy, close that tab and return to your original tab. Select the check box next to the permissions policies that you want anyone who assumes the role to have. If you prefer, you can select no policies at this time, and then attach policies to the role later. By default, a role has no permissions.
9. (Optional) Set a [permissions boundary \(p. 370\)](#). This is an advanced feature.

Open the **Set permissions boundary** section and choose **Use a permissions boundary to control the maximum role permissions**. Select the policy to use for the permissions boundary.

10. Choose **Next: Tags**.
11. (Optional) Add metadata to the role by attaching tags as key-value pairs. For more information about using tags in IAM, see [Tagging IAM Users and Roles \(p. 295\)](#).
12. Choose **Next: Review**.
13. For **Role name**, type a name for your role. Role names must be unique within your AWS account. They are not distinguished by case. For example, you cannot create roles named both `PRODROLE` and `prodrole`. Because other AWS resources might reference the role, you cannot edit the name of the role after it has been created.
14. (Optional) For **Role description**, type a description for the new role.
15. Review the role and then choose **Create role**.

Important

Remember that this is only the first half of the configuration required. You must also give individual users in the trusted account permissions to switch to the role in the console, or assume the role programmatically. For more information about this step, see [Granting a User Permissions to Switch Roles \(p. 255\)](#).

Creating an IAM Role (AWS CLI)

Creating a role from the AWS CLI involves multiple steps. When you use the console to create a role, many of the steps are done for you, but with the AWS CLI you must explicitly perform each step yourself.

You must create the role and then assign a permissions policy to the role. Optionally, you can also set the [permissions boundary \(p. 370\)](#) for your role.

To create a role for cross-account access (AWS CLI)

1. Create a role: [aws iam create-role](#)
2. Attach a managed permissions policy to the role: [aws iam attach-role-policy](#)

or

Create an inline permissions policy for the role: [aws iam put-role-policy](#)
3. (Optional) Add custom attributes to the role by attaching tags: [aws iam tag-role](#)

For more information, see [Managing Tags on IAM Entities \(AWS CLI or AWS API\) \(p. 298\)](#).
4. (Optional) Set the [permissions boundary \(p. 370\)](#) for the role: [aws iam put-role-permissions-boundary](#)

A permissions boundary controls the maximum permissions that a role can have. Permissions boundaries are an advanced AWS feature.

The following example shows the first two, and most common steps for creating a cross-account role in a simple environment. This example allows any user in the 123456789012 account to assume the role and view the `example_bucket` Amazon S3 bucket. This example also assumes that you are using a client computer running Windows, and have already configured your command line interface with your account credentials and Region. For more information, see [Configuring the AWS Command Line Interface](#).

In this example, include the following trust policy in the first command when you create the role. This trust policy allows users in the 123456789012 account to assume the role using the `AssumeRole` operation, but only if the user provides MFA authentication using the `SerialNumber` and `TokenCode` parameters. For more information about MFA, see [Using Multi-Factor Authentication \(MFA\) in AWS \(p. 119\)](#).

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Principal": { "AWS": "arn:aws:iam::123456789012:root" },  
            "Action": "sts:AssumeRole",  
            "Condition": { "Bool": { "aws:MultiFactorAuthPresent": "true" } }  
        }  
    ]  
}
```

Important

If your `Principal` element contains the ARN for a specific IAM role or user, then that ARN is transformed to a unique principal ID when the policy is saved. This helps mitigate the risk of someone escalating their permissions by removing and recreating the role or user. You don't normally see this ID in the console because there is also a reverse transformation back to the ARN when the trust policy is displayed. However, if you delete the role or user, then the principal ID appears in the console because AWS can no longer map it back to an ARN. Therefore, if you delete and recreate a user or role referenced in a trust policy's `Principal` element, you must edit the role to replace the ARN.

When you use the second command, you must attach an existing managed policy to the role. The following permissions policy allows anyone who assumes the role to perform only the `ListBucket` action on the `example_bucket` Amazon S3 bucket.

```
{
```

```
"Version": "2012-10-17",
"Statement": [
    {"Effect": "Allow",
     "Action": "s3>ListBucket",
     "Resource": "arn:aws:s3:::example_bucket"
    }
]
```

To create this `Test-UserAccess-Role` role, you must first save the previous trust policy with the name `trustpolicyforacct123456789012.json` to the `policies` folder in your local C: drive. Then save the previous permissions policy as a customer managed policy in your AWS account with the name `PolicyForRole`. You can then use the following commands to create the role and attach the managed policy.

```
# Create the role and attach the trust policy file that allows users in the specified
# account to assume the role.
$ aws iam create-role --role-name Test-UserAccess-Role --assume-role-policy-document
file://C:\policies\trustpolicyforacct123456789012.json

# Attach the permissions policy (in this example a managed policy) to the role to specify
# what it is allowed to do.
$ aws iam attach-role-policy --role-name Test-UserAccess-Role --policy-arn
arn:aws:iam::123456789012:role/PolicyForRole
```

Important

Remember that this is only the first half of the configuration required. You must also give individual users in the trusted account permissions to switch to the role. For more information about this step, see [Granting a User Permissions to Switch Roles \(p. 255\)](#).

After you create the role and grant it permissions to perform AWS tasks or access AWS resources, any users in the 123456789012 account can assume the role. For more information, see [Switching to an IAM Role \(AWS CLI\) \(p. 262\)](#).

Creating an IAM Role (AWS API)

Creating a role from the AWS API involves multiple steps. When you use the console to create a role, many of the steps are done for you, but with the API you must explicitly perform each step yourself. You must create the role and then assign a permissions policy to the role. Optionally, you can also set the [permissions boundary \(p. 370\)](#) for your role.

To create a role in code (AWS API)

1. Create a role: [CreateRole](#)

For the role's trust policy, you can specify a file location.

2. Attach a managed permission policy to the role: [AttachRolePolicy](#)

or

Create an inline permission policy for the role: [PutRolePolicy](#)

Important

Remember that this is only the first half of the configuration required. You must also give individual users in the trusted account permissions to switch to the role. For more information about this step, see [Granting a User Permissions to Switch Roles \(p. 255\)](#).

3. (Optional) Add custom attributes to the user by attaching tags: [TagRole](#)

For more information, see [Managing Tags on IAM Entities \(AWS CLI or AWS API\) \(p. 298\)](#).

4. (Optional) Set the [permissions boundary \(p. 370\)](#) for the role: [PutRolePermissionsBoundary](#)

A permissions boundary controls the maximum permissions that a role can have. Permissions boundaries are an advanced AWS feature.

After you create the role and grant it permissions to perform AWS tasks or access AWS resources, you must grant permissions to users in the account to allow them to assume the role. For more information about assuming a role, see [Switching to an IAM Role \(AWS API\) \(p. 268\)](#).

How to Use an External ID When Granting Access to Your AWS Resources to a Third Party

At times, you need to give a third party access to your AWS resources (delegate access). One important aspect of this scenario is the *External ID*, optional information that you can use in an IAM role trust policy to designate who can assume the role.

Important

AWS does not treat the external ID as a secret. After you create a secret like an access key pair or a password in AWS, you cannot view them again. The external ID for a role can be seen by anyone with permission to view the role.

To use an external ID, update a role trust policy with the external ID of your choice. Then, when uses the AWS CLI or AWS API to assume that role, they must provide the external ID.

For example, let's say that you decide to hire a third-party company called Example Corp to monitor your AWS account and help optimize costs. In order to track your daily spending, Example Corp needs to access your AWS resources. Example Corp also monitors many other AWS accounts for other customers.

Do not give Example Corp access to an IAM user and its long-term credentials in your AWS account. Instead, use an IAM role and its temporary security credentials. An IAM role provides a mechanism to allow a third party to access your AWS resources without needing to share long-term credentials (for example, an IAM user's access key).

You can use an IAM role to establish a trusted relationship between your AWS account and the Example Corp account. After this relationship is established, a member of the Example Corp account can call the AWS STS [AssumeRole](#) API to obtain temporary security credentials. The Example Corp members can then use the credentials to access AWS resources in your account.

Note

For more information about the [AssumeRole](#) and other AWS API operations that you can call to obtain temporary security credentials, see [Requesting Temporary Security Credentials \(p. 309\)](#).

Here's a more detailed breakdown of this scenario:

1. You hire Example Corp, so they create a unique customer identifier for you. They provide you with this unique customer ID and their AWS account number. You need this information to create an IAM role in the next step.

Note

Example Corp can use any string value they want for the `ExternalId`, as long as it is unique for each customer. It can be a customer account number or even a random string of characters, as long as no two customers have the same value. It is not intended to be a 'secret'. Example Corp must provide the `ExternalId` value to each customer. What is crucial is that it must be generated by Example Corp and **not** their customers.

2. You sign in to AWS and create an IAM role that gives Example Corp access to your resources. Like any IAM role, the role has two policies, a permission policy and a trust policy. The role's trust policy specifies who can assume the role. In our sample scenario, the policy specifies the AWS account number of Example Corp as the `Principal`. This allows identities from that account to assume the role. In addition, you add a `Condition` element to the trust policy. This `Condition` tests the `ExternalId` context key to ensure that it matches the unique customer ID from Example Corp. For example:

```
"Principal": {"AWS": "Example Corp's AWS Account ID"},  
"Condition": {"StringEquals": {"sts:ExternalId": "Unique ID Assigned by Example Corp"}}
```

3. The permission policy for the role specifies what the role allows someone to do. For example, you could specify that the role allows someone to manage only your Amazon EC2 and Amazon RDS resources but not your IAM users or groups. In our sample scenario, you use the permission policy to give Example Corp read-only access to all of the resources in your account.
4. After you create the role, you provide the Amazon Resource Name (ARN) of the role to Example Corp.
5. When Example Corp needs to access your AWS resources, someone from the company calls the AWS `sts:AssumeRole` API. The call includes the ARN of the role to assume and the `ExternalId` parameter that corresponds to their customer ID.

If the request comes from someone using Example Corp's AWS account, and if the role ARN and the external ID are correct, the request succeeds. It then provides temporary security credentials that Example Corp can use to access the AWS resources that your role allows.

In other words, when a role policy includes an external ID, anyone who wants to assume the role must be a principal in the role and must include the correct external ID.

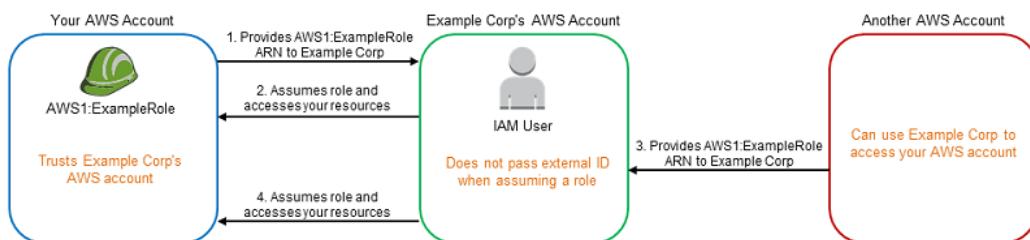
Why Do You Need to Use an External ID?

In abstract terms, the external ID allows the user that is assuming the role to assert the circumstances in which they are operating. It also provides a way for the account owner to permit the role to be assumed only under specific circumstances. The primary function of the external ID is to address and prevent the "confused deputy" problem.

The Confused Deputy Problem

To continue the previous example, Example Corp requires access to certain resources in your AWS account. But in addition to you, Example Corp has other customers and needs a way to access each customer's AWS resources. Instead of asking its customers for their AWS account access keys, which are secrets that should never be shared, Example Corp requests a role ARN from each customer. But another Example Corp customer might be able to guess or obtain your role ARN. That customer could then use your role ARN to gain access to your AWS resources by way of Example Corp. This form of permission escalation is known as the confused deputy problem.

The following diagram illustrates the confused deputy problem.



This diagram assumes the following:

- **AWS1** is your AWS account.
- **AWS1:ExampleRole** is a role in your account. This role's trust policy trusts Example Corp by specifying Example Corp's AWS account as the one that can assume the role.

Here's what happens:

1. When you start using Example Corp's service, you provide the ARN of **AWS1:ExampleRole** to Example Corp.
2. Example Corp uses that role ARN to obtain temporary security credentials to access resources in your AWS account. In this way, you are trusting Example Corp as a "deputy" that can act on your behalf.
3. Another AWS customer also starts using Example Corp's service, and this customer also provides the ARN of **AWS1:ExampleRole** for Example Corp to use. Presumably the other customer learned or guessed the **AWS1:ExampleRole**, which isn't a secret.
4. When the other customer asks Example Corp to access AWS resources in (what it claims to be) its account, Example Corp uses **AWS1:ExampleRole** to access resources in your account.

This is how the other customer could gain unauthorized access to your resources. Because this other customer was able to trick Example Corp into unwittingly acting on your resources, Example Corp is now a "confused deputy."

How Does the External ID Prevent the Confused Deputy Problem?

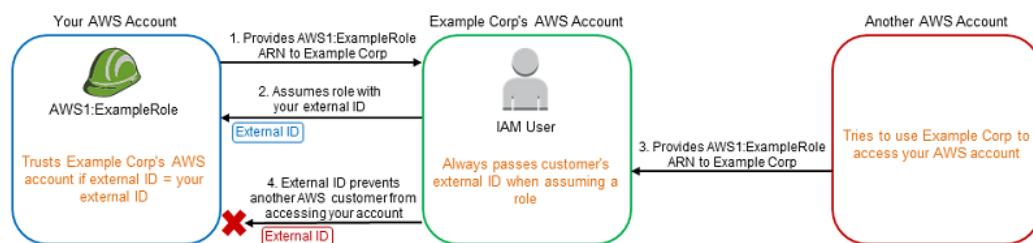
You address the confused deputy problem by including the `ExternalId` condition check in the role's trust policy. The "deputy" company inserts a unique external ID value for each customer into the request for AWS credentials. The external ID is a customer ID value that must be unique among Example Corp's customers and is out of the control of Example Corp's customers. This is why you get it from Example Corp and you don't come up with it on your own. This helps prevent one customer from successfully impersonating another customer. Example Corp always inserts the customer's assigned external ID, so you should never see a request coming from Example Corp with any external ID except your own.

In our scenario, imagine Example Corp's unique identifier for you is "12345," and its identifier for the other customer is "67890." These identifiers are simplified for this scenario. Generally, these identifiers are GUIDs. Assuming that these identifiers are unique among Example Corp's customers, they are sensible values to use for the external ID.

Example Corp gives the external ID value of "12345" to you. You must then add a `Condition` element to the role's trust policy that requires the `sts:ExternalId` value to be 12345, like this:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "sts:AssumeRole",
      "Principal": {"AWS": "Example Corp's AWS Account ID"},
      "Condition": {"StringEquals": {"sts:ExternalId": "12345"}}
    }
  ]
}
```

The `Condition` element in this policy allows Example Corp to assume the role only when the `AssumeRole` API call includes the external ID value of "12345". Example Corp makes sure that whenever it assumes a role on behalf of a customer, it always includes that customer's external ID value in the `AssumeRole` call. Even if another customer supplies Example Corp with your ARN, it cannot control the external ID that Example Corp includes in its request to AWS. This helps prevent an unauthorized customer from gaining access to your resources, as shown in the following diagram.



1. As before, when you start using Example Corp's service, you provide the ARN of **AWS1:ExampleRole** to Example Corp.
2. When Example Corp uses that role ARN to assume the role **AWS1:ExampleRole**, Example Corp includes your external ID ("12345") in the AssumeRole API call. The external ID matches the role's trust policy, so the AssumeRole API call succeeds and Example Corp obtains temporary security credentials to access resources in your AWS account.
3. Another AWS customer also starts using Example Corp's service, and as before, this customer also provides the ARN of **AWS1:ExampleRole** for Example Corp to use.
4. But this time, when Example Corp attempts to assume the role **AWS1:ExampleRole**, it provides the external ID associated with the other customer ("67890"). The other customer has no way to change this. Example Corp does this because the request to use the role came from the other customer, so "67890" indicates the circumstance in which Example Corp is acting. Because you added a condition with your own external ID ("12345") to the trust policy of **AWS1:ExampleRole**, the AssumeRole API call fails. The other customer is prevented from gaining unauthorized access to resources in your account (indicated by the red "X" in the diagram).

The external ID helps prevent any other customer from tricking Example Corp into unwittingly accessing your resources—it mitigates the confused deputy problem.

When Should I Use the External ID?

Use an external ID in the following situations:

- You are an AWS account owner and you have configured a role for a third party that accesses other AWS accounts in addition to yours. You should ask the third party for an external ID that it includes when it assumes your role. Then you check for that external ID in your role's trust policy. Doing so ensures that the external party can assume your role only when it is acting on your behalf.
- You are in the position of assuming roles on behalf of different customers like Example Corp in our previous scenario. You should assign a unique external ID to each customer and instruct them to add the external ID to their role's trust policy. You must then ensure that you always include the correct external ID in your requests to assume roles.

You probably already have a unique identifier for each of your customers, and this unique ID is sufficient for use as an external ID. The external ID is not a special value that you need to create explicitly, or track separately, just for this purpose.

You should always specify the external ID in your AssumeRole API calls. In addition when a customer gives you a role ARN, test whether you can assume the role both with and without the correct external ID. If you can assume the role without the correct external ID, don't store the customer's role ARN in your system. Wait until your customer has updated the role trust policy to require the correct external ID. In this way you help your customers to do the right thing, which helps to keep both of you protected against the confused deputy problem.

Creating a Role to Delegate Permissions to an AWS Service

Many AWS services require that you use roles to allow the service to access resources in other services on your behalf. A role that a service assumes to perform actions on your behalf is called a [service role \(p. 176\)](#). When a role serves a specialized purpose for a service, it is categorized as a [service role for EC2 instances \(p. 176\)](#) (for example), or a [service-linked role \(p. 176\)](#). To see what services support using service-linked roles, or whether a service supports any form of temporary credentials, see [AWS Services That Work with IAM \(p. 586\)](#). To learn how an individual service uses roles, choose the service name in the table to view the documentation for that service.

For information about how roles help you to delegate permissions, see [Roles Terms and Concepts \(p. 176\)](#).

Service Role Permissions

You must configure permissions to allow an IAM entity (such as a user, group, or role) to create or edit a service role.

Note

The ARN for a service-linked role includes a service principal, which is indicated in the policies below as `SERVICE-NAME.amazonaws.com`. Do not try to guess the service principal, because it is case sensitive and the format can vary across AWS services. To view the service principal for a service, see its service-linked role documentation.

To allow an IAM entity to create a specific service role

Add the following policy to the IAM entity that needs to create the service role. This policy allows you to create a service role for the specified service and with a specific name. You can then attach managed or inline policies to that role.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": [  
                "iam:AttachRolePolicy",  
                "iam>CreateRole",  
                "iam:PutRolePolicy"  
            ],  
            "Resource": "arn:aws:iam::*:role/SERVICE-ROLE-NAME"  
        }  
    ]  
}
```

To allow an IAM entity to create any service role

Add the following statement to the permissions policy for the IAM entity that needs to create a service role. This statement allows you to create any service role for any service, and then attach managed or inline policies to that role.

```
{  
    "Effect": "Allow",  
    "Action": [  
        "iam:AttachRolePolicy",  
        "iam>CreateRole",  
        "iam:PutRolePolicy"  
    ],  
    "Resource": "*"  
}
```

To allow an IAM entity to edit a service role

Add the following policy to the IAM entity that needs to edit the service role.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "EditSpecificServiceRole",  
            "Effect": "Allow",  
            "Action": [  
                "iam:AttachRolePolicy",  
                "iam>DeleteRolePolicy",  
                "iam:PutRolePolicy"  
            ]  
        }  
    ]  
}
```

```
        "iam:DetachRolePolicy",
        "iam:GetRole",
        "iam:GetRolePolicy",
        "iam>ListAttachedRolePolicies",
        "iam>ListRolePolicies",
        "iam:PutRolePolicy",
        "iam:UpdateRole",
        "iam:UpdateRoleDescription"
    ],
    "Resource": "arn:aws:iam::*:role/SERVICE-ROLE-NAME"
},
{
    "Sid": "ViewRolesAndPolicies",
    "Effect": "Allow",
    "Action": [
        "iam:GetPolicy",
        "iam>ListRoles"
    ],
    "Resource": ""
}
]
```

To allow an IAM entity to delete a specific service role

Add the following statement to the permissions policy for the IAM entity that needs to delete the specified service role.

```
{
    "Effect": "Allow",
    "Action": "iam>DeleteRole",
    "Resource": "arn:aws:iam::*:role/SERVICE-ROLE-NAME"
}
```

To allow an IAM entity to delete any service role

Add the following statement to the permissions policy for the IAM entity that needs to delete a service role.

```
{
    "Effect": "Allow",
    "Action": "iam>DeleteRole",
    "Resource": "*"
}
```

Creating a Role for an AWS Service (Console)

You can use the AWS Management Console to create a role for a service. Because some services support more than one service role, see the [AWS documentation](#) for your service to see which use case to choose. You can learn how to assign the necessary trust and permissions policies to the role so that the service can assume the role on your behalf. The steps that you can use to control the permissions for your role can vary, depending on how the service defines the use cases, and whether or not you create a service-linked role.

To create a role for an AWS service (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane of the IAM console, choose **Roles**, and then choose **Create role**.
3. For **Select type of trusted entity**, choose **AWS service**.

4. Choose the service that you want to allow to assume this role.
5. Choose the use case for your service. If the specified service has only one use case, it is selected for you. Use cases are defined by the service to include the trust policy that the service requires. Then choose **Next: Permissions**.
6. If possible, select the policy to use for the permissions policy or choose **Create policy** to open a new browser tab and create a new policy from scratch. For more information, see step 4 in the procedure [Creating IAM Policies \(Console\) \(p. 443\)](#). After you create the policy, close that tab and return to your original tab. Select the check box next to the permissions policies that you want the service to have.

Depending on the use case that you selected, the service might allow you to do any of the following:

- Nothing, because the service defines the permissions for the role
 - Allow you to choose from a limited set of permissions
 - Allow you to choose from any permissions
 - Allow you to select no policies at this time, create the policies later, and then attach them to the role
7. (Optional) Set a [permissions boundary \(p. 370\)](#). This is an advanced feature that is available for service roles, but not service-linked roles.

Open the **Set permissions boundary** section and choose **Use a permissions boundary to control the maximum role permissions**. IAM includes a list of the AWS managed and customer managed policies in your account. Select the policy to use for the permissions boundary or choose **Create policy** to open a new browser tab and create a new policy from scratch. For more information, see step 4 in the procedure [Creating IAM Policies \(Console\) \(p. 443\)](#). After you create the policy, close that tab and return to your original tab to select the policy to use for the permissions boundary.

8. Choose **Next: Tags**.
9. (Optional) Add metadata to the role by attaching tags as key-value pairs. For more information about using tags in IAM, see [Tagging IAM Users and Roles \(p. 295\)](#).
10. Choose **Next: Review**.
11. For **Role name**, the degree of role name customization is defined by the service. If the service defines the role's name, this option is not editable. In other cases, the service might define a prefix for the role and allow you to type an optional suffix. Some services allow you to specify the entire name of your role.

If possible, type a role name or role name suffix. Role names must be unique within your AWS account. They are not distinguished by case. For example, you cannot create roles named both **PRODROLE** and **prodrole**. Because other AWS resources might reference the role, you cannot edit the name of the role after it has been created.

12. (Optional) For **Role description**, type a description for the new role.
13. Review the role and then choose **Create role**.

[Creating a Role for a Service \(AWS CLI\)](#)

Creating a role from the AWS CLI involves multiple steps. When you use the console to create a role, many of the steps are done for you, but with the AWS CLI you must explicitly perform each step yourself. You must create the role and then assign a permissions policy to the role. If the service you are working with is Amazon EC2, then you must also create an instance profile and add the role to it. Optionally, you can also set the [permissions boundary \(p. 370\)](#) for your role.

To create a role for an AWS service from the AWS CLI

1. Create a role: [`aws iam create-role`](#)
2. Attach a managed permissions policy to the role: [`aws iam attach-role-policy`](#)

or

- Create an inline permissions policy for the role: [aws iam put-role-policy](#)
3. (Optional) Add custom attributes to the role by attaching tags: [aws iam tag-role](#)
For more information, see [Managing Tags on IAM Entities \(AWS CLI or AWS API\) \(p. 298\)](#).
 4. (Optional) Set the [permissions boundary \(p. 370\)](#) for the role: [aws iam put-role-permissions-boundary](#)

A permissions boundary controls the maximum permissions that a role can have. Permissions boundaries are an advanced AWS feature.

If you are going to use the role with Amazon EC2 or another AWS service that uses Amazon EC2, you must store the role in an instance profile. An instance profile is a container for a role that can be attached to an Amazon EC2 instance when launched. An instance profile can contain only one role, and that limit cannot be increased. If you create the role using the AWS Management Console, the instance profile is created for you with the same name as the role. For more information about instance profiles, see [Using Instance Profiles \(p. 276\)](#). For information about how to launch an EC2 instance with a role, see [Controlling Access to Amazon EC2 Resources](#) in the *Amazon EC2 User Guide for Linux Instances*.

To create an instance profile and store the role in it (AWS CLI)

1. Create an instance profile: [aws iam create-instance-profile](#)
2. Add the role to the instance profile: [aws iam add-role-to-instance-profile](#)

The AWS CLI example command set below demonstrates the first two steps for creating a role and attaching permissions. It also shows the two steps for creating an instance profile and adding the role to the profile. This example trust policy allows the Amazon EC2 service to assume the role and view the `example_bucket` Amazon S3 bucket. The example also assumes that you are running on a client computer running Windows and have already configured your command line interface with your account credentials and Region. For more information, see [Configuring the AWS Command Line Interface](#).

In this example, include the following trust policy in the first command when you create the role. This trust policy allows the Amazon EC2 service to assume the role.

```
{  
    "Version": "2012-10-17",  
    "Statement": {  
        "Effect": "Allow",  
        "Principal": {"Service": "ec2.amazonaws.com"},  
        "Action": "sts:AssumeRole"  
    }  
}
```

When you use the second command, you must attach a permissions policy to the role. The following example permissions policy allows the role to perform only the `ListBucket` action on the `example_bucket` Amazon S3 bucket.

```
{  
    "Version": "2012-10-17",  
    "Statement": {  
        "Effect": "Allow",  
        "Action": "s3>ListBucket",  
        "Resource": "arn:aws:s3:::example_bucket"  
    }  
}
```

To create this `Test-Role-for-EC2` role, you must first save the previous trust policy with the name `trustpolicyforec2.json` and the previous permissions policy with the name `permissionspolicyforec2.json` to the `policies` directory in your local C: drive. You can then use the following commands to create the role, attach the policy, create the instance profile, and add the role to the instance profile.

```
# Create the role and attach the trust policy that allows EC2 to assume this role.  
$ aws iam create-role --role-name Test-Role-for-EC2 --assume-role-policy-document file://C:  
\policies\trustpolicyforec2.json  
  
# Embed the permissions policy (in this example an inline policy) to the role to specify  
what it is allowed to do.  
$ aws iam put-role-policy --role-name Test-Role-for-EC2 --policy-name Permissions-Policy-  
For-Ec2 --policy-document file://permissionspolicyforec2.json  
  
# Create the instance profile required by EC2 to contain the role  
$ aws iam create-instance-profile --instance-profile-name EC2-ListBucket-S3  
  
# Finally, add the role to the instance profile  
$ aws iam add-role-to-instance-profile --instance-profile-name EC2-ListBucket-S3 --role-  
name Test-Role-for-EC2
```

When you launch the EC2 instance, specify the instance profile name in the **Configure Instance Details** page if you use the AWS console. If you use the `aws ec2 run-instances` CLI command, specify the `--iam-instance-profile` parameter.

Creating a Role for a Service (AWS API)

Creating a role from the AWS API involves multiple steps. When you use the console to create a role, many of the steps are done for you, but with the API you must explicitly perform each step yourself. You must create the role and then assign a permissions policy to the role. If the service you are working with is Amazon EC2, then you must also create an instance profile and add the role to it. Optionally, you can also set the [permissions boundary \(p. 370\)](#) for your role.

To create a role for an AWS service (AWS API)

1. Create a role: [CreateRole](#)
For the role's trust policy, you can specify a file location.
2. Attach a managed permissions policy to the role: [AttachRolePolicy](#)
or
Create an inline permissions policy for the role: [PutRolePolicy](#)
3. (Optional) Add custom attributes to the user by attaching tags: [TagRole](#)
For more information, see [Managing Tags on IAM Entities \(AWS CLI or AWS API\) \(p. 298\)](#).
4. (Optional) Set the [permissions boundary \(p. 370\)](#) for the role: [PutRolePermissionsBoundary](#)
A permissions boundary controls the maximum permissions that a role can have. Permissions boundaries are an advanced AWS feature.

If you are going to use the role with Amazon EC2 or another AWS service that uses Amazon EC2, you must store the role in an instance profile. An instance profile is a container for a role. Each instance profile can contain only one role, and that limit cannot be increased. If you create the role in the AWS Management Console, the instance profile is created for you with the same name as the role. For more information about instance profiles, see [Using Instance Profiles \(p. 276\)](#). For information about how to launch an Amazon EC2 instance with a role, see [Controlling Access to Amazon EC2 Resources](#) in the *Amazon EC2 User Guide for Linux Instances*.

To create an instance profile and store the role in it (AWS API)

1. Create an instance profile: [CreateInstanceProfile](#)
2. Add the role to the instance profile: [AddRoleToInstanceProfile](#)

Creating a Role for a Third-Party Identity Provider (Federation)

You can use identity providers instead of creating IAM users in your AWS account. With an identity provider (IdP), you can manage your user identities outside of AWS and give these external user identities permissions to access AWS resources in your account. For more information about federation and identity providers, see [Identity Providers and Federation \(p. 184\)](#).

Creating a Role for Federated Users (Console)

The procedures for creating a role for federated users depend on your choice of third-party providers:

- For Web Identity or OpenID Connect (OIDC), see [Creating a Role for Web Identity or OpenID Connect Federation \(Console\) \(p. 244\)](#).
- For SAML 2.0, see [Creating a Role for SAML 2.0 Federation \(Console\) \(p. 248\)](#).

Creating a Role for Federated Access (AWS CLI)

The steps to create a role for the supported identity providers (OIDC or SAML) from the AWS CLI are identical. The difference is in the contents of the trust policy that you create in the prerequisite steps. Begin by following the steps in the **Prerequisites** section for the type of provider you are using:

- For an OIDC provider, see [Prerequisites for Creating a Role for Web Identity or OIDC \(p. 244\)](#).
- For a SAML provider, see [Prerequisites for Creating a Role for SAML \(p. 248\)](#).

Creating a role from the AWS CLI involves multiple steps. When you use the console to create a role, many of the steps are done for you, but with the AWS CLI you must explicitly perform each step yourself. You must create the role and then assign a permissions policy to the role. Optionally, you can also set the [permissions boundary \(p. 370\)](#) for your role.

To create a role for identity federation (AWS CLI)

1. Create a role: `aws iam create-role`
2. Attach a permissions policy to the role: `aws iam attach-role-policy`

or

Create an inline permissions policy for the role: `aws iam put-role-policy`
3. (Optional) Add custom attributes to the role by attaching tags: `aws iam tag-role`

For more information, see [Managing Tags on IAM Entities \(AWS CLI or AWS API\) \(p. 298\)](#).
4. (Optional) Set the [permissions boundary \(p. 370\)](#) for the role: `aws iam put-role-permissions-boundary`

A permissions boundary controls the maximum permissions that a role can have. Permissions boundaries are an advanced AWS feature.

The following example shows the first two, and most common, steps for creating an identity provider role in a simple environment. This example allows any user in the 123456789012 account to assume the role and view the `example_bucket` Amazon S3 bucket. This example also assumes that you are

running the AWS CLI on a computer running Windows, and have already configured the AWS CLI with your credentials. For more information, see [Configuring the AWS Command Line Interface](#).

In this example, include the following trust policy in the first command when you create the role. This trust policy allows users in the 123456789012 account to assume the role using the `AssumeRole` operation, but only if the user provides MFA authentication using the `SerialNumber` and `TokenCode` parameters. For more information about MFA, see [Using Multi-Factor Authentication \(MFA\) in AWS \(p. 119\)](#).

The following example trust policy is designed for a mobile app if the user signs in using Amazon Cognito. In this example, `us-east:12345678-ffff-ffff-ffff-123456` represents the identity pool ID assigned by Amazon Cognito.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "RoleForCognito",  
            "Effect": "Allow",  
            "Principal": {"Federated": "cognito-identity.amazonaws.com"},  
            "Action": "sts:AssumeRoleWithWebIdentity",  
            "Condition": {"StringEquals": {"cognito-identity.amazonaws.com:aud": "us-east:12345678-ffff-ffff-ffff-123456"}},  
        }  
    ]  
}
```

The following permissions policy allows anyone who assumes the role to perform only the `ListBucket` action on the `example_bucket` Amazon S3 bucket.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": "s3>ListBucket",  
            "Resource": "arn:aws:s3:::example_bucket"  
        }  
    ]  
}
```

To create this `Test-Cognito-Role` role, you must first save the previous trust policy with the name `trustpolicyforcognitofederation.json` and the previous permissions policy with the name `permsspolicyforcognitofederation.json` to the `policies` folder in your local C: drive. You can then use the following commands to create the role and attach the inline policy.

```
# Create the role and attach the trust policy that enables users in an account to assume  
# the role.  
$ aws iam create-role --role-name Test-Cognito-Role --assume-role-policy-document file://C:  
\\policies\\trustpolicyforcognitofederation.json  
  
# Attach the permissions policy to the role to specify what it is allowed to do.  
aws iam put-role-policy --role-name Test-Cognito-Role --policy-name Perms-Policy-For-  
CognitoFederation --policy-document file://C:\\policies\\permsspolicyforcognitofederation.json
```

Creating a Role for Federated Access (AWS API)

The steps to create a role for the supported identity providers (OIDC or SAML) from the AWS CLI are identical. The difference is in the contents of the trust policy that you create in the prerequisite steps. Begin by following the steps in the **Prerequisites** section for the type of provider you are using:

- For an OIDC provider, see [Prerequisites for Creating a Role for Web Identity or OIDC \(p. 244\)](#).
- For a SAML provider, see [Prerequisites for Creating a Role for SAML \(p. 248\)](#).

To create a role for identity federation (AWS API)

1. Create a role: [CreateRole](#)
 2. Attach a permissions policy to the role: [AttachRolePolicy](#)

or

Create an inline permissions policy for the role: PutRolePolicy

3. (Optional) Add custom attributes to the user by attaching tags: [TagRole](#)

For more information, see [Managing Tags on IAM Entities \(AWS CLI or AWS API\) \(p. 298\)](#).

4. (Optional) Set the permissions boundary (p. 370) for the role: PutRolePermissionsBoundary

A permissions boundary controls the maximum permissions that a role can have. Permissions boundaries are an advanced AWS feature.

Creating a Role for Web Identity or OpenID Connect Federation (Console)

You can use Web Identity or OpenID Connect Federation (OIDC) identity providers instead of creating IAM users in your AWS account. With an identity provider (IdP), you can manage your user identities outside of AWS and give these external user identities permissions to access AWS resources in your account. For more information about federation and identity providers, see [Identity Providers and Federation \(p. 184\)](#).

Prerequisites for Creating a Role for Web Identity or OIDC

Before you can create a role for web identity federation, you must first complete the following prerequisite steps.

To prepare to create a role for web identity federation

1. Sign up as a developer with one or more IdPs. If you are creating an app that needs access to your AWS resources, you also configure your app with the provider information. When you do, the provider gives you an application or audience ID that's unique to your app. (Different providers use different terminology for this process. This guide uses the term *configure* for the process of identifying your app with the provider.) You can configure multiple apps with each provider, or multiple providers with a single app. View information about using the identity providers:
 - [Login with Amazon Developer Center](#)
 - [Add Facebook Login to Your App or Website](#) on the Facebook developers site.
 - [Using OAuth 2.0 for Login \(OpenID Connect\)](#) on the Google developers site.
 2. After getting the required information from the identity provider, create an identity provider in IAM. For more information, see [Creating OpenID Connect \(OIDC\) Identity Providers \(p. 194\)](#).
 3. Prepare the policies for the role that the IdP-authenticated users will assume. As with any role, a role for a mobile app includes two policies. One is the trust policy that specifies who can assume the role. The other is the permissions policy that specifies the AWS actions and resources that the mobile app is allowed or denied access to.

For web identity providers, we recommend that you use [Amazon Cognito](#) to manage identities. In this case, use a trust policy similar to this example.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Principal": {"Federated": "cognito-identity.amazonaws.com"},  
            "Action": "sts:AssumeRoleWithWebIdentity",  
            "Condition": {"StringEquals": {"cognito-identity.amazonaws.com:aud": "https://cognito-idp.us-east-1.amazonaws.com/"}},  
            "NotBefore": "2018-01-01T00:00:00Z",  
            "NotAfter": "2018-01-02T00:00:00Z"  
        }  
    ]  
}
```

```

    "Condition": {
        "StringEquals": {"cognito-identity.amazonaws.com:aud": "us-east-2:12345678-
abcd-abcd-abcd-123456"}, 
        "ForAnyValue:StringLike": {"cognito-identity.amazonaws.com:amr": 
"unauthenticated"} 
    }
}

```

Replace `us-east-2:12345678-abcd-abcd-abcd-123456` with the identity pool ID that Amazon Cognito assigned to you.

If you manually configure a web identity IdP, when you create the trust policy, you must use three values that ensure that only your app can assume the role:

- For the `Action` element, use the `sts:AssumeRoleWithWebIdentity` action.
- For the `Principal` element, use the string `{"Federated":providerUrl/providerArn"}`.
 - For some common OpenID Connect (OIDC) IdPs, the `providerUrl` is a URL. The following examples include methods to specify the principal for some common IdPs:

```
"Principal": {"Federated": "cognito-identity.amazonaws.com"}
```

```
"Principal": {"Federated": "www.amazon.com"}
```

```
"Principal": {"Federated": "graph.facebook.com"}
```

```
"Principal": {"Federated": "accounts.google.com"}
```

- For other OIDC providers, use the ARN of the OIDC identity provider that you created in [Step 2](#), such as the following example:

```
"Principal": {"Federated": "arn:aws:iam::123456789012:oidc-provider/
server.example.com"}
```

- For the `Condition` element, use a `StringEquals` condition to limit permissions. Test the identity pool ID for Amazon Cognito or the app ID for other providers. It should match the app ID that you received when you configured the app with the IdP. This ensures that the request is coming from your app. Create a condition element similar to the following examples, depending on the IdP that you are using:

```
"Condition": {"StringEquals": {"cognito-identity.amazonaws.com:aud": "us-
east:12345678-ffff-ffff-ffff-123456"}}
```

```
"Condition": {"StringEquals": {"www.amazon.com:app_id":
"amzn1.application-oa2-123456"}}
```

```
"Condition": {"StringEquals": {"graph.facebook.com:app_id":
"111222333444555"}}
```

```
"Condition": {"StringEquals": {"accounts.google.com:aud":
"66677788899900pro0"}}
```

For OIDC providers, use the fully qualified URL of the OIDC IdP with the `aud` context key, such as the following example:

```
"Condition": {"StringEquals": {"server.example.com:aud":
"appid_from_oidc_idp"}}
```

Notice that the values for the principal in the trust policy for the role are specific to an IdP. A role can specify only one principal. Therefore, if the mobile app allows users to sign in from more than one

If you want to support multiple IdPs, you must create a separate role for each IdP that you want to support. Therefore, you should create separate trust policies for each IdP.

The following example trust policy is designed for a mobile app if the user signs in from Login with Amazon. In the example, `amzn1.application-oa2-123456` represents the app ID that Amazon assigned when you configured the app using Login with Amazon.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Sid": "RoleForLoginWithAmazon",
        "Effect": "Allow",
        "Principal": {"Federated": "www.amazon.com"},
        "Action": "sts:AssumeRoleWithWebIdentity",
        "Condition": {"StringEquals": {"www.amazon.com:app_id": "amzn1.application-oa2-123456"}}
    }]
}
```

The following example trust policy is designed for a mobile app if the user signs in from Facebook. In this example, `111222333444555` represents the app ID assigned by Facebook.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Sid": "RoleForFacebook",
        "Effect": "Allow",
        "Principal": {"Federated": "graph.facebook.com"},
        "Action": "sts:AssumeRoleWithWebIdentity",
        "Condition": {"StringEquals": {"graph.facebook.com:app_id": "111222333444555"}}
    }]
}
```

The following example trust policy is designed for a mobile app if the user signs in from Google. In this example, `666777888999000` represents the app ID assigned by Google.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Sid": "RoleForGoogle",
        "Effect": "Allow",
        "Principal": {"Federated": "accounts.google.com"},
        "Action": "sts:AssumeRoleWithWebIdentity",
        "Condition": {"StringEquals": {"accounts.google.com:aud": "666777888999000"}}
    }]
}
```

The following example trust policy is designed for a mobile app if the user signs in using Amazon Cognito. In this example, `us-east-12345678-ffff-ffff-ffff-123456` represents the identity pool ID assigned by Amazon Cognito.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Sid": "RoleForCognito",
        "Effect": "Allow",
        "Principal": "cognito-identity.amazonaws.com"
    }]
}
```

```
"Principal": {"Federated": "cognito-identity.amazonaws.com"},  
  "Action": "sts:AssumeRoleWithWebIdentity",  
  "Condition": {"StringEquals": {"cognito-identity.amazonaws.com:aud": "us-  
east:12345678-ffff-ffff-ffff-123456"}}  
}]  
}
```

Creating a Role for Web Identity or OIDC

After you complete the prerequisites, you can create the role in IAM. The following procedure describes how to create the role for web identity/OIDC federation in the AWS Management Console. To create a role from the AWS CLI or AWS API, see the procedures at [Creating a Role for a Third-Party Identity Provider \(Federation\) \(p. 242\)](#).

Important

If you are using Amazon Cognito, you should use the Amazon Cognito console to set up the roles. Otherwise, use the IAM console to create a role for web identity federation.

To create an IAM role for web identity federation

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Roles** and then choose **Create role**.
3. Choose the **Web identity** role type.
4. For **Identity provider**, choose the identity provider for your role:
 - If you're creating a role for an individual web identity provider, choose **Login with Amazon, Facebook, or Google**.

Note

You must create a separate role for each identity provider that you want to support.

- If you're creating an advanced scenario role for Amazon Cognito, choose **Amazon Cognito**.

Note

You need to manually create a role for use with Amazon Cognito only when you are working on an advanced scenario. Otherwise, Amazon Cognito can create roles for you. For more information about Amazon Cognito, see [Amazon Cognito Identity](#) in the [AWS Mobile SDK for iOS Developer Guide](#) and [Amazon Cognito Identity](#) in the [AWS Mobile SDK for Android Developer Guide](#).

5. Type the identifier for your application. The label of the identifier changes depending on which provider you choose:
 - If you're creating a role for Login with Amazon, type the app ID into the **Application ID** box.
 - If you're creating a role for Facebook, type the app ID into the **Application ID** box.
 - If you're creating a role for Google, type the audience name into the **Audience** box.
 - If you're creating a role for Amazon Cognito, type the ID of the identity pool that you have created for your Amazon Cognito applications into the **Identity Pool ID** box.
6. (Optional) Click **Add condition (optional)** to create additional conditions that must be met before users of your application can use the permissions that the role grants. For example, you can add a condition that grants access to AWS resources only for a specific IAM user ID.
7. Review your web identity information and then choose **Next: Permissions**.
8. IAM includes a list of the AWS managed and customer managed policies in your account. Select the policy to use for the permissions policy or choose **Create policy** to open a new browser tab and create a new policy from scratch. For more information, see step 4 in the procedure [Creating IAM Policies \(Console\) \(p. 443\)](#). After you create the policy, close that tab and return to your original

tab. Select the check box next to the permissions policies that you want web identity users to have. If you prefer, you can select no policies at this time, and then attach policies to the role later. By default, a role has no permissions.

9. (Optional) Set a [permissions boundary \(p. 370\)](#). This is an advanced feature.

Open the **Set permissions boundary** section and choose **Use a permissions boundary to control the maximum role permissions**. Select the policy to use for the permissions boundary.

10. Choose **Next: Tags**.
11. (Optional) Add metadata to the role by attaching tags as key-value pairs. For more information about using tags in IAM, see [Tagging IAM Users and Roles \(p. 295\)](#).
12. Choose **Next: Review**.
13. For **Role name**, type a role name. Role names must be unique within your AWS account. They are not distinguished by case. For example, you cannot create roles named both **PRODROLE** and **prodrole**. Because other AWS resources might reference the role, you cannot edit the name of the role after it has been created.
14. (Optional) For **Role description**, type a description for the new role.
15. Review the role and then choose **Create role**.

Creating a Role for SAML 2.0 Federation (Console)

You can use SAML 2.0 federation instead of creating IAM users in your AWS account. With an identity provider (IdP), you can manage your user identities outside of AWS and give these external user identities permissions to access AWS resources in your account. For more information about federation and identity providers, see [Identity Providers and Federation \(p. 184\)](#).

Prerequisites for Creating a Role for SAML

Before you can create a role for SAML 2.0 federation, you must first complete the following prerequisite steps:

To prepare to create a role for SAML 2.0 federation

1. Before you create a role for SAML-based federation, you must create a SAML provider in IAM. For more information, see [Creating IAM SAML Identity Providers \(p. 201\)](#).
2. Prepare the policies for the role that the SAML 2.0–authenticated users will assume. As with any role, a role for the SAML federation includes two policies. One is the role trust policy that specifies who can assume the role. The other is the IAM permissions policy that specifies the AWS actions and resources that the federated user is allowed or denied access to.

When you create the trust policy for your role, you must use three values that ensure that the role can be assumed only by your application:

- For the **Action** element, use the `sts:AssumeRoleWithSAML` action.
- For the **Principal** element, use the string `{"Federated": "ARNofIdentityProvider"}`. Replace `ARNofIdentityProvider` with the ARN of the [SAML identity provider \(p. 190\)](#) that you created in Step 1.
- For the **Condition** element, use a `StringEquals` condition to test that the `saml:aud` attribute from the SAML response matches the SAML federation endpoint for AWS.

The following example trust policy is designed for a SAML federated user:

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

```
    "Effect": "Allow",
    "Action": "sts:AssumeRoleWithSAML",
    "Principal": {"Federated": "arn:aws:iam::ACCOUNT-ID-WITHOUT-HYPHENS:saml-provider/PROVIDER-NAME"},
    "Condition": {"StringEquals": {"SAML:aud": "https://signin.aws.amazon.com/saml"}}
}
```

Replace the principal ARN with the actual ARN for the SAML provider that you created in IAM. It will have your own account ID and provider name.

Creating a Role for SAML

After you complete the prerequisite steps, you can create the role for SAML-based federation.

To create a role for SAML-based federation

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane of the IAM console, choose **Roles** and then choose **Create role**.
3. Choose the **SAML 2.0 federation** role type.
4. For **SAML Provider**, choose the provider for your role.
5. Choose the SAML 2.0 access level method.
 - Choose **Allow programmatic access only** to create a role that can be assumed programmatically from the AWS API or AWS CLI.
 - Choose **Allow programmatic and AWS Management Console access** to create a role that can be assumed programmatically and from the console.

The roles created by both are similar, but the role that can also be assumed from the console includes a trust policy with a particular condition. That condition explicitly ensures that the SAML audience (SAML : aud attribute) is set to the AWS sign-in endpoint for SAML (<https://signin.aws.amazon.com/saml>).

6. If you're creating a role for programmatic access, choose an attribute from the **Attribute** list. Then in the **Value** box, type a value to include in the role. This restricts role access to users from the identity provider whose SAML authentication response (assertion) includes the attributes that you specify. You must specify at least one attribute to ensure that your role is limited to a subset of users at your organization.

If you're creating a role for programmatic and console access, the SAML : aud attribute is automatically added and set to the URL of the AWS SAML endpoint (<https://signin.aws.amazon.com/saml>).

7. To add more attribute-related conditions to the trust policy, choose **Add condition (optional)**, select the additional condition, and specify a value.

Note

The list includes the most commonly used SAML attributes. IAM supports additional attributes that you can use to create conditions. (For a list of the supported attributes, see [Available Keys for SAML Federation](#) in the topic [IAM JSON Policy Elements Reference \(p. 598\)](#).) If you need a condition for a supported SAML attribute that's not in the list, you can manually add that condition. To do that, edit the trust policy after you create the role.

8. Review your SAML 2.0 trust information and then choose **Next: Permissions**.
9. IAM includes a list of the AWS managed and customer managed policies in your account. Select the policy to use for the permissions policy or choose **Create policy** to open a new browser tab and

create a new policy from scratch. For more information, see step 4 in the procedure [Creating IAM Policies \(Console\) \(p. 443\)](#). After you create the policy, close that tab and return to your original tab. Select the check box next to the permissions policies that you want web identity users to have. If you prefer, you can select no policies at this time, and then attach policies to the role later. By default, a role has no permissions.

10. (Optional) Set a [permissions boundary \(p. 370\)](#). This is an advanced feature.

Open the **Set permissions boundary** section and choose **Use a permissions boundary to control the maximum role permissions**. Select the policy to use for the permissions boundary.

11. Choose **Next: Tags**.
12. (Optional) Add metadata to the role by attaching tags as key-value pairs. For more information about using tags in IAM, see [Tagging IAM Users and Roles \(p. 295\)](#).
13. Choose **Next: Review**.
14. For **Role name**, type a role name. Role names must be unique within your AWS account. They are not distinguished by case. For example, you cannot create roles named both **PRODROLE** and **prodrole**. Because other AWS resources might reference the role, you cannot edit the name of the role after it has been created.
15. (Optional) For **Role description**, type a description for the new role.
16. Review the role and then choose **Create role**.

After you create the role, you complete the SAML trust by configuring your identity provider software with information about AWS. This information includes the roles that you want your federated users to use. This is referred to as configuring the relying party trust between your IdP and AWS. For more information, see [Configuring your SAML 2.0 IdP with Relying Party Trust and Adding Claims \(p. 203\)](#).

Examples of Policies for Delegating Access

The following examples show how you can allow or grant an AWS account access to the resources in another AWS account. To learn how to create an IAM policy using these example JSON policy documents, see the section called [“Creating Policies on the JSON Tab” \(p. 446\)](#).

Topics

- [Using Roles to Delegate Access to Another AWS Account's Resources \(p. 250\)](#)
- [Using a Policy to Delegate Access To Services \(p. 251\)](#)
- [Using a Resource-Based Policy to Delegate Access to an Amazon S3 Bucket in Another Account \(p. 251\)](#)
- [Using a Resource-Based Policy to Delegate Access to an Amazon SQS Queue in Another Account \(p. 252\)](#)
- [Cannot Delegate Access When the Account is Denied Access \(p. 253\)](#)

Using Roles to Delegate Access to Another AWS Account's Resources

For a tutorial that shows how to use IAM roles to grant users in one account access to AWS resources that are in another account, see [Tutorial: Delegate Access Across AWS Accounts Using IAM Roles \(p. 30\)](#).

Important

You can include the ARN for a specific role or user in the `Principal` element of a role trust policy. When you save the policy, AWS transforms the ARN to a unique principal ID. This helps mitigate the risk of someone escalating their privileges by removing and recreating the role or user. You don't normally see this ID in the console, because there is also a reverse transformation back to the ARN when the trust policy is displayed. However, if you delete the role or user, then the relationship is broken. The policy no longer applies, even if you recreate the user or role because it does not match the principal ID stored in the trust policy. When this

happens, the principal ID shows up in the console because AWS can no longer map it back to an ARN. The result is that if you delete and recreate a user or role referenced in a trust policy's `Principal` element, you must edit the role to replace the ARN. It is transformed into the new principal ID when you save the policy.

Using a Policy to Delegate Access To Services

The following example shows a policy that can be attached to a role. The policy enables two services, Amazon EMR and AWS Data Pipeline, to assume the role. The services can then perform any tasks granted by the permissions policy assigned to the role (not shown). To specify multiple service principals, you do not specify two `Service` elements; you can have only one. Instead, you use an array of multiple service principals as the value of a single `Service` element.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "elasticmapreduce.amazonaws.com",
          "datapipeline.amazonaws.com"
        ]
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Using a Resource-Based Policy to Delegate Access to an Amazon S3 Bucket in Another Account

In this example, account A uses a resource-based policy (an Amazon S3 [bucket policy](#)) to grant account B full access to account A's S3 bucket. Then account B creates an IAM user policy to delegate that access to account A's bucket to one of the users in account B.

The S3 bucket policy in account A might look like the following policy. In this example, account A's S3 bucket is named *mybucket*, and account B's account number is 111122223333. It does not specify any individual users or groups in account B, only the account itself.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AccountBAccess1",
      "Effect": "Allow",
      "Principal": {"AWS": "111122223333"},
      "Action": "s3:*",
      "Resource": [
        "arn:aws:s3:::mybucket",
        "arn:aws:s3:::mybucket/*"
      ]
    }
  ]
}
```

Alternatively, account A can use Amazon S3 [Access Control Lists \(ACLs\)](#) to grant account B access to an S3 bucket or a single object within a bucket. In that case, the only thing that changes is how account A grants access to account B. Account B still uses a policy to delegate access to an IAM group in account B, as described in the next part of this example. For more information about controlling access on S3 buckets and objects, go to [Access Control](#) in the *Amazon Simple Storage Service Developer Guide*.

The administrator of account B might create the following policy sample. The policy allows read access to a group or user in account B. The preceding policy grants access to account B. However, individual groups and users in account B cannot access the resource until a group or user policy explicitly grants permissions to the resource. The permissions in this policy can only be a subset of those in the preceding cross-account policy. Account B cannot grant more permissions to its groups and users than account A granted to account B in the first policy. In this policy, the `Action` element is explicitly defined to allow only `List` actions, and the `Resource` element of this policy matches the `Resource` for the bucket policy implemented by account A.

To implement this policy account B uses IAM to attach it to the appropriate user (or group) in account B.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3>List*",
    "Resource": [
      "arn:aws:s3:::mybucket",
      "arn:aws:s3:::mybucket/*"
    ]
  }
}
```

Using a Resource-Based Policy to Delegate Access to an Amazon SQS Queue in Another Account

In the following example, account A has an Amazon SQS queue that uses a resource-based policy attached to the queue to grant queue access to account B. Then account B uses an IAM group policy to delegate access to a group in account B.

The following example queue policy gives account B permission to perform the `SendMessage` and `ReceiveMessage` actions on account A's queue named `queue1`, but only between noon and 3:00 p.m. on November 30, 2014. Account B's account number is 1111-2222-3333. Account A uses Amazon SQS to implement this policy.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {"AWS": "111122223333"},
    "Action": [
      "sns:SendMessage",
      "sns:ReceiveMessage"
    ],
    "Resource": [ "arn:aws:sns:*:123456789012:queue1" ],
    "Condition": {
      "DateGreaterThan": {"aws:CurrentTime": "2014-11-30T12:00Z"},
      "DateLessThan": {"aws:CurrentTime": "2014-11-30T15:00Z"}
    }
  }
}
```

Account B's policy for delegating access to a group in account B might look like the following example. Account B uses IAM to attach this policy to a group (or user).

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
```

```
    "Action": "sns:*",
    "Resource": "arn:aws:sns::123456789012:queue1"
}
```

In the preceding IAM user policy example, account B uses a wildcard to grant its user access to all Amazon SQS actions on account A's queue. However account B can delegate access only to the extent that account B has been granted access. The account B group that has the second policy can access the queue only between noon and 3:00 p.m. on November 30, 2014. The user can only perform the SendMessage and ReceiveMessage actions, as defined in account A's Amazon SQS queue policy.

Cannot Delegate Access When the Account is Denied Access

An AWS account cannot delegate access to another account's resources if the other account has explicitly denied access to the user's parent account. The deny propagates to the users under that account whether or not the users have existing policies granting them access.

For example, account A writes a bucket policy on account A's S3 bucket that explicitly denies account B access to account A's bucket. But account B writes an IAM user policy that grants a user in account B access to account A's bucket. The explicit deny applied to account A's S3 bucket propagates to the users in account B. It overrides the IAM user policy granting access to the user in account B. (For detailed information how permissions are evaluated, see [Policy Evaluation Logic \(p. 636\)](#).)

Account A's bucket policy might look like the following policy. In this example, account A's S3 bucket is named *mybucket*, and account B's account number is 1111-2222-3333. Account A uses Amazon S3 to implement this policy.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AccountBDeny",
      "Effect": "Deny",
      "Principal": {"AWS": "111122223333"},
      "Action": "s3:*",
      "Resource": "arn:aws:s3:::mybucket/*"
    }
  ]
}
```

This explicit deny overrides any policies in account B that provide permission to access the S3 bucket in account A.

Using IAM Roles

Before an IAM user, application, or service can use a role that you created, you must grant permissions to switch to the role. You can use any policy attached to one of an IAM user's groups or to the user itself to grant the necessary permissions. This section describes how to grant users permission to use a role, and then how the user can switch to a role using the AWS Management Console, the Tools for Windows PowerShell, the AWS Command Line Interface (AWS CLI) and the [AssumeRole API](#).

Important

If you create a role programmatically instead of in the IAM console, then you have an option to add a Path of up to 512 characters in addition to the RoleName, which can be up to 64 characters long. However, if you intend to use a role with the **Switch Role** feature in the AWS console, then the combined Path and RoleName cannot exceed 64 characters.

You can switch roles from the AWS Management Console. You can assume a role by calling an AWS CLI or API operation or by using a custom URL. The method that you use determines who can assume the role and how long the role session can last.

Comparing methods for using roles

Method	Who can assume the role	Method to specify credential lifetime	Credential lifetime (min max default)
AWS Management Console	IAM user (by switching roles (p. 260))	None	1h 1h 1h
assume-role CLI or AssumeRole API operation	IAM user or role ¹	duration-seconds CLI or DurationSeconds API parameter	15m Maximum session duration setting ² 1hr
assume-role-with-saml CLI or AssumeRoleWithSAML API operation	Any user authenticated using SAML	duration-seconds CLI or DurationSeconds API parameter	15m Maximum session duration setting ² 1hr
assume-role-with-web-identity CLI or AssumeRoleWithWebIdentity API operation	Any user authenticated using a web identity provider	duration-seconds CLI or DurationSeconds API parameter	15m Maximum session duration setting ² 1hr
Console URL (p. 213) constructed with AssumeRole	IAM user or role	SessionDuration HTML parameter in the URL	15m 12hr 1hr
Console URL (p. 213) constructed with AssumeRoleWithSAML	Any user authenticated using SAML	SessionDuration HTML parameter in the URL	15m 12hr 1hr
Console URL (p. 213) constructed with AssumeRoleWithWebIdentity	Any user authenticated using a web identity provider	SessionDuration HTML parameter in the URL	15m 12hr 1hr

¹ Using the credentials for one role to assume a different role is called [role chaining \(p. 177\)](#). When you use role chaining, your new credentials are limited to a maximum duration of one hour.

² The maximum session duration is a setting that you can apply to a role from the console, the AWS CLI, or the API. This setting specifies the maximum session duration for the role when it is assumed from the CLI or API. This setting can have a value from 1 hour to 12 hours. For details about the maximum session duration setting, see [Modifying a Role \(p. 279\)](#). This setting determines the maximum session duration that you can request when you get the role credentials. For example, when you use the [AssumeRole*](#) API operations to assume a role, you can specify a session length using the DurationSeconds parameter. Use this parameter to specify the length of the role session from 900 seconds (15 minutes) up to the maximum session duration setting for the role. To learn how to view the maximum value for your role, see [View the Maximum Session Duration Setting for a Role \(p. 255\)](#) later in this page.

Note

The maximum session duration setting applies only to sessions created using the [AssumeRole*](#) API operations or [assume-role*](#) CLI commands. The setting does not limit sessions assumed by AWS services.

Topics

- [View the Maximum Session Duration Setting for a Role \(p. 255\)](#)
- [Granting a User Permissions to Switch Roles \(p. 255\)](#)
- [Granting a User Permissions to Pass a Role to an AWS Service \(p. 258\)](#)
- [Switching to a Role \(Console\) \(p. 260\)](#)
- [Switching to an IAM Role \(AWS CLI\) \(p. 262\)](#)
- [Switching to an IAM Role \(Tools for Windows PowerShell\) \(p. 266\)](#)
- [Switching to an IAM Role \(AWS API\) \(p. 268\)](#)
- [Using an IAM Role to Grant Permissions to Applications Running on Amazon EC2 Instances \(p. 269\)](#)
- [Revoking IAM Role Temporary Security Credentials \(p. 277\)](#)

View the Maximum Session Duration Setting for a Role

When you use an AWS CLI or API operation to assume a role, you can specify a value for the `DurationSeconds` parameter. You can use this parameter to specify the duration of the role session, from 900 seconds (15 minutes) up to the **Maximum CLI/API session duration** setting for the role. Before you specify the parameter, you should view this setting for your role. If you specify a value for the `DurationSeconds` parameter that is higher than the maximum setting, the operation fails.

To view a role's maximum session duration (console)

1. In the navigation pane of the IAM console, choose **Roles**.
2. Choose the name of the role that you want to view.
3. Next to **Maximum CLI/API session duration**, view the maximum session length that you can specify in your AWS CLI or API operation.

To view a role's maximum session duration setting (AWS CLI)

1. If you don't know the name of the role that you want to assume, run the following command to list the roles in your account:
 - `aws iam list-roles`
2. To view the role's maximum session duration, run the following command. Then view the maximum session duration parameter.
 - `aws iam get-role`

To view a role's maximum session duration setting (AWS API)

1. If you don't know the name of the role that you want to assume, call the following operation to list the roles in your account:
 - `ListRoles`
2. To view the role's maximum session duration, run the following operation. Then view the maximum session duration parameter.
 - `GetRole`

Granting a User Permissions to Switch Roles

When you [create a role for cross-account access \(p. 229\)](#), you establish trust from the account that owns the role and the resources (trusting account) to the account that contains the users (trusted account). To do this, you specify the trusted account number as the `Principal` in the role's trust policy.

That allows *potentially* any user in the trusted account to assume the role. To complete the configuration, the administrator of the trusted account must give specific groups or users in that account permission to switch to the role.

To grant a user permission to switch to a role, you create a new policy for the user or edit an existing policy to add the required elements. You can then send the users a link that takes the user to the **Switch Role** page with all the details already filled in. Alternatively, you can provide the user with the account ID number or account alias that contains the role and the role name. The user then goes to the **Switch Role** page and adds the details manually. For details on how a user switches roles, see [Switching to a Role \(Console\) \(p. 260\)](#).

Note that you can switch roles only when you sign in as an IAM user. You cannot switch roles when you sign in as the AWS account root user.

Important

You cannot switch roles in the AWS Management Console to a role that requires an [ExternalId \(p. 233\)](#) value. You can switch to such a role only by calling the AssumeRole API that supports the `ExternalId` parameter.

Notes

- This topic discusses policies for a *user*, because we are ultimately granting permissions to a user to accomplish a task. However, it is [best practice not to grant permissions directly to an individual user \(p. 61\)](#). For easier management, we recommend assigning policies and granting permissions to IAM groups and then making the users members of the appropriate groups.
- When you switch roles in the AWS Management Console, the console always uses your original credentials to authorize the switch. This applies whether you sign in as an IAM user, as a SAML-federated role, or as a web-identity federated role. For example, if you switch to RoleA, it uses your original user or federated role credentials to determine if you are allowed to assume RoleA. If you then try to switch to RoleB *while you are using RoleA*, your **original** user or federated role credentials are used to authorize your attempt, not the credentials for RoleA.

Topics

- [Creating or Editing the Policy \(p. 256\)](#)
- [Providing Information to the User \(p. 257\)](#)

Creating or Editing the Policy

A policy that grants a user permission to assume a role must include a statement with the `Allow` effect on the following:

- The `sts:AssumeRole` action
- The Amazon Resource Name (ARN) of the role in a `Resource` element

This is as shown in the following example. Users that get the policy (either through group membership or directly attached) are allowed to switch to the specified role.

Note

Note that if `Resource` is set to `*`, the user can assume any role in any account that trusts the user's account (the role's trust policy specifies the user's account as `Principal`). As a best practice, we recommend that you follow the [principle of least privilege](#) and specify the complete ARN for only the roles that the user needs.

The following example shows a policy that lets the user assume roles in only one account. In addition, the policy uses a wildcard (*) to specify that the user can switch to a role only if the role name begins with the letters `Test`.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": "sts:AssumeRole",  
            "Resource": "arn:aws:iam::ACCOUNT-ID-WITHOUT-HYPHENS:role/Test*"  
        }  
    ]  
}
```

Note

The permissions that the role grants to the user do not add to the permissions already granted to the user. When a user switches to a role, the user temporarily gives up his or her original permissions in exchange for those granted by the role. When the user exits the role, then the original user permissions are automatically restored. For example, let's say the user's permissions allow working with Amazon EC2 instances, but the role's permissions policy does not grant those permissions. In that case, while using the role, the user cannot work with Amazon EC2 instances in the console. In addition, temporary credentials obtained via AssumeRole do not work with Amazon EC2 instances programmatically.

Providing Information to the User

After you create a role and grant your user permissions to switch to it, you must provide the user with the following:

- The role name
- The account ID number or account alias that contains the role

You can make things easier for your users by sending them a link that is preconfigured with the account ID and role name. You can see the role link on the final page of the **Create Role** wizard or in the **Role Summary** page for any cross-account enabled role.

Note

If you create the role with the AWS CLI , Tools for Windows PowerShell, or the AWS API, then you can create the role with a *path* in addition to a name. If you do so, then you must provide the complete path and role name to your users to type on the **Switch Role** page of the AWS Management Console. For example: division_abc/subdivision_efg/role_xyz.

Important

If you create the role programmatically instead of in the IAM console, then you can add a Path of up to 512 characters in addition to the RoleName. The RoleName can be up to 64 characters long. However, to use a role with the Switch Role feature in the AWS console, the combined Path and RoleName cannot exceed 64 characters.

You can also use the following format to manually construct the link. Substitute your account ID or alias and the role name for the two parameters in the request:

`https://signin.aws.amazon.com/switchrole?
account=YourAccountIDOrAliasHere&roleName=pathIfAny/YourRoleNameHere`

We recommend that you direct your users to the topic [Switching to a Role \(Console\) \(p. 260\)](#) to step them through the process.

Note

For security purposes, you can use AWS CloudTrail to audit role switching. If CloudTrail is turned on for the account, IAM logs actions that are performed with the role's temporary security credentials. For more information, see [CloudTrail Event Reference](#) in the *AWS CloudTrail User Guide*.

Granting a User Permissions to Pass a Role to an AWS Service

To configure many AWS services, you must *pass* an IAM role to the service. This allows the service to later assume the role and perform actions on your behalf. You only have to pass the role to the service once during set-up, and not every time that the service assumes the role. For example, assume that you have an application running on an Amazon EC2 instance. That application requires temporary credentials for authentication, and permissions to authorize the application to perform actions in AWS. When you set up the application, you must pass a role to EC2 to use with the instance that provides those credentials. You define the permissions for the applications running on the instance by attaching an IAM policy to the role. The application assumes the role every time it needs to perform the actions that are allowed by the role.

To pass a role (and its permissions) to an AWS service, a user must have permissions to *pass the role* to the service. This helps administrators ensure that only approved users can configure a service with a role that grants permissions. To allow a user to pass a role to an AWS service, you must grant the `PassRole` permission to the user's IAM user, role, or group.

Note

You cannot limit permissions to pass a role based on tags attached to that role using the `ResourceTag/key-name` condition key. For more information, see [Controlling Access to AWS Resources \(p. 392\)](#).

When you create a service-linked role, you must also have permission to pass that role to the service. Some services automatically create a service-linked role in your account when you perform an action in that service. For example, Amazon EC2 Auto Scaling creates the `AWSServiceRoleForAutoScaling` service-linked role for you the first time that you create an Auto Scaling group. If you try to create an Auto Scaling group without the `PassRole` permission, you receive an error. To learn which services support service-linked roles, see [AWS Services That Work with IAM \(p. 586\)](#). To learn which services automatically create a service-linked role when you perform an action in that service, choose the `Yes` link and view the service-linked role documentation for the service.

A user can pass a role ARN as a parameter in any API operation that uses the role to assign permissions to the service. The service then checks whether that user has the `iam:PassRole` permission. To limit the user to passing only approved roles, you can filter the `iam:PassRole` permission with the `Resources` element of the IAM policy statement.

Example 1

Imagine that you want to grant a user the ability to pass any of an approved set of roles to the Amazon EC2 service upon launching an instance. You need three elements:

- An IAM *permissions policy* attached to the role that determines what the role can do. Scope permissions to only the actions that the role must perform, and to only the resources that the role needs for those actions. You can use AWS managed or customer-created IAM permissions policy.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": [ "A list of the permissions the role is allowed to use" ],  
            "Resource": [ "A list of the resources the role is allowed to access" ]  
        }  
    ]  
}
```

- A *trust policy* for the role that allows the service to assume the role. For example, you could attach the following trust policy to the role with the `UpdateAssumeRolePolicy` action. This trust policy allows Amazon EC2 to use the role and the permissions attached to the role.

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "TrustPolicyStatementThatAllowsEC2ServiceToAssumeTheAttachedRole",
            "Effect": "Allow",
            "Principal": { "Service": "ec2.amazonaws.com" },
            "Action": "sts:AssumeRole"
        }
    ]
```

- An IAM *permissions policy* attached to the IAM user that allows the user to pass only those roles that are approved. `iam:PassRole` usually is accompanied by `iam:GetRole` so that the user can get the details of the role to be passed. In this example, the user can pass only roles that exist in the specified account with names that begin with `EC2-roles-for-XYZ-`:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iam:GetRole",
                "iam:PassRole"
            ],
            "Resource": "arn:aws:iam:<account-id>:role/EC2-roles-for-XYZ-*"
        }
    ]
}
```

Now the user can start an Amazon EC2 instance with an assigned role. Applications running on the instance can access temporary credentials for the role through the instance profile metadata. The permission policies attached to the role determine what the instance can do.

Example 2

Amazon Relational Database Service (Amazon RDS) supports a feature called Enhanced Monitoring. This feature enables Amazon RDS to monitor a database instance using an agent. It also allows Amazon RDS to log metrics to Amazon CloudWatch Logs. To enable this feature, you must create a service role to give Amazon RDS permissions to monitor and write metrics to your logs.

To create a role for Amazon RDS Enhanced Monitoring

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. Choose **Roles**, and then choose **Create role**.
3. Choose the **AWS Service** role type, and then choose the **Amazon RDS Role for Enhanced Monitoring** service. Then choose **Next: Permissions**.
4. Choose the **AmazonRDSEnhancedMonitoringRole**, permissions policy.
5. Choose **Next: Tags**.
6. (Optional) Add metadata to the user by attaching tags as key-value pairs. For more information about using tags in IAM, see [Tagging IAM Users and Roles \(p. 295\)](#).
7. Choose **Next: Review**.
8. For **Role name**, type a role name that helps you identify the purpose of this role. Role names must be unique within your AWS account. They are not distinguished by case. For example, you cannot create roles named both **PRODROLE** and **prodrole**. Because various entities might reference the role, you cannot edit the name of the role after it has been created.
9. (Optional) For **Role description**, type a description for the new role.
10. Review the role and then choose **Create role**.

The role automatically gets a trust policy that grants the `monitoring.rds.amazonaws.com` service permissions to assume the role. After it does, Amazon RDS can perform all of the actions that the `AmazonRDSEnhancedMonitoringRole` policy allows.

The user that you want to enable Enhanced Monitoring needs a policy that includes a statement that allows the user to pass the role, like the following. Use your account number and replace the role name with the name you provided in step 3:

```
{  
    "Sid": "PolicyStatementToAllowUserToPassOneSpecificRole",  
    "Effect": "Allow",  
    "Action": [ "iam:PassRole" ],  
    "Resource": "arn:aws:iam:::role/RDS-Monitoring-Role"  
}
```

You can combine this statement with statements in another policy or put it in its own policy. To instead specify that the user can pass any role that begins with `RDS-`, you can replace the role name in the resource ARN with a wildcard, for example:

```
"Resource": "arn:aws:iam:::role/RDS-*"
```

Switching to a Role (Console)

A *role* specifies a set of permissions that you can use to access AWS resources that you need. In that sense, it is similar to a [user in AWS Identity and Access Management \(IAM\)](#). When you sign in as a user, you get a specific set of permissions. However, you don't sign in to a role, but once signed in you can switch to a role. This temporarily sets aside your original user permissions and instead gives you the permissions assigned to the role. The role can be in your own account or any other AWS account. For more information about roles, their benefits, and how to create them, see [IAM Roles \(p. 175\)](#), and [Creating IAM Roles \(p. 228\)](#).

Important

The permissions of your IAM user and any roles that you switch to are not cumulative. Only one set of permissions is active at a time. When you switch to a role, you temporarily give up your user permissions and work with the permissions that are assigned to the role. When you exit the role, your user permissions are automatically restored.

By default, your AWS Management Console session lasts for one hour.

When you switch roles in the AWS Management Console, the console always uses your original credentials to authorize the switch. This applies whether you sign in as an IAM user, as a SAML-federated role, or as a web-identity federated role. For example, if you switch to RoleA, IAM uses your original user or federated role credentials to determine if you are allowed to assume RoleA. If you then switch to RoleB *while you are using RoleA*, IAM still uses your **original** user or federated role credentials to authorize the switch, not the credentials for RoleA.

This section describes how to use the IAM console to switch to a role:

- You can only switch roles when you sign in as an IAM user. You cannot switch roles if you sign in as the AWS account root user.
- If your administrator provides you with a link, choose the link and then skip to step [Step 5](#) in the following procedure. The link takes you to the appropriate webpage and fills in the account ID (or alias) and the role name for you.
- You can manually construct the link and then skip to step [Step 5](#) in the following procedure. To construct your link, use the following format:

`https://signin.aws.amazon.com/switchrole?
account=account_id_number&roleName=role_name&displayName=text_to_display`

Where you replace the following text:

- **account_id_number**—The 12-digit account identifier provided to you by your administrator. Alternatively, your administrator might create an account alias so that the URL includes your account name instead of an account ID. For more information, see [Your AWS Account ID and Its Alias \(p. 76\)](#).
- **role_name**—The name of the role that you want to assume. You can get this from the end of the role's ARN. For example, provide the TestRole role name from the following role ARN:
`arn:aws:iam::403299380220:role/TestRole`.
- (Optional) **text_to_display**—The text that you want to appear on the navigation bar in place of your user name when this role is active.
- You can manually switch roles using the information your administrator provides by using the procedures below.

To troubleshoot common issues that you might encounter when you assume a role, see [I Can't Assume a Role \(p. 563\)](#).

To switch to a role (console)

1. Sign in to the AWS Management Console as an IAM user and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the IAM console, choose your user name on the navigation bar in the upper right. It typically looks like this: **username@account_ID_number_or_alias**.
3. Choose **Switch Role**. If this is the first time choosing this option, a page appears with more information. After reading it, choose **Switch Role**. If you clear your browser cookies, this page can appear again.
4. On the **Switch Role** page, type the account ID number or the account alias and the name of the role that was provided by your administrator.

Note

If your administrator created the role with a path, such as `division_abc/subdivision_efg/roleToDoX`, then you must type that complete path and name in the **Role** box. If you type only the role name, or if the combined Path and RoleName exceed 64 characters, the role switch fails. This is a limit of the browser cookies that store the role name. If this happens, contact your administrator and ask them to reduce the size of the path and role name.

5. (Optional) Type text that you want to appear on the navigation bar in place of your user name when this role is active. A name is suggested, based on the account and role information, but you can change it to whatever has meaning for you. You can also select a color to highlight the display name. The name and color can help remind you when this role is active, which changes your permissions. For example, for a role that gives you access to the test environment, you might specify a **Display Name of Test** and select the green **Color**. For the role that gives you access to production, you might specify a **Display Name of Production** and select red as the **Color**.
6. Choose **Switch Role**. The display name and color replace your user name on the navigation bar, and you can start using the permissions that the role grants you.

Tip

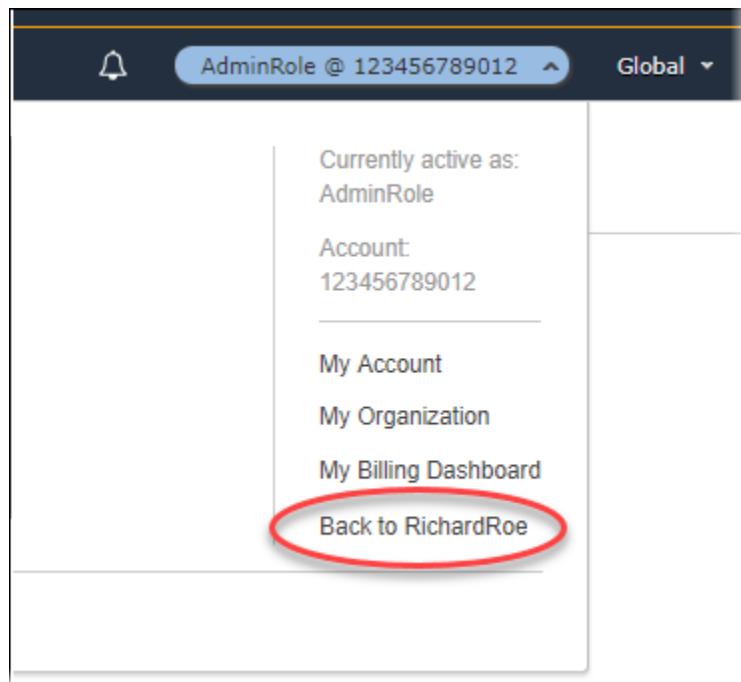
The last several roles that you used appear on the menu. The next time you need to switch to one of those roles, you can simply choose the role you want. You only need to type the account and role information manually if the role is not displayed on the Identity menu.

To stop using a role (console)

1. In the IAM console, choose your role's **Display Name** on the navigation bar in the upper right. It typically looks like this: **rolename@account_ID_number_or_alias**.

2. Choose **Back to *username***. The role and its permissions are deactivated, and the permissions associated with your IAM user and groups are automatically restored.

For example, assume you are signed in to account number 123456789012 using the user name RichardRoe. After you use the AdminRole role, you want to stop using the role and return to your original permissions. To stop using a role, choose **AdminRole @ 123456789012**, and then choose **Back to RichardRoe**.



Switching to an IAM Role (AWS CLI)

A *role* specifies a set of permissions that you can use to access AWS resources that you need. In that sense, it is similar to a [user in AWS Identity and Access Management \(IAM\)](#). When you sign in as a user, you get a specific set of permissions. However, you don't sign in to a role, but after signing in as a user, you can switch to a role. This temporarily sets aside your original user permissions and instead gives you the permissions assigned to the role. The role can be in your own account or any other AWS account. For more information about roles, their benefits, and how to create and configure them, see [IAM Roles \(p. 175\)](#), and [Creating IAM Roles \(p. 228\)](#). To learn about the different methods that you can use to assume a role, see [Using IAM Roles \(p. 253\)](#).

Important

The permissions of your IAM user and any roles that you assume are not cumulative. Only one set of permissions is active at a time. When you assume a role, you temporarily give up your previous user or role permissions and work with the permissions that are assigned to the role. When you exit the role, your user permissions are automatically restored.

You can use a role to run an AWS CLI command when you are signed in as an IAM user. You can also use a role to run an AWS CLI command when you are signed in as an [externally authenticated user \(p. 184\)](#) ([SAML \(p. 190\)](#) or [OIDC \(p. 185\)](#)) that is already using a role. In addition, you can use a role to run an AWS CLI command from within an Amazon EC2 instance that is attached to a role through its instance profile. You can also use [role chaining \(p. 177\)](#), which is using a role to assume a second role. You cannot assume a role when you are signed in as the AWS account root user.

By default, your role session lasts for one hour. When you assume this role using the `assume-role*` CLI operations, you can specify a value for the `duration-seconds` parameter. This value can range from

900 seconds (15 minutes) up to the maximum session duration setting for the role. To learn how to view the maximum value for your role, see [View the Maximum Session Duration Setting for a Role \(p. 255\)](#).

If you use role chaining, your session duration is limited to a maximum of one hour. If you then use the `duration-seconds` parameter to provide a value greater than one hour, the operation fails.

Example Scenario: Switch to a Production Role

Imagine that you have an IAM user for working in the development environment and you occasionally need to work with the production environment at the command line with the [AWS CLI](#). You already have an access key credential set available to you. This can be the access key pair that is assigned to your standard IAM user. Or, if you signed in as a federated user, it can be the access key pair for the role that was initially assigned to you. If your current permissions grant you the ability to assume a specific IAM role, then you can identify that role in a "profile" in the AWS CLI configuration files. That command is then run with the permissions of the specified IAM role, not the original identity. Note that when you specify that profile in an AWS CLI command, you are using the new role. In this situation, you cannot make use of your original permissions in the development account at the same time. The reason is that only one set of permissions can be in effect at a time.

Note

For security purposes, you can use AWS CloudTrail to audit the use of roles in the account. To identify a role's actions in CloudTrail logs, you can use the role session name. When the AWS CLI assumes a role on a user's behalf as described in this topic, a role session name is automatically created as `AWS-CLI-session-nnnnnnnnnn`. Here `nnnnnnnnnn` is an integer that represents the time in [Unix epoch time](#) (the number of seconds since midnight UTC on January 1, 1970). For more information, see [CloudTrail Event Reference](#) in the [AWS CloudTrail User Guide](#).

To switch to a production role (AWS CLI)

1. If you have never used the AWS CLI, then you must first configure your default CLI profile. Open a command prompt and set up your AWS CLI installation to use the access key from your IAM user or from your federated role. For more information, see [Configuring the AWS Command Line Interface](#) in the [AWS Command Line Interface User Guide](#).

```
$ aws configure
AWS Access Key ID [None]: AKIAIOSFODNN7EXAMPLE
AWS Secret Access Key [None]: wJalrXutnFEMI/K7MDENG/bPxRficyEXAMPLEKEY
Default region name [None]: us-east-2
Default output format [None]: json
```

2. Create a new profile for the role in the `.aws/config` file in Unix or Linux, or the `C:\Users\USERNAME\.aws\config` file in Windows. The following example creates a profile called `prodaccess` that switches to the role `ProductionAccessRole` in the `123456789012` account. You get the role ARN from the account administrator who created the role. When this profile is invoked, the AWS CLI uses the credentials of the `source_profile` to request credentials for the role. Because of that, the identity referenced as the `source_profile` must have `sts:AssumeRole` permissions to the role that is specified in the `role_arn`.

```
[profile prodaccess]
role_arn = arn:aws:iam::123456789012:role/ProductionAccessRole
source_profile = default
```

3. After you create the new profile, any AWS CLI command that specifies the parameter `--profile prodaccess` runs under the permissions that are attached to the IAM role `ProductionAccessRole` instead of the default user.

```
$ aws iam list-users --profile prodaccess
```

This command works if the permissions assigned to the `ProductionAccessRole` enable listing the users in the current AWS account.

4. To return to the permissions granted by your original credentials, run commands without the `--profile` parameter. The AWS CLI reverts to using the credentials in your default profile, which you configured in [Step 1](#).

For more information, see [Assuming a Role in the AWS Command Line Interface User Guide](#).

Example Scenario: Allow an Instance Profile Role to Switch to a Role in Another Account

Imagine that you are using two AWS accounts, and you want to allow an application running on an Amazon EC2 instance to run [AWS CLI](#) commands in both accounts. Assume that the EC2 instance exists in account 111111111111. That instance includes the `abcd` instance profile role that allows the application to perform read-only Amazon S3 tasks on the `my-bucket-1` bucket within the same 111111111111 account. However, the application must also be allowed to assume the `efgh` cross-account role to perform tasks in account 222222222222. To do this, the `abcd` EC2 instance profile role must have the following permissions policy:

Account 111111111111 abcd Role Permissions Policy

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "AllowAccountLevelS3Actions",  
            "Effect": "Allow",  
            "Action": [  
                "s3>ListAllMyBuckets",  
                "s3:HeadBucket"  
            ],  
            "Resource": "*"  
        },  
        {  
            "Sid": "AllowListAndReadS3ActionOnMyBucket",  
            "Effect": "Allow",  
            "Action": [  
                "s3:Get*",  
                "s3>List*"  
            ],  
            "Resource": [  
                "arn:aws:s3::::my-bucket-1/*",  
                "arn:aws:s3::::my-bucket-1"  
            ]  
        },  
        {  
            "Sid": "AllowIPToAssumeCrossAccountRole",  
            "Effect": "Allow",  
            "Action": "sts:AssumeRole",  
            "Resource": "arn:aws:iam:222222222222:role/efgh"  
        }  
    ]  
}
```

Assume that the `efgh` cross-account role allows read-only Amazon S3 tasks on the `my-bucket-2` bucket within the same 222222222222 account. To do this, the `efgh` cross-account role must have the following permissions policy:

Account 222222222222 efgh Role Permissions Policy

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAccountLevelS3Actions",
            "Effect": "Allow",
            "Action": [
                "s3>ListAllMyBuckets",
                "s3>HeadBucket"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowListAndReadS3ActionOnMyBucket",
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3>List*"
            ],
            "Resource": [
                "arn:aws:s3:::my-bucket-2/*",
                "arn:aws:s3:::my-bucket-2"
            ]
        }
    ]
}
```

The `efgh` role must allow the `abcd` instance profile role to assume it. To do this, the `efgh` role must have the following trust policy:

Account 222222222222 efgh Role Trust Policy

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "efghTrustPolicy",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {"AWS": "arn:aws:iam::111111111111:role/abcd"}
        }
    ]
}
```

To then run AWS CLI commands in account 222222222222, you must update the CLI configuration file. Identify the `efgh` role as the "profile" and the `abcd` EC2 instance profile role as the "credential source" in the AWS CLI configuration file. Then your CLI commands are run with the permissions of the `efgh` role, not the original `abcd` role.

Note

For security purposes, you can use AWS CloudTrail to audit the use of roles in the account. To identify a role's actions in CloudTrail logs, you can use the role session name. When the AWS CLI assumes a role on a user's behalf as described in this topic, a role session name is automatically created as `AWS-CLI-session-nnnnnnnn`. Here `nnnnnnnn` is an integer that represents the time in [Unix epoch time](#) (the number of seconds since midnight UTC on January 1, 1970). For more information, see [CloudTrail Event Reference](#) in the [AWS CloudTrail User Guide](#).

To allow an EC2 instance profile role to switch to a cross-account role (AWS CLI)

1. You do not have to configure a default CLI profile. Instead, you can load credentials from the EC2 instance profile metadata. Create a new profile for the role in the `.aws/config` file. The

following example creates an `instancecrossaccount` profile that switches to the role `efgh` in the `222222222222` account. When this profile is invoked, the AWS CLI uses the credentials of the EC2 instance profile metadata to request credentials for the role. Because of that, the EC2 instance profile role must have `sts:AssumeRole` permissions to the role specified in the `role_arn`.

```
[profile instancecrossaccount]
role_arn = arn:aws:iam::222222222222:role/efgh
credential_source = Ec2InstanceMetadata
```

2. After you create the new profile, any AWS CLI command that specifies the parameter `--profile instancecrossaccount` runs under the permissions that are attached to the `efgh` role in account `222222222222`.

```
$ aws s3 ls my-bucket-2 --profile instancecrossaccount
```

This command works if the permissions that are assigned to the `efgh` role allow listing the users in the current AWS account.

3. To return to the original EC2 instance profile permissions in account `111111111111`, run the CLI commands without the `--profile` parameter.

For more information, see [Assuming a Role in the AWS Command Line Interface User Guide](#).

Switching to an IAM Role (Tools for Windows PowerShell)

A *role* specifies a set of permissions that you can use to access AWS resources that you need. In that sense, it is similar to a [user in AWS Identity and Access Management \(IAM\)](#). When you sign in as a user, you get a specific set of permissions. However, you don't sign in to a role, but once signed in you can switch to a role. This temporarily sets aside your original user permissions and instead gives you the permissions assigned to the role. The role can be in your own account or any other AWS account. For more information about roles, their benefits, and how to create and configure them, see [IAM Roles \(p. 175\)](#), and [Creating IAM Roles \(p. 228\)](#).

Important

The permissions of your IAM user and any roles that you switch to are not cumulative. Only one set of permissions is active at a time. When you switch to a role, you temporarily give up your user permissions and work with the permissions that are assigned to the role. When you exit the role, your user permissions are automatically restored.

This section describes how to switch roles when you work at the command line with the AWS Tools for Windows PowerShell.

Imagine that you have an account in the development environment and you occasionally need to work with the production environment at the command line using the [Tools for Windows PowerShell](#). You already have one access key credential set available to you. These can be an access key pair assigned to your standard IAM user. Or, if you signed-in as a federated user, they can be the access key pair for the role initially assigned to you. You can use these credentials to run the `Use-STSRole` cmdlet that passes the ARN of a new role as a parameter. The command returns temporary security credentials for the requested role. You can then use those credentials in subsequent PowerShell commands with the role's permissions to access resources in production. While you use the role, you cannot make use of your user permissions in the Development account because only one set of permissions can be in effect at a time.

Note

For security purposes, you can use AWS CloudTrail to audit the use of roles in the account. The cmdlet `Use-STSRole` must include a `-RoleSessionName` parameter with a value between 2 and 64 characters long that can include letters, numbers, and the `=`, `.@-` characters. The

role session name identifies actions in CloudTrail logs that are performed with the temporary security credentials. For more information, see [CloudTrail Event Reference](#) in the *AWS CloudTrail User Guide*.

Note that all access keys and tokens are examples only and cannot be used as shown. Replace with the appropriate values from your live environment.

To switch to a role (Tools for Windows PowerShell)

1. Open a PowerShell command prompt and configure the default profile to use the access key from your current IAM user or from your federated role. If you have previously used the Tools for Windows PowerShell, then this is likely already done. Note that you can switch roles only if you are signed in as an IAM user, not the AWS account root user.

```
PS C:\> Set-AWSCredentials -AccessKey AKIAIOSFODNN7EXAMPLE -SecretKey wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY -StoreAs MyMainUserProfile
PS C:\> Initialize-AWSDefaults -ProfileName MyMainUserProfile -Region us-east-2
```

For more information, see [Using AWS Credentials](#) in the *AWS Tools for Windows PowerShell User Guide*.

2. To retrieve credentials for the new role, run the following command to switch to the `RoleName` role in the 123456789012 account. You get the role ARN from the account administrator who created the role. The command requires that you provide a session name as well. You can choose any text for that. The following command requests the credentials and then captures the `Credentials` property object from the returned results object and stores it in the `$creds` variable.

```
PS C:\> $creds = (Use-STSSRole -RoleArn "arn:aws:iam::123456789012:role/RoleName" -RoleSessionName "MyRoleSessionName").Credentials
```

`$creds` is an object that now contains the `AccessKeyId`, `SecretAccessKey`, and `SessionToken` elements that you need in the following steps. The following sample commands illustrate typical values:

```
PS C:\> $creds.AccessKeyId
AKIAIOSFODNN7EXAMPLE

PS C:\> $creds.SecretAccessKey
wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY

PS C:\> $creds.SessionToken
AQoDYXdzEGcaEXAMPLE2gsYULo+Im5ZEXAMPLEeYjs1M2FUIgIJx9tQqNMBEXAMPLEcvSRyh0FW7jEXAMPLEw+vE/7s1HRp
XviG7b+qYf4nD00EXAMPLEmj4wxS04L/uZEXAMPLEcihzFB5lTYLto9dyBgSDyEXAMPLE9/
g7QRUhZp4bqbEXAMPLENwGPY
Oj59pFA41NKCikVgkREXAMPLEjlzxQ7y52gekeVEXAMPLEDiB9ST3UuysgsKdEXAMPLE1TVastU1A0SKFEXAMPLEiywCC/
C
s8EXAMPLEpZgOs+6hz4AP4KEXAMPLERbASP+4eZScEXAMPLEsnf87eNhyDHq6ikBQ==

PS C:\> $creds.Expiration
Thursday, June 18, 2018 2:28:31 PM
```

3. To use these credentials for any subsequent command, include them with the `-Credentials` parameter. For example, the following command uses the credentials from the role and works only if the role is granted the `iam>ListRoles` permission and can therefore run the `Get-IAMRoles` cmdlet:

```
PS C:\> Get-IAMRoles -Credential $creds
```

4. To return to your original credentials, simply stop using the `-Credentials $Creds` parameter and allow PowerShell to revert to the credentials that are stored in the default profile.

Switching to an IAM Role (AWS API)

A *role* specifies a set of permissions that you can use to access AWS resources. In that sense, it is similar to an [IAM user](#). A principal (person or application) assumes a role to receive temporary permissions to carry out required tasks and interact with AWS resources. The role can be in your own account or any other AWS account. For more information about roles, their benefits, and how to create and configure them, see [IAM Roles \(p. 175\)](#), and [Creating IAM Roles \(p. 228\)](#). To learn about the different methods that you can use to assume a role, see [Using IAM Roles \(p. 253\)](#).

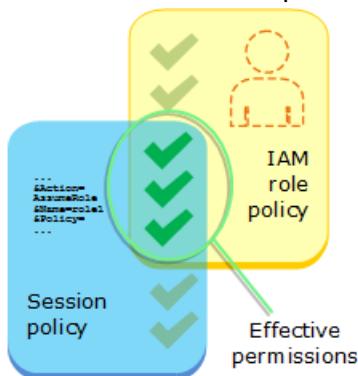
Important

The permissions of your IAM user and any roles that you assume are not cumulative. Only one set of permissions is active at a time. When you assume a role, you temporarily give up your previous user or role permissions and work with the permissions that are assigned to the role. When you exit the role, your original permissions are automatically restored.

To assume a role, an application calls the AWS STS [AssumeRole](#) API operation and passes the ARN of the role to use. The operation creates a new session with temporary credentials. This session has the same permissions as the identity-based policies for that role.

When you call [AssumeRole](#), you can optionally pass inline or managed [session policies \(p. 358\)](#).

Session policies are advanced policies that you pass as a parameter when you programmatically create a temporary credential session for a role or federated user. You can pass a single JSON inline session policy document using the `Policy` parameter. You can use the `PolicyArns` parameter to specify up to 10 managed session policies. The resulting session's permissions are the intersection of the entity's identity-based policies and the session policies. Session policies are useful when you need to give the role's temporary credentials to someone else. They can use the role's temporary credentials in subsequent AWS API calls to access resources in the account that owns the role. You cannot use session policies to grant more permissions than those allowed by the identity-based policy. To learn more about how AWS determines the effective permissions of a role, see [Policy Evaluation Logic \(p. 636\)](#).



You can call [AssumeRole](#) when you are signed in as an IAM user, or as an [externally authenticated user \(p. 184\)](#) ([SAML \(p. 190\)](#) or [OIDC \(p. 185\)](#)) already using a role. You can also use [role chaining \(p. 177\)](#), which is using a role to assume a second role. You cannot assume a role when you are signed in as the AWS account root user.

By default, your role session lasts for one hour. When you assume this role using the AWS STS [AssumeRole*](#) API operations, you can specify a value for the `DurationSeconds` parameter. This value can range from 900 seconds (15 minutes) up to the maximum session duration setting for the role. To learn how to view the maximum value for your role, see [View the Maximum Session Duration Setting for a Role \(p. 255\)](#).

If you use role chaining, your session is limited to a maximum of one hour. If you then use the DurationSeconds parameter to provide a value greater than one hour, the operation fails.

Note

For security purposes, you can use AWS CloudTrail to audit the use of roles in the account. The call to AssumeRole must include a role session name between 2 and 64 characters long that can include letters, numbers, and the =, .@- characters. The role session name is used in CloudTrail logs to identify actions performed by the temporary security credentials. For more information, see [CloudTrail Event Reference](#) in the *AWS CloudTrail User Guide*.

The following example in Python using the Boto3 interface to AWS ([AWS SDK for Python \(Boto\) V3](#)) shows how to call AssumeRole. It also shows how to use the temporary security credentials returned by AssumeRole to list all Amazon S3 buckets in the account that owns the role.

```
import boto3

# The calls to AWS STS AssumeRole must be signed with the access key ID
# and secret access key of an existing IAM user or by using existing temporary
# credentials such as those from another role. (You cannot call AssumeRole
# with the access key for the root account.) The credentials can be in
# environment variables or in a configuration file and will be discovered
# automatically by the boto3.client() function. For more information, see the
# Python SDK documentation:
# http://boto3.readthedocs.io/en/latest/reference/services/sts.html#client

# Create an STS client object that represents a live connection to the
# STS service
sts_client = boto3.client('sts')

# Call the assume_role method of the STSConnection object and pass the role
# ARN and a role session name.
assumed_role_object=sts_client.assume_role(
    RoleArn="arn:aws:iam::account-of-role-to-assume:role/name-of-role",
    RoleSessionName="AssumeRoleSession1"
)

# From the response that contains the assumed role, get the temporary
# credentials that can be used to make subsequent API calls
credentials=assumed_role_object['Credentials']

# Use the temporary credentials that AssumeRole returns to make a
# connection to Amazon S3
s3_resource=boto3.resource(
    's3',
    aws_access_key_id=credentials['AccessKeyId'],
    aws_secret_access_key=credentials['SecretAccessKey'],
    aws_session_token=credentials['SessionToken'],
)

# Use the Amazon S3 resource object that is now configured with the
# credentials to access your S3 buckets.
for bucket in s3_resource.buckets.all():
    print(bucket.name)
```

Using an IAM Role to Grant Permissions to Applications Running on Amazon EC2 Instances

Applications that run on an EC2 instance must include AWS credentials in their AWS API requests. You could have your developers store AWS credentials directly within the EC2 instance and allow applications in that instance to use those credentials. But developers would then have to manage the credentials and ensure that they securely pass the credentials to each instance and update each EC2 instance when it's time to rotate the credentials. That's a lot of additional work.

Instead, you can and should use an IAM role to manage *temporary* credentials for applications that run on an EC2 instance. When you use a role, you don't have to distribute long-term credentials (such as a user name and password or access keys) to an EC2 instance. Instead, the role supplies temporary permissions that applications can use when they make calls to other AWS resources. When you launch an EC2 instance, you specify an IAM role to associate with the instance. Applications that run on the instance can then use the role-supplied temporary credentials to sign API requests.

Using roles to grant permissions to applications that run on EC2 instances requires a bit of extra configuration. An application running on an EC2 instance is abstracted from AWS by the virtualized operating system. Because of this extra separation, an additional step is needed to assign an AWS role and its associated permissions to an EC2 instance and make them available to its applications. This extra step is the creation of an *instance profile* that is attached to the instance. The instance profile contains the role and can provide the role's temporary credentials to an application that runs on the instance. Those temporary credentials can then be used in the application's API calls to access resources and to limit access to only those resources that the role specifies. Note that only one role can be assigned to an EC2 instance at a time, and all applications on the instance share the same role and permissions.

Using roles in this way has several benefits. Because role credentials are temporary and rotated automatically, you don't have to manage credentials, and you don't have to worry about long-term security risks. In addition, if you use a single role for multiple instances, you can make a change to that one role and the change is propagated automatically to all the instances.

Note

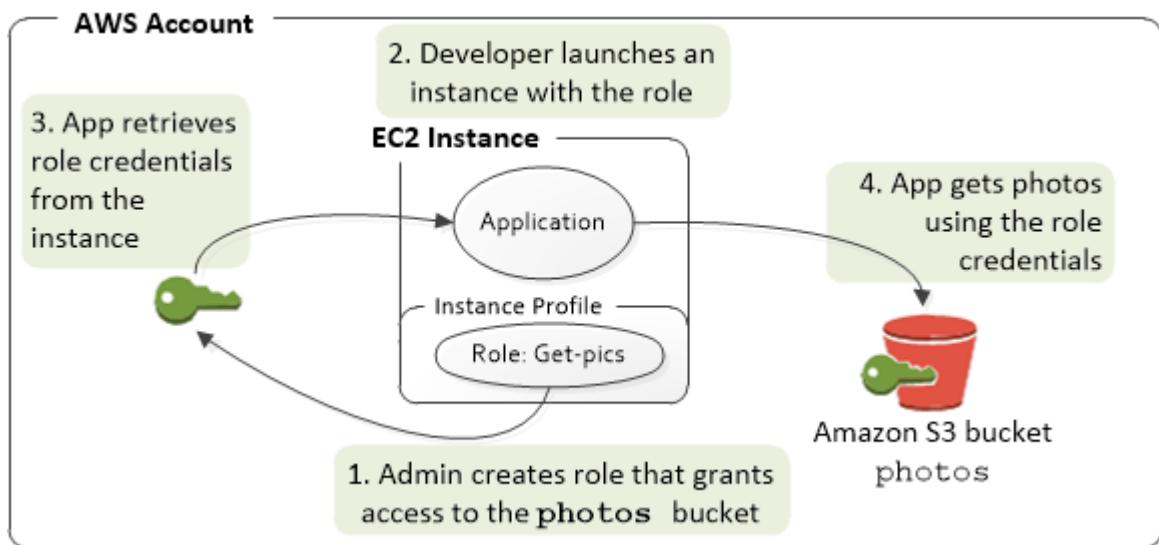
Although a role is usually assigned to an EC2 instance when you launch it, a role can also be attached to an EC2 instance that is already running. To learn how to attach a role to a running instance, see [IAM Roles for Amazon EC2](#).

Topics

- [How Do Roles for EC2 Instances Work? \(p. 270\)](#)
- [Permissions Required for Using Roles with Amazon EC2 \(p. 272\)](#)
- [How Do I Get Started? \(p. 275\)](#)
- [Related Information \(p. 275\)](#)
- [Using Instance Profiles \(p. 276\)](#)

How Do Roles for EC2 Instances Work?

In the following figure, a developer runs an application on an EC2 instance that requires access to the S3 bucket named photos. An administrator creates the Get-pics service role and attaches the role to the EC2 instance. The role includes a permissions policy that grants read-only access to the specified S3 bucket. It also includes a trust policy that allows the EC2 instance to assume the role and retrieve the temporary credentials. When the application runs on the instance, it can use the role's temporary credentials to access the photos bucket. The administrator doesn't have to grant the developer permission to access the photos bucket, and the developer never has to share or manage credentials.



1. The administrator uses IAM to create the **Get-pics** role. In the role's trust policy, the administrator specifies that only EC2 instances can assume the role. In the role's permission policy, the administrator specifies read-only permissions for the photos bucket.
2. A developer launches an EC2 instance and assigns the **Get-pics** role to that instance.

Note

If you use the IAM console, the instance profile is managed for you and is mostly transparent to you. However, if you use the AWS CLI or API to create and manage the role and EC2 instance, then you must create the instance profile and assign the role to it as separate steps. Then, when you launch the instance, you must specify the instance profile name instead of the role name.

3. When the application runs, it obtains temporary security credentials from Amazon EC2 [instance metadata](#), as described in [Retrieving Security Credentials from Instance Metadata](#). These are [temporary security credentials](#) (p. 307) that represent the role and are valid for a limited period of time.

With some [AWS SDKs](#), the developer can use a provider that manages the temporary security credentials transparently. (The documentation for individual AWS SDKs describes the features supported by that SDK for managing credentials.)

Alternatively, the application can get the temporary credentials directly from the instance metadata of the EC2 instance. Credentials and related values are available from the `iam/security-credentials/role-name` category (in this case, `iam/security-credentials/Get-pics`) of the metadata. If the application gets the credentials from the instance metadata, it can cache the credentials.

4. Using the retrieved temporary credentials, the application accesses the photo bucket. Because of the policy attached to the **Get-pics** role, the application has read-only permissions.

The temporary security credentials that are available on the instance are automatically rotated before they expire so that a valid set is always available. The application just needs to make sure that it gets a new set of credentials from the instance metadata before the current ones expire. If the AWS SDK manages credentials, the application doesn't need to include additional logic to refresh the credentials. However, if the application gets temporary security credentials from the instance metadata and has cached them, it should get a refreshed set of credentials every hour, or at least 15 minutes before the current set expires. The expiration time is included in the information that is returned in the `iam/security-credentials/role-name` category.

Permissions Required for Using Roles with Amazon EC2

To launch an instance with a role, the developer must have permission to launch EC2 instances and permission to pass IAM roles.

The following sample policy allows users to use the AWS Management Console to launch an instance with a role. The policy includes wildcards (*) to allow a user to pass any role and to perform all Amazon EC2 actions. The `ListInstanceProfiles` action allows users to view all of the roles that are available in the AWS account.

Example policy that grants a user permission to use the Amazon EC2 console to launch an instance with any role

```
{  
    "Version": "2012-10-17",  
    "Statement": [{  
        "Effect": "Allow",  
        "Action": [  
            "iam:PassRole",  
            "iam>ListInstanceProfiles",  
            "ec2:*"  
        ],  
        "Resource": "*"  
    }]  
}
```

Restricting Which Roles Can Be Passed to EC2 Instances (Using PassRole)

You can use the `PassRole` permission to restrict which role a user can pass to an EC2 instance when the user launches the instance. This helps prevent the user from running applications that have more permissions than the user has been granted—that is, from being able to obtain elevated privileges. For example, imagine that user Alice has permissions only to launch EC2 instances and to work with Amazon S3 buckets, but the role she passes to an EC2 instance has permissions to work with IAM and Amazon DynamoDB. In that case, Alice might be able to launch the instance, log into it, get temporary security credentials, and then perform IAM or DynamoDB actions that she's not authorized for.

To restrict which roles a user can pass to an EC2 instance, you create a policy that allows the `PassRole` action. You then attach the policy to the user (or to an IAM group that the user belongs to) who will launch EC2 instances. In the `Resource` element of the policy, you list the role or roles that the user is allowed to pass to EC2 instances. When the user launches an instance and associates a role with it, Amazon EC2 checks whether the user is allowed to pass that role. Of course, you should also ensure that the role that the user can pass does not include more permissions than the user is supposed to have.

Note

`PassRole` is not an API action in the same way that `RunInstances` or `ListInstanceProfiles` is. Instead, it's a permission that AWS checks whenever a role ARN is passed as a parameter to an API (or the console does this on the user's behalf). It helps an administrator to control which roles can be passed by which users. In this case, it ensures that the user is allowed to attach a specific role to an Amazon EC2 instance.

Example policy that grants a user permission to launch an EC2 instance with a specific role

The following sample policy allows users to use the Amazon EC2 API to launch an instance with a role. The `Resource` element specifies the Amazon Resource Name (ARN) of a role. By specifying the ARN, the policy grants the user the permission to pass only the `Get-pics` role. If the user tries to specify a different role when launching an instance, the action fails. The user does have permissions to run any instance, regardless of whether they pass a role.

```
{
```

```

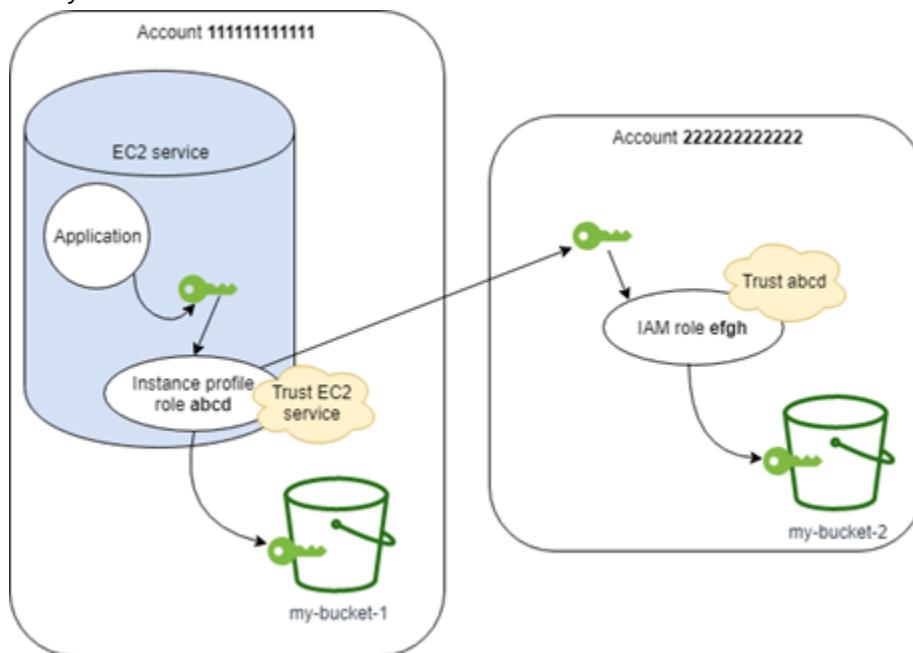
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:RunInstances",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::ACCOUNT-ID-WITHOUT-HYPHENS:role/Get-pics"
        }
    ]
}

```

Allowing an Instance Profile Role to Switch to a Role in Another Account

You can allow an application running on an Amazon EC2 instance to run commands in another account. To do this, you must allow the EC2 instance role in the first account to switch to a role in the second account.

Imagine that you are using two AWS accounts and you want to allow an application running on an Amazon EC2 instance to run [AWS CLI](#) commands in both accounts. Assume that the EC2 instance exists in account 111111111111. That instance includes the abcd instance profile role that allows the application to perform read-only Amazon S3 tasks on the my-bucket-1 bucket within the same 111111111111 account. However, the application must also be allowed to assume the efg role cross-account role to access the my-bucket-2 Amazon S3 bucket in account 222222222222.



The abcd EC2 instance profile role must have the following permissions policy to allow the application to access the my-bucket-1 Amazon S3 bucket:

Account 111111111111 abcd Role Permissions Policy

```

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

```

```

    "Sid": "AllowAccountLevelS3Actions",
    "Effect": "Allow",
    "Action": [
        "s3>ListAllMyBuckets",
        "s3>HeadBucket"
    ],
    "Resource": "*"
},
{
    "Sid": "AllowListAndReadS3ActionOnMyBucket",
    "Effect": "Allow",
    "Action": [
        "s3:Get*",
        "s3>List*"
    ],
    "Resource": [
        "arn:aws:s3:::my-bucket-1/*",
        "arn:aws:s3:::my-bucket-1"
    ]
},
{
    "Sid": "AllowIPToAssumeCrossAccountRole",
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam:222222222222:role/efgh"
}
]
}

```

The abcd role must trust the Amazon EC2 service to assume the role. To do this, the abcd role must have the following trust policy:

Account 111111111111 abcd Role Trust Policy

```

{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "abcdTrustPolicy",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {"Service": "ec2.amazonaws.com"}
        }
    ]
}

```

Assume that the efgh cross-account role allows read-only Amazon S3 tasks on the my-bucket-2 bucket within the same 22222222222 account. To do this, the efgh cross-account role must have the following permissions policy:

Account 222222222222 efgh Role Permissions Policy

```

{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAccountLevelS3Actions",
            "Effect": "Allow",
            "Action": [
                "s3>ListAllMyBuckets",
                "s3>HeadBucket"
            ],
            "Resource": "*"
        }
    ]
}

```

```
        "Resource": "*"
    },
{
    "Sid": "AllowListAndReadS3ActionOnMyBucket",
    "Effect": "Allow",
    "Action": [
        "s3:Get*",
        "s3>List*"
    ],
    "Resource": [
        "arn:aws:s3:::my-bucket-2/*",
        "arn:aws:s3:::my-bucket-2"
    ]
}
}
```

The `efgh` role must trust the `abcd` instance profile role to assume it. To do this, the `efgh` role must have the following trust policy:

Account 222222222222 efg Role Trust Policy

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "efghTrustPolicy",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {"AWS": "arn:aws:iam::111111111111:role/abcd"}
        }
    ]
}
```

How Do I Get Started?

To understand how roles work with EC2 instances, you need to use the IAM console to create a role, launch an EC2 instance that uses that role, and then examine the running instance. You can examine the [instance metadata](#) to see how the role's temporary credentials are made available to an instance. You can also see how an application that runs on an instance can use the role. Use the following resources to learn more.

- SDK walkthroughs. The AWS SDK documentation includes walkthroughs that show an application running on an EC2 instance that uses temporary credentials for roles to read an Amazon S3 bucket. Each of the following walkthroughs presents similar steps with a different programming language:
 - [Configure IAM Roles for Amazon EC2 with the SDK for Java](#) in the *AWS SDK for Java Developer Guide*
 - [Launch an EC2 Instance using the SDK for .NET](#) in the *AWS SDK for .NET Developer Guide*
 - [Creating an Amazon EC2 Instance with the SDK for Ruby](#) in the *AWS SDK for Ruby Developer Guide*

Related Information

For more information about creating roles or roles for EC2 instances, see the following information:

- For more information about [using IAM roles with Amazon EC2 instances](#), go to the *Amazon EC2 User Guide for Linux Instances*.
- To create a role, see [Creating IAM Roles](#) (p. 228)

- For more information about using temporary security credentials, see [Temporary Security Credentials \(p. 307\)](#).
- If you work with the IAM API or CLI, you must create and manage IAM instance profiles. For more information about instance profiles, see [Using Instance Profiles \(p. 276\)](#).
- For more information about temporary security credentials for roles in the instance metadata, see [Retrieving Security Credentials from Instance Metadata](#) in the *Amazon EC2 User Guide for Linux Instances*.

Using Instance Profiles

An instance profile is a container for an IAM role that you can use to pass role information to an EC2 instance when the instance starts.

Managing Instance Profiles (Console)

If you use the AWS Management Console to create a role for Amazon EC2, the console automatically creates an instance profile and gives it the same name as the role. When you then use the Amazon EC2 console to launch an instance with an IAM role, you can select a role to associate with the instance. In the console, the list that's displayed is actually a list of instance profile names. The console does not create an instance profile for a role that is not associated with Amazon EC2.

Managing Instance Profiles (AWS CLI or AWS API)

If you manage your roles from the AWS CLI or the AWS API, you create roles and instance profiles as separate actions. Because roles and instance profiles can have different names, you must know the names of your instance profiles as well as the names of roles they contain. That way you can choose the correct instance profile when you launch an EC2 instance.

Note

An instance profile can contain only one IAM role, although a role can be included in multiple instance profiles. This limit of one role per instance profile cannot be increased. You can remove the existing role and then add a different role to an instance profile. You must then wait for the change to appear across all of AWS because of [eventual consistency](#). To force the change, you must [disassociate the instance profile](#) and then [associate the instance profile](#), or you can stop your instance and then restart it.

Managing Instance Profiles (AWS CLI)

You can use the following AWS CLI commands to work with instance profiles in an AWS account.

- Create an instance profile: `aws iam create-instance-profile`
- Add a role to an instance profile: `aws iam add-role-to-instance-profile`
- List instance profiles: `aws iam list-instance-profiles`, `aws iam list-instance-profiles-for-role`
- Get information about an instance profile: `aws iam get-instance-profile`
- Remove a role from an instance profile: `aws iam remove-role-from-instance-profile`
- Delete an instance profile: `aws iam delete-instance-profile`

You can also attach a role to an already running EC2 instance by using the following commands. For more information, see [IAM Roles for Amazon EC2](#).

- Attach an instance profile with a role to a stopped or running EC2 instance: `aws ec2 associate-iam-instance-profile`
- Get information about an instance profile attached to an EC2 instance: `aws ec2 describe-iam-instance-profile-associations`

- Detach an instance profile with a role from a stopped or running EC2 instance: [aws ec2 disassociate-iam-instance-profile](#)

Managing Instance Profiles (AWS API)

You can call the following AWS API operations to work with instance profiles in an AWS account.

- Create an instance profile: [CreateInstanceProfile](#)
- Add a role to an instance profile: [AddRoleToInstanceProfile](#)
- List instance profiles: [ListInstanceProfiles](#), [ListInstanceProfilesForRole](#)
- Get information about an instance profile: [GetInstanceProfile](#)
- Remove a role from an instance profile: [RemoveRoleFromInstanceProfile](#)
- Delete an instance profile: [DeleteInstanceProfile](#)

You can also attach a role to an already running EC2 instance by calling the following operations. For more information, see [IAM Roles for Amazon EC2](#).

- Attach an instance profile with a role to a stopped or running EC2 instance: [AssociateIamInstanceProfile](#)
- Get information about an instance profile attached to an EC2 instance: [DescribeIamInstanceProfileAssociations](#)
- Detach an instance profile with a role from a stopped or running EC2 instance: [DisassociateIamInstanceProfile](#)

Revoking IAM Role Temporary Security Credentials

Warning

If you follow the steps on this page, all users with current sessions created by assuming the role are denied access to all AWS actions and resources. This can result in users losing unsaved work.

When you enable users to access the AWS Management Console with a long session duration time (such as 12 hours), their temporary credentials do not expire as quickly. If users inadvertently expose their credentials to an unauthorized third party, that party has access for the duration of the session. However, you can immediately revoke all permissions to the role's credentials issued before a certain point in time if you need to. All temporary credentials for that role issued before the specified time become invalid. This forces all users to reauthenticate and request new credentials.

Note

You cannot revoke the session for a [service-linked role \(p. 176\)](#).

When you revoke permissions for a role using the procedure in this topic, AWS attaches a new inline policy to the role that denies all permissions to all actions. It includes a condition that applies the restrictions only if the user assumed the role *before* the point in time when you revoke the permissions. If the user assumes the role *after* you revoked the permissions, then the deny policy does not apply to that user.

Important

This deny policy applies to all users of the specified role, not just those with longer duration console sessions.

Minimum Permissions to Revoke Session Permissions from a Role

To successfully revoke session permissions from a role, you must have the `PutRolePolicy` permission for the role. This allows you to attach the `AWSRevokeOlderSessions` inline policy to the role.

Revoking Session Permissions

You can revoke the session permissions from a role.

To immediately deny all permissions to any current user of role credentials

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane of the **IAM Dashboard**, choose **Roles**, and then choose the name (not the check box) of the role whose permissions you want to revoke.
3. On the **Summary** page for the selected role, choose the **Revoke sessions** tab.
4. On the **Revoke sessions** tab, choose **Revoke active sessions**.
5. AWS asks you to confirm the action. Choose **Revoke active sessions** on the dialog box.

IAM immediately attaches a policy named `AWSRevokeOlderSessions` to the role. The policy denies all access to users who assumed the role before the moment you chose **Revoke active sessions**. Any user who assumes the role *after* you chose **Revoke active sessions** is **not** affected.

When you apply a new policy to a user or a resource, it may take a few minutes for policy updates to take effect.

Note

Don't worry about remembering to delete the policy. Any user who assumes the role *after* you revoked sessions is not affected by the policy. If you choose to **Revoke Sessions** again later, then the date/time stamp in the policy is refreshed and it again denies all permissions to any user who assumed the role before the new specified time.

Valid users whose sessions are revoked in this way must acquire temporary credentials for a new session to continue working. Note that the AWS CLI caches credentials until they expire. To force the CLI to delete and refresh cached credentials that are no longer valid, run one of the following commands:

Linux, macOS, or Unix

```
$ rm -r ~/.aws/cli/cache
```

Windows

```
C:\> del /s /q %UserProfile%\.aws\cli\cache
```

For more information, see [Disabling Permissions for Temporary Security Credentials \(p. 329\)](#).

Managing IAM Roles

Occasionally you need to modify or delete the roles that you have created. To change a role, you can do any of the following:

- Modify the policies that are associated with the role
- Change who can access the role
- Edit the permissions that the role grants to users
- Change the maximum session duration setting for roles that are assumed using the AWS CLI or API

You can also delete roles that are no longer needed. You can manage your roles from the AWS Management Console, the AWS CLI, and the API.

Topics

- [Modifying a Role \(p. 279\)](#)
- [Deleting Roles or Instance Profiles \(p. 289\)](#)

Modifying a Role

You can use the AWS Management Console, the AWS CLI, or the IAM API to make changes to a role.

Topics

- [View Role Access \(p. 279\)](#)
- [Modifying a Role \(Console\) \(p. 279\)](#)
- [Modifying a Role \(AWS CLI\) \(p. 282\)](#)
- [Modifying a Role \(AWS API\) \(p. 286\)](#)

View Role Access

Before you change the permissions for a role, you should review its recent service-level activity. This is important because you don't want to remove access from a principal (person or application) who is using it. For more information about viewing service last accessed data, see [Refining Permissions Using Service Last Accessed Data \(p. 475\)](#).

Modifying a Role (Console)

You can use the AWS Management Console to modify a role. To change the set of tags on a role, see [Managing Tags on IAM Entities \(Console\) \(p. 298\)](#).

Topics

- [Modifying a Role Trust Policy \(Console\) \(p. 279\)](#)
- [Modifying a Role Permissions Policy \(Console\) \(p. 281\)](#)
- [Modifying a Role Description \(Console\) \(p. 281\)](#)
- [Modifying a Role Maximum Session Duration \(Console\) \(p. 281\)](#)
- [Modifying a Role Permissions Boundary \(Console\) \(p. 282\)](#)

Modifying a Role Trust Policy (Console)

To change who can assume a role, you must modify the role's trust policy. You cannot modify the trust policy for a [service-linked role \(p. 176\)](#).

Note

If a user is listed as the principal in a role's trust policy but cannot assume the role, check the user's [permissions boundary \(p. 370\)](#). If a permissions boundary is set for the user, then it must allow the `sts:AssumeRole` action.

To modify a role trust policy (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane of the IAM console, choose **Roles**.
3. In the list of roles in your account, choose the name of the role that you want to modify.
4. Choose the **Trust relationships** tab, and then choose **Edit trust relationship**.

5. Edit the trust policy as needed. To add additional principals that can assume the role, specify them in the `Principal` element. For example, the following policy snippet shows how to reference two AWS accounts in the `Principal` element:

```
"Principal": {
    "AWS": [
        "arn:aws:iam::111122223333:root",
        "arn:aws:iam::444455556666:root"
    ]
},
```

If you specify a principal in another account, adding an account to the trust policy of a role is only half of establishing the cross-account trust relationship. By default, no users in the trusted accounts can assume the role. The administrator for the newly trusted account must grant the users the permission to assume the role. To do that, the administrator must create or edit a policy that is attached to the user to allow the user access to the `sts:AssumeRole` action. For more information, see the following procedure or [Granting a User Permissions to Switch Roles \(p. 255\)](#).

The following policy snippet shows how to reference two AWS services in the `Principal` element:

```
"Principal": {
    "Service": [
        "opsworks.amazonaws.com",
        "ec2.amazonaws.com"
    ]
},
```

6. When you are finished editing your trust policy, choose **Update Trust Policy** to save your changes.

For more information about policy structure and syntax, see [Policies and Permissions \(p. 356\)](#) and the [IAM JSON Policy Elements Reference \(p. 598\)](#).

To allow users in a trusted external account to use the role (console)

For more information and detail about this procedure, see [Granting a User Permissions to Switch Roles \(p. 255\)](#).

1. Sign in to the trusted external AWS account.
2. Decide whether to attach the permissions to a user or to a group. In the navigation pane of the IAM console, choose **Users** or **Groups** accordingly.
3. Choose the name of the user or group to which you want to grant access, and then choose the **Permissions** tab.
4. Do one of the following:
 - To edit a customer managed policy, choose the name of the policy, choose **Edit policy**, and then choose the **JSON** tab. You cannot edit an AWS managed policy. AWS managed policies appear with the AWS icon (). For more information about the difference between AWS managed policies and customer managed policies, see [Managed Policies and Inline Policies \(p. 364\)](#).
 - To edit an inline policy, choose the arrow next to the name of the policy and choose **Edit policy**.
5. In the policy editor, add a new `Statement` element that specifies the following:

```
{
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::ACCOUNT-ID:role/ROLE-NAME"
```

}

Replace the ARN in the statement with the ARN of the role that the user can assume.

6. Follow the prompts on screen to finish editing the policy.

Modifying a Role Permissions Policy (Console)

To change the permissions allowed by the role, modify the role's permissions policy (or policies). You cannot modify the permissions policy for a [service-linked role \(p. 176\)](#) in IAM. You might be able to modify the permissions policy within the service that depends on the role. To check whether a service supports this feature, see [AWS Services That Work with IAM \(p. 586\)](#) 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.

To change the permissions allowed by a role (console)

1. Open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane of the IAM console, choose **Roles**.
3. Choose the name of the role that you want to modify, and then choose the **Permissions** tab.
4. Do one of the following:
 - To edit an existing customer managed policy, choose the name of the policy and then choose **Edit policy**.

Note

You cannot edit an AWS managed policy. AWS managed policy appear with the AWS icon



(). For more information about the difference between AWS managed policies and customer managed policies, see [Managed Policies and Inline Policies \(p. 364\)](#).

- To attach an existing managed policy to the role, choose **Add permissions**.
- To edit an existing inline policy, choose the arrow next to the name of the policy and choose **Edit Policy**.
- To embed a new inline policy, choose **Add inline policy**.

Modifying a Role Description (Console)

To change the description of the role, modify the description text.

To change the description of a role (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane of the IAM console, choose **Roles**.
3. Choose the name of the role to modify.
4. Next to **Role description** and on the far right, choose **Edit**.
5. Type a new description in the box and choose **Save**.

Modifying a Role Maximum Session Duration (Console)

To specify the maximum session duration setting for roles that are assumed using the AWS CLI or API, modify the maximum session duration setting's value. This setting can have a value from 1 hour to 12 hours. If you do not specify a value, the default maximum of 1 hour is applied. This setting does not limit sessions assumed by AWS services.

Note

Anyone who assumes the role from the AWS CLI or API can use the `duration-seconds` CLI parameter or the `DurationSeconds` API parameter to request a longer session. The `MaxSessionDuration` setting determines the maximum duration of the role session that can be requested using the `DurationSeconds` parameter. If users don't specify a value for the `DurationSeconds` parameter, their security credentials are valid for one hour.

To change the maximum session duration setting for roles that are assumed using the AWS CLI or API (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane of the IAM console, choose **Roles**.
3. Choose the name of the role to modify.
4. Next to **Maximum CLI/API session duration** choose a value. Or choose **Custom duration** and type a value (in seconds).
5. Choose **Save**.

Your changes don't take effect until the next time someone assumes this role. To learn how to revoke existing sessions for this role, see [Revoking IAM Role Temporary Security Credentials \(p. 277\)](#).

Modifying a Role Permissions Boundary (Console)

To change the maximum permissions allowed for a role, modify the role's [permissions boundary \(p. 370\)](#).

To change the policy used to set the permissions boundary for a role

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Roles**.
3. Choose the name of the role whose [permissions boundary \(p. 370\)](#) you want to change.
4. Choose the **Permissions** tab. If necessary, open the **Permissions boundary** section and then choose **Change boundary**.
5. Select the policy that you want to use for the permissions boundary.
6. Choose **Change boundary**.

Your changes don't take effect until the next time someone assumes this role.

Modifying a Role (AWS CLI)

You can use the AWS Command Line Interface to modify a role. To change the set of tags on a role, see [Managing Tags on IAM Entities \(Console\) \(p. 298\)](#).

Topics

- [Modifying a Role Trust Policy \(AWS CLI\) \(p. 283\)](#)
- [Modifying a Role Permissions Policy \(AWS CLI\) \(p. 284\)](#)
- [Modifying a Role Description \(AWS CLI\) \(p. 284\)](#)
- [Modifying a Role Maximum Session Duration \(AWS CLI\) \(p. 285\)](#)
- [Modifying a Role Permissions Boundary \(AWS CLI\) \(p. 285\)](#)

Modifying a Role Trust Policy (AWS CLI)

To change who can assume a role, you must modify the role's trust policy. You cannot modify the trust policy for a [service-linked role \(p. 176\)](#).

Note

If a user is listed as the principal in a role's trust policy but cannot assume the role, check the user's [permissions boundary \(p. 370\)](#). If a permissions boundary is set for the user, then it must allow the `sts:AssumeRole` action.

To modify a role trust policy (AWS CLI)

1. (Optional) If you don't know the name of the role that you want to modify, run the following command to list the roles in your account:
 - [aws iam list-roles](#)
2. (Optional) To view the current trust policy for a role, run the following command:
 - [aws iam get-role](#)
3. To modify the trusted principals that can access the role, create a text file with the updated trust policy. You can use any text editor to construct the policy.

For example, the following trust policy shows how to reference two AWS accounts in the `Principal` element. This allows users within two separate AWS accounts to assume this role.

```
{  
    "Version": "2012-10-17",  
    "Statement": {  
        "Effect": "Allow",  
        "Principal": {"AWS": [  
            "arn:aws:iam::111122223333:root",  
            "arn:aws:iam::444455556666:root"  
        ]},  
        "Action": "sts:AssumeRole"  
    }  
}
```

If you specify a principal in another account, adding an account to the trust policy of a role is only half of establishing the cross-account trust relationship. By default, no users in the trusted accounts can assume the role. The administrator for the newly trusted account must grant the users the permission to assume the role. To do that, the administrator must create or edit a policy that is attached to the user to allow the user access to the `sts:AssumeRole` action. For more information, see the following procedure or [Granting a User Permissions to Switch Roles \(p. 255\)](#).

4. To use the file that you just created to update the trust policy, run the following command:
 - [aws iam update-assume-role-policy](#)

To allow users in a trusted external account to use the role (AWS CLI)

For more information and detail about this procedure, see [Granting a User Permissions to Switch Roles \(p. 255\)](#).

1. Create a JSON file that contains a permissions policy that grants permissions to assume the role. For example, the following policy contains the minimum necessary permissions:

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

```
        "Effect": "Allow",
        "Action": "sts:AssumeRole",
        "Resource": "arn:aws:iam::ACCOUNT-ID-THAT-CONTAINS-ROLE:role/ROLE-NAME"
    }
}
```

Replace the ARN in the statement with the ARN of the role that the user can assume.

2. Run the following command to upload the JSON file that contains the trust policy to IAM:

- [aws iam create-policy](#)

The output of this command includes the ARN of the policy. Make a note of this ARN because you will need it in a later step.

3. Decide which user or group to attach the policy to. If you don't know the name of the intended user or group, use one of the following commands to list the users or groups in your account:
 - [aws iam list-users](#)
 - [aws iam list-groups](#)
4. Use one of the following commands to attach the policy that you created in the previous step to the user or group:
 - [aws iam attach-user-policy](#)
 - [aws iam attach-group-policy](#)

Modifying a Role Permissions Policy (AWS CLI)

To change the permissions allowed by the role, modify the role's permissions policy (or policies). You cannot modify the permissions policy for a [service-linked role \(p. 176\)](#) in IAM. You might be able to modify the permissions policy within the service that depends on the role. To check whether a service supports this feature, see [AWS Services That Work with IAM \(p. 586\)](#) 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.

To change the permissions allowed by a role (AWS CLI)

1. (Optional) To view the current permissions associated with a role, run the following commands:
 1. [aws iam list-role-policies](#) to list inline policies
 2. [aws iam list-attached-role-policies](#) to list managed policies
2. The command to update permissions for the role differs depending on whether you are updating a managed policy or an inline policy.

To update a managed policy, run the following command to create a new version of the managed policy:

- [aws iam create-policy-version](#)

To update an inline policy, run the following command:

- [aws iam put-role-policy](#)

Modifying a Role Description (AWS CLI)

To change the description of the role, modify the description text.

To change the description of a role (AWS CLI)

1. (Optional) To view the current description for a role, run the following command:
 - [aws iam get-role](#)
2. To update a role's description, run the following command with the `description` parameter:
 - [aws iam update-role](#)

Modifying a Role Maximum Session Duration (AWS CLI)

To specify the maximum session duration setting for roles that are assumed using the AWS CLI or API, modify the maximum session duration setting's value. This setting can have a value from 1 hour to 12 hours. If you do not specify a value, the default maximum of 1 hour is applied. This setting does not limit sessions assumed by AWS services.

Note

Anyone who assumes the role from the AWS CLI or API can use the `duration-seconds` CLI parameter or the `DurationSeconds` API parameter to request a longer session. The `MaxSessionDuration` setting determines the maximum duration of the role session that can be requested using the `DurationSeconds` parameter. If users don't specify a value for the `DurationSeconds` parameter, their security credentials are valid for one hour.

To change the maximum session duration setting for roles that are assumed using the AWS CLI (AWS CLI)

1. (Optional) To view the current maximum session duration setting for a role, run the following command:
 - [aws iam get-role](#)
2. To update a role's maximum session duration setting, run the following command with the `max-sessionduration` CLI parameter or the `MaxSessionDuration` API parameter:
 - [aws iam update-role](#)

Your changes don't take effect until the next time someone assumes this role. To learn how to revoke existing sessions for this role, see [Revoking IAM Role Temporary Security Credentials \(p. 277\)](#).

Modifying a Role Permissions Boundary (AWS CLI)

To change the maximum permissions allowed for a role, modify the role's [permissions boundary \(p. 370\)](#).

To change the managed policy used to set the permissions boundary for a role (AWS CLI)

1. (Optional) To view the current [permissions boundary \(p. 370\)](#) for a role, run the following command:
 - [aws iam get-role](#)
2. To use a different managed policy to update the permissions boundary for a role, run the following command:
 - [aws iam put-role-permissions-boundary](#)

A role can have only one managed policy set as a permissions boundary. If you change the permissions boundary, you change the maximum permissions allowed for a role.

Modifying a Role (AWS API)

You can use the AWS API to modify a role. To change the set of tags on a role, see [Managing Tags on IAM Entities \(Console\) \(p. 298\)](#).

Topics

- [Modifying a Role Trust Policy \(AWS API\) \(p. 286\)](#)
- [Modifying a Role Permissions Policy \(AWS API\) \(p. 287\)](#)
- [Modifying a Role Description \(AWS API\) \(p. 288\)](#)
- [Modifying a Role Maximum Session Duration \(AWS API\) \(p. 288\)](#)
- [Modifying a Role Permissions Boundary \(AWS API\) \(p. 289\)](#)

Modifying a Role Trust Policy (AWS API)

To change who can assume a role, you must modify the role's trust policy. You cannot modify the trust policy for a [service-linked role \(p. 176\)](#).

Note

If a user is listed as the principal in a role's trust policy but cannot assume the role, check the user's [permissions boundary \(p. 370\)](#). If a permissions boundary is set for the user, then it must allow the `sts:AssumeRole` action.

To modify a role trust policy (AWS API)

1. (Optional) If you don't know the name of the role that you want to modify, call the following operation to list the roles in your account:
 - [ListRoles](#)
2. (Optional) To view the current trust policy for a role, call the following operation:
 - [GetRole](#)
3. To modify the trusted principals that can access the role, create a text file with the updated trust policy. You can use any text editor to construct the policy.

For example, the following trust policy shows how to reference two AWS accounts in the `Principal` element. This allows users within two separate AWS accounts to assume this role.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Principal": {"AWS": [  
                "arn:aws:iam::111122223333:root",  
                "arn:aws:iam::444455556666:root"  
            ]},  
            "Action": "sts:AssumeRole"  
        }  
    ]  
}
```

If you specify a principal in another account, adding an account to the trust policy of a role is only half of establishing the cross-account trust relationship. By default, no users in the trusted accounts can assume the role. The administrator for the newly trusted account must grant the users the

permission to assume the role. To do that, the administrator must create or edit a policy that is attached to the user to allow the user access to the `sts:AssumeRole` action. For more information, see the following procedure or [Granting a User Permissions to Switch Roles \(p. 255\)](#).

4. To use the file that you just created to update the trust policy, call the following operation:
 - [UpdateAssumeRolePolicy](#)

To allow users in a trusted external account to use the role (AWS API)

For more information and detail about this procedure, see [Granting a User Permissions to Switch Roles \(p. 255\)](#).

1. Create a JSON file that contains a permissions policy that grants permissions to assume the role. For example, the following policy contains the minimum necessary permissions:

```
{  
    "Version": "2012-10-17",  
    "Statement": {  
        "Effect": "Allow",  
        "Action": "sts:AssumeRole",  
        "Resource": "arn:aws:iam::ACCOUNT-ID-THAT-CONTAINS-ROLE:role/ROLE-NAME"  
    }  
}
```

Replace the ARN in the statement with the ARN of the role that the user can assume.

2. Call the following operation to upload the JSON file that contains the trust policy to IAM:
 - [CreatePolicy](#)

The output of this operation includes the ARN of the policy. Make a note of this ARN because you will need it in a later step.

3. Decide which user or group to attach the policy to. If you don't know the name of the intended user or group, call one of the following operations to list the users or groups in your account:
 - [ListUsers](#)
 - [ListGroups](#)
4. Call one of the following operations to attach the policy that you created in the previous step to the user or group:
 - API: [AttachUserPolicy](#)
 - [AttachGroupPolicy](#)

Modifying a Role Permissions Policy (AWS API)

To change the permissions allowed by the role, modify the role's permissions policy (or policies). You cannot modify the permissions policy for a [service-linked role \(p. 176\)](#) in IAM. You might be able to modify the permissions policy within the service that depends on the role. To check whether a service supports this feature, see [AWS Services That Work with IAM \(p. 586\)](#) 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.

To change the permissions allowed by a role (AWS API)

1. (Optional) To view the current permissions associated with a role, call the following operations:
 1. [ListRolePolicies](#) to list inline policies

2. [ListAttachedRolePolicies](#) to list managed policies
2. The operation to update permissions for the role differs depending on whether you are updating a managed policy or an inline policy.

To update a managed policy, call the following operation to create a new version of the managed policy:

- [CreatePolicyVersion](#)

To update an inline policy, call the following operation:

- [PutRolePolicy](#)

Modifying a Role Description (AWS API)

To change the description of the role, modify the description text.

To change the description of a role (AWS API)

1. (Optional) To view the current description for a role, call the following operation:
 - [GetRole](#)
2. To update a role's description, call the following operation with the description parameter:
 - [UpdateRole](#)

Modifying a Role Maximum Session Duration (AWS API)

To specify the maximum session duration setting for roles that are assumed using the AWS CLI or API, modify the maximum session duration setting's value. This setting can have a value from 1 hour to 12 hours. If you do not specify a value, the default maximum of 1 hour is applied. This setting does not limit sessions assumed by AWS services.

Note

Anyone who assumes the role from the AWS CLI or API can use the `duration-seconds` CLI parameter or the `DurationSeconds` API parameter to request a longer session. The `MaxSessionDuration` setting determines the maximum duration of the role session that can be requested using the `DurationSeconds` parameter. If users don't specify a value for the `DurationSeconds` parameter, their security credentials are valid for one hour.

To change the maximum session duration setting for roles that are assumed using the API (AWS API)

1. (Optional) To view the current maximum session duration setting for a role, call the following operation:
 - [GetRole](#)
2. To update a role's maximum session duration setting, call the following operation with the `max-sessionduration` CLI parameter or the `MaxSessionDuration` API parameter:
 - [UpdateRole](#)

Your changes don't take effect until the next time someone assumes this role. To learn how to revoke existing sessions for this role, see [Revoking IAM Role Temporary Security Credentials \(p. 277\)](#).

Modifying a Role Permissions Boundary (AWS API)

To change the maximum permissions allowed for a role, modify the role's [permissions boundary \(p. 370\)](#).

To change the managed policy used to set the permissions boundary for a role (AWS API)

1. (Optional) To view the current [permissions boundary \(p. 370\)](#) for a role, call the following operation:
 - [GetRole](#)
2. To use a different managed policy to update the permissions boundary for a role, call the following operation:
 - [PutRolePermissionsBoundary](#)

A role can have only one managed policy set as a permissions boundary. If you change the permissions boundary, you change the maximum permissions allowed for a role.

Deleting Roles or Instance Profiles

If you no longer need a role, we recommend that you delete the role and its associated permissions. That way you don't have an unused entity that is not actively monitored or maintained.

If the role was associated with an EC2 instance, you can also remove the role from the instance profile and then delete the instance profile.

Warning

Make sure that you do not have any Amazon EC2 instances running with the role or instance profile you are about to delete. Deleting a role or instance profile that is associated with a running instance will break any applications that are running on the instance.

If you prefer not to permanently delete a role, you can disable a role. To do this, change the role's policies and then revoke all current sessions. For example, you could add a policy to the role that denied access to all of AWS. You could also edit the trust policy to deny access to anyone attempting to assume the role. For more information about revoking sessions, see [Revoking IAM Role Temporary Security Credentials \(p. 277\)](#).

Topics

- [View Role Access \(p. 289\)](#)
- [Deleting a Service-Linked Role \(p. 290\)](#)
- [Deleting an IAM Role \(Console\) \(p. 290\)](#)
- [Deleting an IAM Role \(AWS CLI\) \(p. 291\)](#)
- [Deleting an IAM Role \(AWS API\) \(p. 292\)](#)
- [Related Information \(p. 292\)](#)

View Role Access

Before you delete a role, we recommend that you review when the role was last used. You can do this using the AWS Management Console, the AWS CLI, or the AWS API. It's important to view this information because you don't want to remove access from someone who is using it.

The date of the role's last activity might not match the last date reported in the **Access Advisor** tab. The [Access Advisor \(p. 479\)](#) tab reports activity only for services that are allowed by the role's permissions policies. The date of the role's last activity includes the last attempt to access any service in AWS.

Note

The tracking period for a role's last activity and Access Advisor data is for the trailing 400 days. This period can be shorter if your Region began supporting these features within the last year. The role might have been used more than 400 days ago. For more information about the tracking period, see [Regions Where Data Is Tracked \(p. 479\)](#).

To view when a role was last used (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Roles**.
3. Find the row of the role whose activity you want to view. You can use the search field to narrow the results. View the **Last activity** column to see the number of days since the role was last used. If the role has not been used within the tracking period, then the table displays **None**.
4. Choose the name of the role to view more information. The role's **Summary** page also includes **Last activity**, which displays the date that the role was last used. If the role has not been used within the last 400 days, then **Last activity** displays **Not accessed in the tracking period**.

To view when a role was last used (AWS CLI)

```
aws iam get-role
```

- Run this command to return information about a role, including the `RoleLastUsed` object. This object contains the `LastUsedDate` and the `Region` in which the role was last used. If `RoleLastUsed` is present but does not contain a value, then the role has not been used within the tracking period.

To view when a role was last used (AWS API)

`GetRole` - Call this operation to return information about a role, including the `RoleLastUsed` object. This object contains the `LastUsedDate` and the `Region` in which the role was last used. If `RoleLastUsed` is present but does not contain a value, then the role has not been used within the tracking period.

Deleting a Service-Linked Role

If the role is a [service-linked role \(p. 176\)](#), review the documentation for the linked service to learn how to delete the role. You can view the service-linked roles in your account by going to the IAM **Roles** page in the console. Service-linked roles appear with **(Service-linked role)** in the **Trusted entities** column of the table. A banner on the role's **Summary** page also indicates that the role is a service-linked role.

If the service does not include documentation for deleting the service-linked role, you can use the IAM console, AWS CLI, or API to delete the role. For more information, see [Deleting a Service-Linked Role \(p. 226\)](#).

Deleting an IAM Role (Console)

When you use the AWS Management Console to delete a role, IAM also automatically deletes the policies associated with the role. It also deletes any Amazon EC2 instance profile that contains the role.

Important

In some cases, a role might be associated with an Amazon EC2 instance profile, and the role and the instance profile might have the same name. In that case you can use the AWS Management Console to delete the role and the instance profile. This linkage happens automatically for roles and instance profiles that you create in the console. If you created the role from the AWS CLI, Tools for Windows PowerShell, or the AWS API, then the role and the instance profile might have different names. In that case you cannot use the console to delete them. Instead, you must use the AWS CLI, Tools for Windows PowerShell, or AWS API to first remove the role from the instance profile. You must then take a separate step to delete the role.

To delete a role (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Roles**, and then select the check box next to the role name that you want to delete.
3. At the top of the page, choose **Delete role**.
4. In the confirmation dialog box, review the service last accessed data, which shows when each of the selected roles last accessed an AWS service. This helps you to confirm whether the role is currently active. If you want to proceed, choose **Yes, Delete**. If you are sure, you can proceed with the deletion even if the service last accessed data is still loading.

Note

You cannot use the console to delete an instance profile unless it has the same name as the role. In addition, you must delete the instance profile as part of the process of deleting a role as described in the preceding procedure. To delete an instance profile without also deleting the role, you must use the AWS CLI or AWS API. For more information, see the following sections.

Deleting an IAM Role (AWS CLI)

When you use the AWS CLI to delete a role, you must first delete the policies that are associated with the role. Also, if you want to delete the associated instance profile that contains the role, you must delete it separately.

To delete a role (AWS CLI)

1. If you don't know the name of the role that you want to delete, enter the following command to list the roles in your account:

```
$ aws iam list-roles
```

The list includes the Amazon Resource Name (ARN) of each role. Use the role name, not the ARN, to refer to roles with the CLI commands. For example, if a role has the following ARN: `arn:aws:iam::123456789012:role/myrole`, you refer to the role as `myrole`.

2. Remove the role from all instance profiles that the role is in.
 - a. To list all instance profiles that the role is associated with, enter the following command:

```
$ aws iam list-instance-profiles-for-role --role-name role-name
```

- b. To remove the role from an instance profile, enter the following command for each instance profile:

```
$ aws iam remove-role-from-instance-profile --instance-profile-name instance-profile-name --role-name role-name
```

3. Delete all policies that are associated with the role.
 - a. To list all policies that are in the role, enter the following command:

```
$ aws iam list-role-policies --role-name role-name
```

- b. To delete each policy from the role, enter the following command for each policy:

```
$ aws iam delete-role-policy --role-name role-name --policy-name policy-name
```

4. Enter the following command to delete the role:

```
$ aws iam delete-role --role-name role-name
```

5. If you do not plan to reuse the instance profiles that were associated with the role, you can enter the following command to delete them:

```
$ aws iam delete-instance-profile --instance-profile-name instance-profile-name
```

Deleting an IAM Role (AWS API)

When you use the IAM API to delete a role, you must first delete the policies associated with the role. Also, if you want to delete the associated instance profile that contains the role, you must delete it separately.

To delete a role (AWS API)

1. To list all instance profiles that a role is in, call [ListInstanceProfilesForRole](#).

To remove the role from all instance profiles that the role is in, call [RemoveRoleFromInstanceProfile](#). You must pass the role name and instance profile name.

If you are not going to reuse an instance profile that was associated with the role, call [DeleteInstanceProfile](#) to delete it.

2. To list all policies for a role, call [ListRolePolicies](#).

To delete all policies that are associated with the role, call [DeleteRolePolicy](#). You must pass the role name and policy name.

3. Call [DeleteRole](#) to delete the role.

Related Information

For general information about instance profiles, see [Using Instance Profiles \(p. 276\)](#).

For general information about service-linked roles, see [Using Service-Linked Roles \(p. 221\)](#).

How IAM Roles Differ from Resource-based Policies

For some AWS services, you can grant cross-account access to your resources. To do this, you attach a policy directly to the resource that you want to share, instead of using a role as a proxy. The resource that you want to share must support [resource-based policies \(p. 379\)](#). Unlike an identity-based policy, a resource-based policy specifies who (which principal) can access that resource.

Note

IAM roles and resource-based policies delegate access across accounts only within a single partition. For example, assume that you have an account in US West (N. California) in the standard aws partition. You also have an account in China (Beijing) in the aws-cn partition. You can't use an Amazon S3 resource-based policy in your account in China (Beijing) to allow access for users in your standard aws account.

Cross-account access with a resource-based policy has some advantages over cross-account access with a role. With a resource that is accessed through a resource-based policy, the principal still works in the trusted account and does not have to give up his or her permissions to receive the role permissions. In other words, the principal continues to have access to resources in the trusted account at the same time

as he or she has access to the resource in the trusting account. This is useful for tasks such as copying information to or from the shared resource in the other account. To learn whether principals in accounts outside of your zone of trust (trusted organization, OU, or account) have access to assume your roles, see [What is IAM Access Analyzer?](#).

The principals that you can specify in a resource based policy include accounts, IAM users, federated users, IAM roles, assumed-role sessions, or AWS services. For more information, see [Specifying a Principal \(p. 601\)](#).

The following list includes some of the AWS services that support resource-based policies. For a complete list of the growing number of AWS services that support attaching permission policies to resources instead of principals, see [AWS Services That Work with IAM \(p. 586\)](#) and look for the services that have Yes in the **Resource Based** column.

- **Amazon S3 buckets** – The policy is attached to the bucket, but the policy controls access to both the bucket and the objects in it. For more information, go to [Access Control](#) in the *Amazon Simple Storage Service Developer Guide*.

In some cases, it may be best to use roles for cross-account access to Amazon S3. For more information, see the [example walkthroughs](#) in the *Amazon Simple Storage Service Developer Guide*.

- **Amazon Simple Notification Service (Amazon SNS) topics** – For more information, go to [Managing Access to Your Amazon SNS Topics](#) in the *Amazon Simple Notification Service Developer Guide*.
- **Amazon Simple Queue Service (Amazon SQS) queues** – For more information, go to [Appendix: The Access Policy Language](#) in the *Amazon Simple Queue Service Developer Guide*.

About Delegating AWS Permissions in a Resource-based Policy

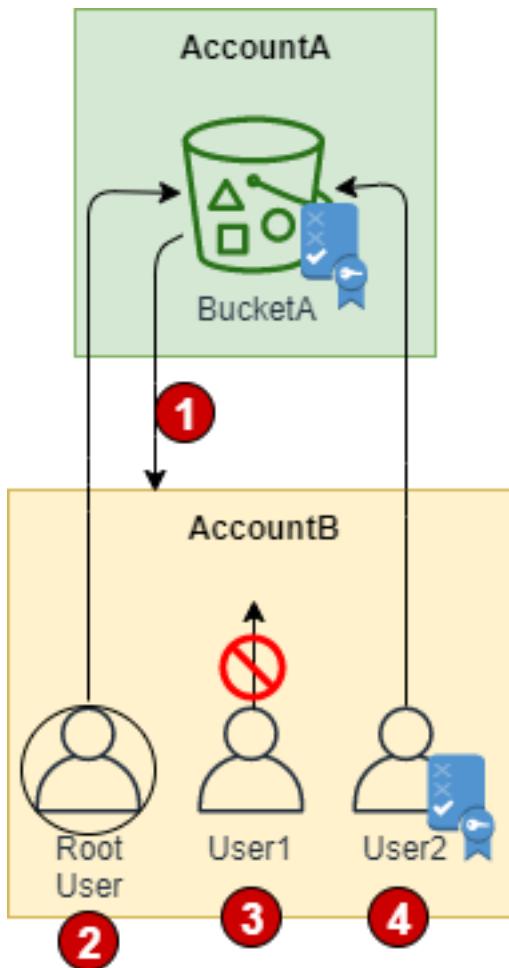
If a resource grants permissions to principals in your account, you can then delegate those permissions to specific IAM identities. Identities are users, groups of users, or roles in your account. You delegate permissions by attaching a policy to the identity. You can grant up to the maximum permissions that are allowed by the resource-owning account.

Assume that a resource-based policy allows all principals in your account full administrative access to a resource. Then you can delegate full access, read-only access, or any other partial access to principals in your AWS account. Alternatively, if the resource-based policy allows only list permissions, then you can delegate only list access. If you try to delegate more permissions than your account has, your principals will still have only list access. For information about attaching a policy to an IAM identity, see [Managing IAM Policies \(p. 442\)](#).

Note

IAM roles and resource-based policies delegate access across accounts only within a single partition. For example, you can't add cross-account access between an account in the standard aws partition and an account in the aws-cn partition.

For example, assume that you manage AccountA and AccountB. In AccountA, you have the Amazon S3 bucket named BucketA. You attach a resource-based policy to BucketA that allows all AccountB principals full access to objects in your bucket. They can create, read, or delete any objects in that bucket. In AccountB, you attach a policy to the IAM user named User2. That policy allows the user read-only access to the objects in BucketA. That means that User2 can view the objects, but not create, edit, or delete them.



1. AccountA gives AccountB full access to BucketA by naming AccountB as a principal in the resource-based policy. As a result, AccountB is authorized to perform any action on BucketA, and the AccountB administrator can delegate access to its users in AccountB.
2. The AccountB root user has all of the permissions that are granted to the account. Therefore, the root user has full access to BucketA.
3. The AccountB administrator does not give access to User1. By default, users do not have any permissions except those that are explicitly granted. Therefore, User1 does not have access to BucketA.
4. The AccountB administrator grants User2 read-only access to BucketA. User2 can view the objects in the bucket. The maximum level of access that AccountB can delegate is the access level that is granted to the account. In this case, the resource-based policy granted full access to AccountB, but User2 is granted only read-only access.

IAM evaluates a principal's permissions at the time the principal makes a request. Therefore, if you use wildcards (*) to give users full access to your resources, principals can access any resources that your AWS account has access to. This is true even for resources you add or gain access to after creating the user's policy.

In the preceding example, if AccountB had attached a policy to User2 that allowed full access to all resources in all accounts, User2 would automatically have access to any resources that AccountB has access to. This includes the BucketA access and access to any other resources granted by resource-based policies in AccountA.

Important

Give access only to entities you trust, and give the minimum level of access necessary. Whenever the trusted entity is another AWS account, that account can in turn delegate access to any of its IAM users. The trusted AWS account can delegate access only to the extent that it has been granted access; it cannot delegate more access than the account itself has been granted.

For information about permissions, policies, and the permission policy language that you use to write policies, see [Access Management \(p. 355\)](#).

Tagging IAM Users and Roles

You can use IAM tags to add custom attributes to an IAM entity (user or role) using a tag key-value pair. For example, to add location information to a user, you can add the tag key **location** and the tag value **us_wa_seattle**. Or you could use three separate location tag key-value pairs: **loc-country = us**, **loc-state = wa**, and **loc-city = seattle**. You can use tags to control an entity's access to resources or to control what tags can be attached to an entity. To learn more about using tags to control access, see [Controlling Access to and for IAM Users and Roles Using IAM Resource Tags \(p. 389\)](#).

You can also use tags in AWS STS to add custom attributes when you assume a role or federate a user. For more information, see [Passing Session Tags in AWS STS \(p. 299\)](#).

Choose an AWS Tag Naming Convention

When you begin attaching tags to your IAM users and roles, choose your tag naming convention carefully. Apply the same convention to all of your AWS tags. This is especially important if you use tags in policies to control access to AWS resources. If you already use tags in AWS, review your naming convention and adjust it accordingly. To learn more about creating a naming strategy, see [AWS Tagging Strategies](#).

Rules for Tagging in IAM And AWS STS

A number of conventions govern the creation and application of tags in IAM and AWS STS.

Naming Tags

Observe the following conventions when formulating a tag naming convention for IAM users, IAM roles, AWS STS assume-role sessions, and AWS STS federated user sessions:

- Tag keys and values can include any combination of letters, numbers, spaces, and `_ . : / = + - @ .` symbols.
- Tag key-value pairs are not case sensitive, but case is preserved. This means that you cannot have separate **Department** and **department** tag keys. If you have tagged a user with the **Department=foo** tag and you add the **department=bar** tag, it replaces the first tag. A second tag is not added.
- You cannot create a tag key or value that begins with the text **aws:**. This tag prefix is reserved for AWS internal use.
- You can create a tag with an empty value such as **phoneNumber = .** You cannot create an empty tag key.
- You cannot specify multiple values in a single tag, but you can create a custom multivalue structure in the single value. For example, assume that the user Zhang works on the engineering team and the QA team. If you attach the **team = Engineering** tag and then attach the **team = QA** tag, you change the value of the tag from **Engineering** to **QA**. Instead, you can include multiple values in a single tag with a custom separator. In this example, you could attach the **team = Engineering:QA** tag to Zhang.

Note

To control access to engineers in this example using the `team` tag, you must create a policy that allows for every configuration that might include `Engineering`, including `Engineering:QA`. To learn more about using tags in policies, see [Controlling Access to and for IAM Users and Roles Using IAM Resource Tags \(p. 389\)](#).

Applying and Editing Tags

Observe the following conventions when attaching tags to IAM entities (users or roles):

- You can tag users or roles but not groups or policies.
- You cannot use Tag Editor to tag IAM entities. Tag Editor does not support IAM tags. For information about using Tag Editor with other services, see [Working with Tag Editor in the AWS Management Console User Guide](#).
- To tag an IAM entity, you must have specific permissions. To tag or untag roles and users, you must also have permission to list tags. For more information, see [Permissions Required for Tagging IAM Entities \(p. 296\)](#) following.
- There are limits to the number and size of tags you can attach to a user or role. For details, see [IAM and STS Limits \(p. 581\)](#).
- You can apply the same tag to multiple IAM entities. For example, suppose you have a department named `AWS_Development` with 12 members. You can have 12 users and a role with the tag key of `department` and a value of `awsDevelopment` (`department = awsDevelopment`). You can also use the same tag on resources in other [services that support tagging \(p. 586\)](#).
- An IAM entity cannot have multiple instances of the same tag key. For example, if you have a user with the tag key-value pair `costCenter = 1234`, you can then attach the tag key-value pair `costCenter = 5678`. IAM updates the value of the `costCenter` tag to `5678`.
- To edit a tag that is attached to an IAM user or role, attach a tag with a new value to overwrite the existing tag. For example, assume that you have a user with the tag key-value pair `department = Engineering`. If you need to move the user to the QA department, then you can attach the `department = QA` tag key-value pair to the user. This results in the `Engineering` value of the `department` tag key being replaced with the `QA` value.

Permissions Required for Tagging IAM Entities

You must configure permissions to allow an IAM entity (user or role) to tag other entities. You can specify one or all of the following IAM tag actions in an IAM policy:

- `iam>ListRoleTags`
- `iam>ListUserTags`
- `iam:TagRole`
- `iam:TagUser`
- `iam:UntagRole`
- `iam:UntagUser`

To allow an IAM entity to add, list, or remove a tag for a specific user

Add the following statement to the permissions policy for the IAM entity that needs to manage tags. Use your account number and replace `<username>` with the name of the user that needs to be managed. To learn how to create a policy using this example JSON policy document, see [the section called "Creating Policies on the JSON Tab" \(p. 446\)](#).

```
{  
    "Effect": "Allow",  
    "Action": [  
        "iam>ListUserTags",  
        "iam:TagUser",  
        "iam:UntagUser"  
    ],  
    "Resource": "arn:aws:iam:*:<account-number>:user/<username>"  
}
```

To allow an IAM user to self-manage tags

Add the following statement to the permissions policy for users to allow users to manage their own tags. To learn how to create a policy using this example JSON policy document, see [the section called “Creating Policies on the JSON Tab” \(p. 446\)](#).

```
{  
    "Effect": "Allow",  
    "Action": [  
        "iam>ListUserTags",  
        "iam:TagUser",  
        "iam:UntagUser"  
    ],  
    "Resource": "arn:aws:iam:*:user/${aws:username}"  
}
```

To allow an IAM entity to add a tag to a specific user

Add the following statement to the permissions policy for the IAM entity that needs to add, but not remove, tags for a specific user.

Note

The `iam:AddRoleTags` and `iam:AddUserTags` actions require that you also include the `iam>ListRoleTags` and `iam>ListUserTags` actions.

To use this policy, replace `<username>` with the name of the user that needs to be managed. To learn how to create a policy using this example JSON policy document, see [the section called “Creating Policies on the JSON Tab” \(p. 446\)](#).

```
{  
    "Effect": "Allow",  
    "Action": [  
        "iam>ListUserTags",  
        "iam:TagUser"  
    ],  
    "Resource": "arn:aws:iam:*:<account-number>:user/<username>"  
}
```

To allow an IAM entity to add, list, or remove a tag for a specific role

Add the following statement to the permissions policy for the IAM entity that needs to manage tags. Replace `<rolename>` with the name of the role that needs to be managed. To learn how to create a policy using this example JSON policy document, see [the section called “Creating Policies on the JSON Tab” \(p. 446\)](#).

```
{  
    "Effect": "Allow",  
    "Action": [  
        "iam>ListRoleTags",  
        "iam:TagRole",  
        "iam:UntagRole"  
    ],  
    "Resource": "arn:aws:iam:*:<account-number>:role/<rolename>"  
}
```

```
        "iam:UntagRole"
],
"Resource": "arn:aws:iam::*:<account-number>:role/<rolename>"
}
```

Alternatively, you can use an AWS managed policy such as [IAMFullAccess](#) to provide full access to IAM.

Managing Tags on IAM Entities (Console)

You can manage tags for IAM users or roles from the AWS Management Console.

To manage tags on users or roles (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane of the console, choose **Roles** or **Users** and then choose the name of the entity that you want to edit.
3. Choose the **Tags** tab and then complete one of the following actions:
 - Choose **Add tags** if the entity does not yet have tags.
 - Choose **Edit tags** to manage the existing set of tags.
4. Add or remove tags to complete the set of tags. Then choose **Save changes**.

Managing Tags on IAM Entities (AWS CLI or AWS API)

You can list, attach, or remove tags for IAM users and roles. You can use the AWS CLI or the AWS API to manage tags for IAM users and roles.

To list the tags currently attached to an IAM role (AWS CLI or AWS API)

- AWS CLI: [aws iam list-role-tags](#)
- AWS API: [ListRoleTags](#)

To list the tags currently attached to an IAM user (AWS CLI or AWS API)

- AWS CLI: [aws iam list-user-tags](#)
- AWS API: [ListUserTags](#)

To attach tags to an IAM role (AWS CLI or AWS API)

- AWS CLI: [aws iam tag-role](#)
- AWS API: [TagRole](#)

To attach tags to an IAM user (AWS CLI or AWS API)

- AWS CLI: [aws iam tag-user](#)
- AWS API: [TagUser](#)

To remove tags from an IAM role (AWS CLI or AWS API)

- AWS CLI: [aws iam untag-role](#)

- AWS API: [UntagRole](#)

To remove tags from an IAM user (AWS CLI or AWS API)

- AWS CLI: `aws iam untag-user`
- AWS API: [UntagUser](#)

For information about attaching tags to resources for other AWS services, see the documentation for those services.

For information about using tags to set more granular permissions with IAM permissions policies, see [IAM Policy Elements: Variables and Tags \(p. 628\)](#).

Passing Session Tags in AWS STS

Session tags are key-value pair attributes that you pass when you assume an IAM role or federate a user in AWS STS. You do this by making an AWS CLI or AWS API request through STS or through your identity provider (IdP). When you use AWS STS to request temporary security credentials, you generate a session. Sessions expire and have [credentials](#), such as an access key pair and a session token. When you use the session credentials to make a subsequent request, the [request context \(p. 612\)](#) includes the [aws:PrincipalTag \(p. 670\)](#) context key. You can use the `aws:PrincipalTag` key in the Condition element of your policies to allow or deny access based on those tags.

When you use temporary credentials to make a request, your principal might include a set of tags. These tags come from the following sources:

1. **Session tags** – These tags were passed when you assumed the role or federated the user using the AWS CLI or AWS API. For more information about these operations, see [Session Tagging Operations \(p. 299\)](#) below.
2. **Incoming transitive session tags** – These tags were inherited from a previous session in a role chain. For more information, see [Chaining Roles with Session Tags \(p. 305\)](#) later in this topic.
3. **IAM tags** – These tags were attached to the IAM role that you assumed.

Topics

- [Session Tagging Operations \(p. 299\)](#)
- [Things to Know About Session Tags \(p. 300\)](#)
- [Permissions Required to Add Session Tags \(p. 301\)](#)
- [Passing Session Tags Using AssumeRole \(p. 303\)](#)
- [Passing Session Tags using AssumeRoleWithSAML \(p. 303\)](#)
- [Passing Session Tags using AssumeRoleWithWebIdentity \(p. 304\)](#)
- [Passing Session Tags using GetFederationToken \(p. 305\)](#)
- [Chaining Roles with Session Tags \(p. 305\)](#)
- [Using Session Tags for ABAC \(p. 306\)](#)
- [Viewing Session Tags in CloudTrail \(p. 306\)](#)

Session Tagging Operations

You can pass session tags using the following AWS CLI or AWS API operations in AWS STS. The AWS Management Console [Switch Role \(p. 260\)](#) feature does not allow you to pass session tags.

You can also set the session tags as transitive. Transitive tags persist during role chaining. For more information, see [Chaining Roles with Session Tags \(p. 305\)](#).

Comparing Methods for Passing Session Tags

Method	Who Can Assume the Role	Method to Pass Tags	Method to Set Transitive Tags
assume-role CLI or <code>AssumeRole</code> API operation	IAM user or a session	Tags API parameter or <code>--tags</code> CLI option	<code>TransitiveTagKeys</code> API parameter or <code>--transitive-tag-keys</code> CLI option
assume-role-with-saml CLI or <code>AssumeRoleWithSAML</code> API operation	Any user authenticated using SAML identity provider	<code>PrincipalTag</code> SAML attribute	<code>TransitiveTagKeys</code> SAML Attribute
assume-role-with-web-identity CLI or <code>AssumeRoleWithWebIdentity</code> API operation	Any user authenticated using a web identity provider	<code>PrincipalTag</code> web identity token	<code>TransitiveTagKeys</code> web identity token
get-federation-token CLI or <code>GetFederationToken</code> API operation	IAM user or root user	Tags API parameter or <code>--tags</code> CLI option	Not supported

Operations that support session tagging can fail if any of the following conditions are true:

- You pass more than 50 session tags.
- The plain text of your session tag keys exceeds 128 characters.
- The plain text of your session tag values exceeds 256 characters.
- The total size of the plain text of session policies exceeds 2048 characters.
- The total packed size of the session policies and session tags combined is too large. If the operation fails, the error message indicates by percentage how close the policies and tags combined are to the upper size limit.

Things to Know About Session Tags

Before you use session tags, review the following details about sessions and tags.

- Session tags are principal tags that you specify while requesting a session. The tags apply to requests that you make using the session's credentials.
- Session tags are key-value pairs. For example, to add contact information to a session, you can add the session tag key `email` and the tag value `johndoe@example.com`.
- Session tags must follow the [rules for naming tags in IAM and AWS STS \(p. 295\)](#). This topic includes information about case sensitivity and restricted prefixes that apply to your session tags.
- New session tags override existing assumed role or federated user tags with the same tag key, regardless of case.
- You cannot pass session tags using the AWS Management Console.
- Session tags are valid only for the current session.

- Session tags support [role chaining \(p. 177\)](#). By default, tags are not passed to subsequent role sessions. However, you can set session tags as transitive. Transitive tags persist during role chaining. For more information, see [Chaining Roles with Session Tags \(p. 305\)](#).
- You can use session tags to control access to resources or to control what tags can be passed into a subsequent session. For more information, see [Using SAML Session Tags for ABAC \(p. 53\)](#).
- You can view the principal tags for your session, including its session tags, in the AWS CloudTrail logs. For more information, see [Viewing Session Tags in CloudTrail \(p. 306\)](#).
- You must pass a single value for each session tag. Multivalued session tags are not supported.
- You can pass a maximum of 50 session tags. For information about these and other limits, see [IAM and STS Limits \(p. 581\)](#).
- An AWS conversion compresses the passed session policies and session tags combined into a packed binary format that has a separate limit. If you exceed this limit, the AWS CLI or AWS API error message indicates by percentage how close the policies and tags combined are to the upper size limit.

Permissions Required to Add Session Tags

In addition to the action that matches the API operation, you must have the following permissions-only action in your policy:

```
sts:TagSession
```

You can use this action with the following condition keys.

- [aws:PrincipalTag \(p. 670\)](#) – Use this key to compare the tag that is attached to the principal making the request with the tag that you specify in the policy. For example, you can allow a principal to pass session tags only if the principal making the request has the specified tags.
- [aws:RequestTag \(p. 672\)](#) – Use this key to compare the tag key-value pair that was passed in the request with the tag pair that you specify in the policy. For example, you can allow the principal to pass the specified session tags, but only with the specified values.
- [aws:ResourceTag \(p. 672\)](#) – Use this key to compare the tag key-value pair that you specify in the policy with the key-value pair that is attached to the resource. For example, you can allow the principal to pass session tags only if the role they are assuming includes the specified tags.
- [aws:TagKeys \(p. 674\)](#) – Use this key to compare the tag keys in a request with the keys that you specify in the policy. For example, you can allow the principal to pass only session tags with the specified tag keys. This condition key limits the maximum set of session tags that can be passed.
- [sts:TransitiveTagKeys \(p. 686\)](#) – Use this key to compare the transitive session tag keys in the request with those specified in the policy. For example, you can write a policy to allow a principal to set only specific tags as transitive. Transitive tags persist during role chaining. For more information, see [Chaining Roles with Session Tags \(p. 305\)](#).

For example, the following [role trust policy \(p. 178\)](#) allows the test-session-tags user to assume the role to which the policy is attached. When that user assumes the role, they must use the AWS CLI or AWS API to pass the three required session tags and the required [external ID \(p. 233\)](#). Additionally, the user can choose to set the Project and Department tags as transitive.

Example Role Trust Policy for Session Tags

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "AllowIamUserAssumeRole",  
            "Effect": "Allow",  
            "Action": "sts:AssumeRole",  
            "Condition": {  
                "StringEquals": {  
                    "sts:ExternalId": "test-session-tags",  
                    "sts:TagKeys": ["Project", "Department"]  
                }  
            }  
        }  
    ]  
}
```

```

    "Principal": {"AWS": "arn:aws:iam::123456789012:user/test-session-tags"},
    "Condition": {
        "StringLike": {
            "aws:RequestTag/Project": "*",
            "aws:RequestTag/CostCenter": "*",
            "aws:RequestTag/Department": "*"
        },
        "StringEquals": {"sts:ExternalId": "Example987"}
    }
},
{
    "Sid": "AllowPassSessionTagsAndTransitive",
    "Effect": "Allow",
    "Action": "sts:TagSession",
    "Principal": {"AWS": "arn:aws:iam::123456789012:user/test-session-tags"},
    "Condition": {
        "StringLike": {
            "aws:RequestTag/Project": "*",
            "aws:RequestTag/CostCenter": "*"
        },
        "StringEquals": {
            "aws:RequestTag/Department": [
                "Engineering",
                "Marketing"
            ]
        },
        "ForAllValues:StringEquals": {
            "sts:TransitiveTagKeys": [
                "Project",
                "Department"
            ]
        }
    }
}
]
}

```

What does this policy do?

- The AllowIAMUserAssumeRole statement allows the test-session-tags user to assume the role to which the policy is attached. When that user assumes the role, they must pass the required session tags and external ID ([p. 233](#)).
- The first condition block of this statement requires the user to pass the Project, CostCenter, and Department session tags. The tag values don't matter in this statement, so we used wildcards (*) for the tag values. This block ensures that user passes at least these three session tags or the operation will fail. The user can pass additional tags.
- The second condition block requires the user to pass an external ID ([p. 233](#)) with the value Example987.
- The AllowPassSessionTagsAndTransitive statement allows the sts:TagSession permissions-only action. This action must be allowed before the user can pass session tags. If your policy includes the first statement without the second statement, the user can't assume the role.
 - The first condition block of this statement allows the user to pass any value for the CostCenter and Project session tags. You do this by using wildcards (*) for the tag value in the policy, which requires that you use the StringLike ([p. 614](#)) condition operator.
 - The second condition block allows the user to pass only the Engineering or Marketing value for the Department session tag.
 - The third condition block lists the maximum set of tags that can be set as transitive. The user can choose to set a subset or no tags as transitive. But they cannot set additional tags as transitive. You can require that they set at least one of the tags as transitive by adding another condition block that includes "Null": {"sts:TransitiveTagKeys": "false"}.

Passing Session Tags Using AssumeRole

The `AssumeRole` operation returns a set of temporary credentials that you can use to access AWS resources. You can use IAM user or role credentials to call `AssumeRole`. To pass session tags while assuming a role, use the `--tags` AWS CLI option or the `Tags` AWS API parameter.

To set tags as transitive, use the `--transitive-tag-keys` AWS CLI option or the `TransitiveTagKeys` AWS API parameter. Transitive tags persist during role chaining. For more information, see [Chaining Roles with Session Tags \(p. 305\)](#).

The following example shows a sample request that uses `AssumeRole`. In this example, when you assume the `my-role-example` role, you create a session named `my-session`. You add the session tag key-value pairs `Project = Automation`, `CostCenter = 12345`, and `Department = Engineering`. You also set the `Project` and `Department` tags as transitive by specifying their keys.

Example AssumeRole CLI Request

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/my-role-example \
--role-session-name my-session \
--tags Key=Project,Value=Automation Key=CostCenter,Value=12345
Key=Department,Value=Engineering \
--transitive-tag-keys Project Department \
--external-id Example987
```

Passing Session Tags using AssumeRoleWithSAML

The `AssumeRoleWithSAML` operation is authenticated using SAML-based federation. This operation returns a set of temporary credentials that you can use to access AWS resources. For more information about using SAML-based federation for AWS Management Console access, see [Enabling SAML 2.0 Federated Users to Access the AWS Management Console \(p. 211\)](#). For details about AWS CLI or AWS API access, see [About SAML 2.0-based Federation \(p. 190\)](#). For a tutorial of setting up SAML federation for your Active Directory users, see [AWS Federated Authentication with Active Directory Federation Services \(ADFS\) in the AWS Security Blog](#).

As an administrator, you can allow members of your company directory to federate into AWS using the AWS STS `AssumeRoleWithSAML` operation. To do this, you must complete the following tasks:

1. [Configure your network as a SAML provider for AWS \(p. 204\)](#)
2. [Create a SAML provider in IAM \(p. 201\)](#)
3. [Configure a role and its permissions in AWS for your federated users \(p. 248\)](#)
4. [Finish configuring the SAML IdP and create assertions for the SAML authentication response \(p. 206\)](#)

AWS includes partners that have certified the end-to-end experience for session tags with their identity solutions. To learn how to use these identity providers to configure session tags, see [Integrating Third-Party SAML Solution Providers with AWS \(p. 204\)](#).

To pass SAML attributes as session tags, include the `Attribute` element with the `Name` attribute set to `https://aws.amazon.com/SAML/Attributes/PrincipalTag:{TagKey}`. Use the `AttributeValue` element to specify the value of the tag. Include a separate `Attribute` element for each session tag.

For example, assume that you want to pass the following identity attributes as session tags:

- `Project:Automation`
- `CostCenter:12345`
- `Department:Engineering`

To pass these attributes, include the following elements in your SAML assertion.

Example Snippet of a SAML Assertion

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:Project">
    <AttributeValue>Automation</AttributeValue>
</Attribute>
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:CostCenter">
    <AttributeValue>12345</AttributeValue>
</Attribute>
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:Department">
    <AttributeValue>Engineering</AttributeValue>
</Attribute>
```

To set the tags above as transitive, include another `Attribute` element with the `Name` attribute set to `https://aws.amazon.com/SAML/Attributes/TransitiveTagKeys`. Transitive tags persist during role chaining. For more information, see [Chaining Roles with Session Tags \(p. 305\)](#).

To set the `Project` and `Department` tags as transitive, use the following multivalued attribute.

Example Snippet of a SAML Assertion

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/TransitiveTagKeys">
    <AttributeValue>Project</AttributeValue>
    <AttributeValue>Department</AttributeValue>
</Attribute>
```

Passing Session Tags using AssumeRoleWithWebIdentity

The `AssumeRoleWithWebIdentity` operation is authenticated using OpenID Connect (OIDC)-compliant web identity federation. This operation returns a set of temporary credentials that you can use to access AWS resources. For more information about using web identity federation for AWS Management Console access, see [About Web Identity Federation \(p. 185\)](#).

To pass session tags from OpenID Connect (OIDC), you must include the session tags in the JSON Web Token (JWT). Include session tags in the `https://aws.amazon.com/tags` namespace in the token when you submit the `AssumeRoleWithWebIdentity` request. To learn more about OIDC tokens and claims, see [Using Tokens with User Pools](#) in the *Amazon Cognito Developer Guide*.

For example, the following decoded JWT is a token that is used to call `AssumeRoleWithWebIdentity` with the `Project`, `CostCenter`, and `Department` session tags. The token also sets the `Project` and `CostCenter` tags as transitive. Transitive tags persist during role chaining. For more information, see [Chaining Roles with Session Tags \(p. 305\)](#).

Example Decoded JSON Web Token

```
{
    "sub": "johndoe",
    "aud": "ac_oic_client",
    "jti": "ZYUCeRMQVtqHypVPWAN3VB",
    "iss": "https://xyz.com",
    "iat": 1566583294,
    "exp": 1566583354,
    "auth_time": 1566583292,
    "https://aws.amazon.com/tags": {
        "principal_tags": {
            "Project": ["Automation"],
            "CostCenter": ["987654"],
            "Department": ["Engineering"]
```

```
        },
        "transitive_tag_keys": [
            "Project",
            "CostCenter"
        ]
    }
}
```

Passing Session Tags using GetFederationToken

The `GetFederationToken` lets you federate your user. This operation returns a set of temporary credentials that you can use to access AWS resources. To add tags to your federated user session, use the `--tags` AWS CLI option or the `Tags` AWS API parameter. You can't set session tags as transitive when you use `GetFederationToken`. This is because you can't use the temporary credentials to assume a role, which means that role chaining is not possible.

The following example shows a sample request using `GetFederationToken`. In this example, when you request the token, you create a session named `my-fed-user`. You add the session tag key-value pairs `Project = Automation` and `Department = Engineering`.

Example GetFederationToken CLI Request

```
aws sts get-federation-token \
--name my-fed-user \
--tags key=Project,value=Automation key=Department,value=Engineering
```

When you use the temporary credentials that are returned by the `GetFederationToken` operation, the session's principal tags include the user's tags and the passed session tags.

Chaining Roles with Session Tags

You can assume one role and then use the temporary credentials to assume another role. You can continue from session to session. This is called [role chaining \(p. 177\)](#). When you pass session tags while assuming a role, you can set the keys as transitive. This ensures that those session tags pass to subsequent sessions in a role chain. You cannot set role tags as transitive. To pass these tags to subsequent sessions, specify them as session tags.

The following example will help you understand how session tags, transitive tags, and role tags are passed into subsequent sessions in a role chain.

In the following example role chaining scenario, you use an IAM user's access keys in the AWS CLI to assume a role named `Role1`. You then use the resulting session credentials to assume a second role named `Role2`. You can then use the second session credentials to assume a third role named `Role3`. These requests occur as three separate operations. Each role is already tagged in IAM. And during each request, you pass additional session tags.

When you chain roles, you can ensure that tags from an earlier session persist to the later sessions. To do this using the `assume-role` CLI command, you must pass the tag as a session tag and set the tag as transitive. You pass the tag `Star = 1` as a session tag. The tag `Heart = 1` is attached to the role and will apply as a principal tag when you use the session. However, you also want the `Heart = 1` tag to automatically pass to the second or third session. To do that, you manually include it as a session tag. That way the resulting session's principal tags include these two tags, and they are set as transitive.

You perform this request using the following AWS CLI command:

Example AssumeRole CLI Request

```
aws sts assume-role \
```

```
--role-arn arn:aws:iam::123456789012:role/Role1 \
--role-session-name Session1 \
--tags Key=Star,Value=1 Key=Heart,Value=1 \
--transitive-tag-keys Star Heart
```

You then use the credentials for that session to assume Role2. The tag Sun = 2 is attached to the second role and will apply as a principal tag when you use the second session. The Heart and Star tags are inherited from the transitive session tags in the first session. The second session's resulting principal tags are Heart = 1, Star = 1, and Sun = 2. Heart and Star will continue to be transitive. The Sun tag that was attached to Role2 is not marked as transitive because it is not a session tag. This tag will not be inherited by future sessions.

You perform this second request using the following AWS CLI command:

Example AssumeRole CLI Request

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/Role2 \
--role-session-name Session2
```

You then use the second session credentials to assume Role3. The principal tags for the third session come from any new session tags, the inherited transitive session tags, and the role tags. The Heart = 1 and Star = 1 tags on the second session were inherited from the transitive session tag in the first session. If you try to pass the Heart = 3 session tag, the operation will fail. The inherited Star = 1 session tag overrides the role's Star = 3 tag. The role's Lightning tag also applies to the third session, and is not set as transitive.

You perform the third request using the following AWS CLI command:

Example AssumeRole CLI Request

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/Role3 \
--role-session-name Session3
```

Using Session Tags for ABAC

Attribute-based access control (ABAC) is an authorization strategy that defines permissions based on tag attributes.

If your company uses a SAML-based identity provider (IdP) for corporate user identities, you can configure your SAML assertion to pass session tags to AWS. When your employees federate into AWS, their attributes are applied to their resulting principal in AWS. You can then use ABAC to allow or deny permissions based on those attributes. For details, see [Using SAML Session Tags for ABAC \(p. 53\)](#).

Viewing Session Tags in CloudTrail

You can use AWS CloudTrail to view the requests made to assume roles or federate users. The CloudTrail log file includes information about the principal tags for the assumed-role or federated user session. For more information, see [Logging IAM and AWS STS API Calls with AWS CloudTrail \(p. 341\)](#).

For example, assume that you make an AWS STS AssumeRoleWithSAML request, pass session tags, and set those tags as transitive. You can find the following information in your CloudTrail log.

Example AssumeRoleWithSAML CloudTrail Log

```
"requestParameters": {
```

```
"SAMLAssertionID": "_c0046cEXAMPLEb9d4b8eEXAMPLE2619aEXAMPLE",
"roleSessionName": "MyRoleSessionName",
"principalTags": {
    "CostCenter": "987654",
    "Project": "Unicorn"
},
"transitiveTagKeys": [
    "CostCenter",
    "Project"
],
"durationSeconds": 3600,
"roleArn": "arn:aws:iam::123456789012:role/SAMLTestRoleShibboleth",
"principalArn": "arn:aws:iam::123456789012:saml-provider/Shibboleth"
},
```

You can view the following example CloudTrail logs to view events that use session tags.

- [Example AWS STS Role Chaining API Event in CloudTrail Log File \(p. 348\)](#)
- [Example SAML AWS STS API Event in CloudTrail Log File \(p. 350\)](#)
- [Example Web Identity AWS STS API Event in CloudTrail Log File \(p. 351\)](#)

Temporary Security Credentials

You can use the AWS Security Token Service (AWS STS) to create and provide trusted users with temporary security credentials that can control access to your AWS resources. Temporary security credentials work almost identically to the long-term access key credentials that your IAM users can use, with the following differences:

- Temporary security credentials are *short-term*, as the name implies. They can be configured to last for anywhere from a few minutes to several hours. After the credentials expire, AWS no longer recognizes them or allows any kind of access from API requests made with them.
- Temporary security credentials are not stored with the user but are generated dynamically and provided to the user when requested. When (or even before) the temporary security credentials expire, the user can request new credentials, as long as the user requesting them still has permissions to do so.

These differences lead to the following advantages for using temporary credentials:

- You do not have to distribute or embed long-term AWS security credentials with an application.
- You can provide access to your AWS resources to users without having to define an AWS identity for them. Temporary credentials are the basis for [roles and identity federation \(p. 175\)](#).
- The temporary security credentials have a limited lifetime, so you do not have to rotate them or explicitly revoke them when they're no longer needed. After temporary security credentials expire, they cannot be reused. You can specify how long the credentials are valid, up to a maximum limit.

AWS STS and AWS Regions

Temporary security credentials are generated by AWS STS. By default, AWS STS is a global service with a single endpoint at <https://sts.amazonaws.com>. However, you can also choose to make AWS STS API calls to endpoints in any other supported Region. This can reduce latency (server lag) by sending the requests to servers in a Region that is geographically closer to you. No matter which Region your credentials come from, they work globally. For more information, see [Managing AWS STS in an AWS Region \(p. 333\)](#).

Common Scenarios for Temporary Credentials

Temporary credentials are useful in scenarios that involve identity federation, delegation, cross-account access, and IAM roles.

Identity Federation

You can manage your user identities in an external system outside of AWS and grant users who sign in from those systems access to perform AWS tasks and access your AWS resources. IAM supports two types of identity federation. In both cases, the identities are stored outside of AWS. The distinction is where the external system resides—in your data center or an external third party on the web. For more information about external identity providers, see [Identity Providers and Federation \(p. 184\)](#).

- **Enterprise identity federation** – You can authenticate users in your organization's network, and then provide those users access to AWS without creating new AWS identities for them and requiring them to sign in with a separate user name and password. This is known as the *single sign-on* (SSO) approach to temporary access. AWS STS supports open standards like Security Assertion Markup Language (SAML) 2.0, with which you can use Microsoft AD FS to leverage your Microsoft Active Directory. You can also use SAML 2.0 to manage your own solution for federating user identities. For more information, see [About SAML 2.0-based Federation \(p. 190\)](#).
- **Custom federation broker** – You can use your organization's authentication system to grant access to AWS resources. For an example scenario, see [Enabling Custom Identity Broker Access to the AWS Console \(p. 213\)](#).
- **Federation using SAML 2.0** – You can use your organization's authentication system and SAML to grant access to AWS resources. For more information and an example scenario, see [About SAML 2.0-based Federation \(p. 190\)](#).
- **Web identity federation** – You can let users sign in using a well-known third party identity provider such as Login with Amazon, Facebook, Google, or any OpenID Connect (OIDC) 2.0 compatible provider. You can exchange the credentials from that provider for temporary permissions to use resources in your AWS account. This is known as the *web identity federation* approach to temporary access. When you use web identity federation for your mobile or web application, you don't need to create custom sign-in code or manage your own user identities. Using web identity federation helps you keep your AWS account secure, because you don't have to distribute long-term security credentials, such as IAM user access keys, with your application. For more information, see [About Web Identity Federation \(p. 185\)](#).

AWS STS web identity federation supports Login with Amazon, Facebook, Google, and any OpenID Connect (OIDC)-compatible identity provider.

Note

For mobile applications, we recommend that you use Amazon Cognito. You can use this service with the [AWS Mobile SDK for iOS](#) and the [AWS Mobile SDK for Android and Fire OS](#) to create unique identities for users and authenticate them for secure access to your AWS resources. Amazon Cognito supports the same identity providers as AWS STS, and also supports unauthenticated (guest) access and lets you migrate user data when a user signs in. Amazon Cognito also provides API operations for synchronizing user data so that it is preserved as users move between devices. For more information, see the following:

- [Amazon Cognito Identity in the AWS Mobile SDK for iOS Developer Guide](#)
- [Amazon Cognito Identity in the AWS Mobile SDK for Android Developer Guide](#)

Roles for Cross-account Access

Many organizations maintain more than one AWS account. Using roles and cross-account access, you can define user identities in one account, and use those identities to access AWS resources in other accounts

that belong to your organization. This is known as the *delegation* approach to temporary access. For more information about creating cross-account roles, see [Creating a Role to Delegate Permissions to an IAM User \(p. 229\)](#). To learn whether principals in accounts outside of your zone of trust (trusted organization, OU, or account) have access to assume your roles, see [What is IAM Access Analyzer?](#).

Roles for Amazon EC2

If you run applications on Amazon EC2 instances and those applications need access to AWS resources, you can provide temporary security credentials to your instances when you launch them. These temporary security credentials are available to all applications that run on the instance, so you don't need to store any long-term credentials on the instance. For more information, see [Using an IAM Role to Grant Permissions to Applications Running on Amazon EC2 Instances \(p. 269\)](#).

Other AWS Services

You can use temporary security credentials to access most AWS services. For a list of the services that accept temporary security credentials, see [AWS Services That Work with IAM \(p. 586\)](#).

Requesting Temporary Security Credentials

To request temporary security credentials, you can use AWS Security Token Service (AWS STS) operations in the AWS API. These include operations to create and provide trusted users with temporary security credentials that can control access to your AWS resources. For more information about AWS STS, see [Temporary Security Credentials \(p. 307\)](#). To learn about the different methods that you can use to request temporary security credentials by assuming a role, see [Using IAM Roles \(p. 253\)](#).

To call the API operations, you can use one of the [AWS SDKs](#). The SDKs are available for a variety of programming languages and environments, including Java, .NET, Python, Ruby, Android, and iOS. The SDKs take care of tasks such as cryptographically signing your requests, retrying requests if necessary, and handling error responses. You can also use the AWS STS Query API, which is described in the [AWS Security Token Service API Reference](#). Finally, two command line tools support the AWS STS commands: the [AWS Command Line Interface](#), and the [AWS Tools for Windows PowerShell](#).

The AWS STS API operations create a new session with temporary security credentials that include an access key pair and a session token. The access key pair consists of an access key ID and a secret key. Users (or an application that the user runs) can use these credentials to access your resources. You can create a role session and pass session policies and session tags programmatically using AWS STS API operations. The resulting session's permissions are the intersection of the role's identity-based policies and the session policies. For more information about session policies, see [Session Policies \(p. 358\)](#). For more information about session tags, see [Passing Session Tags in AWS STS \(p. 299\)](#).

Note

The size of the security token that AWS STS API operations return is not fixed. We strongly recommend that you make no assumptions about the maximum size. The typical token size is less than 4096 bytes, but that can vary.

Using AWS STS with AWS Regions

You can send AWS STS API calls either to a global endpoint or to one of the Regional endpoints. If you choose an endpoint closer to you, you can reduce latency and improve the performance of your API calls. You also can choose to direct your calls to an alternative Regional endpoint if you can no longer communicate with the original endpoint. If you are using one of the various AWS SDKs, then use that SDK's method to select a Region before you make the API call. If you are manually constructing HTTP API requests, then you must direct the request to the correct endpoint yourself. For more information, see the [AWS STS section of Regions and Endpoints](#) and [Managing AWS STS in an AWS Region \(p. 333\)](#).

The following are the API operations that you can use to acquire temporary credentials for use in your AWS environment and applications.

AssumeRole—Cross-Account Delegation and Federation Through a Custom Identity Broker

The `AssumeRole` API operation is useful for allowing existing IAM users to access AWS resources that they don't already have access to. For example, the user might need access to resources in another AWS account. It is also useful as a means to temporarily gain privileged access—for example, to provide multi-factor authentication (MFA). You must call this API using existing IAM user credentials. For more information, see [Creating a Role to Delegate Permissions to an IAM User \(p. 229\)](#) and [Configuring MFA-Protected API Access \(p. 145\)](#).

This call must be made using valid AWS security credentials. When you make this call, you pass the following information:

- The Amazon Resource Name (ARN) of the role that the app should assume.
- (Optional) Duration, which specifies the duration of the temporary security credentials. Use the `DurationSeconds` parameter to specify the duration of the role session from 900 seconds (15 minutes) up to the maximum session duration setting for the role. To learn how to view the maximum value for your role, see [View the Maximum Session Duration Setting for a Role \(p. 255\)](#). If you do not pass this parameter, the temporary credentials expire in one hour. The `DurationSeconds` parameter from this API is separate from the `SessionDuration` HTTP parameter that you use to specify the duration of a console session. Use the `SessionDuration` HTTP parameter in the request to the federation endpoint for a console sign-in token. For more information, see [Enabling Custom Identity Broker Access to the AWS Console \(p. 213\)](#).
- (Optional) Role session name, which is a string value that you can use to identify the session. This value can be captured and logged by CloudTrail to help you distinguish between your role users during an audit.
- (Optional) Inline or managed session policies. These policies limit the permissions from the role's identity-based policy that are assigned to the role session. The resulting session's permissions are the intersection of the role's identity-based policies and the session policies. Session policies cannot be used to grant more permissions than those allowed by the identity-based policy of the role that is being assumed. For more information about role session permissions, see [Session Policies \(p. 358\)](#).
- (Optional) Session tags. You can assume a role and then use the temporary credentials to make a request. When you do, the session's principal tags include the role's tags and the passed session tags. If you make this call using temporary credentials, the new session also inherits transitive session tags from the calling session. For more information about session tags, see [Passing Session Tags in AWS STS \(p. 299\)](#).
- (Optional) MFA information. If configured to use multi-factor authentication (MFA), then you include the identifier for an MFA device and the one-time code provided by that device.
- (Optional) `ExternalId` value that can be used when delegating access to your account to a third party. This value helps ensure that only the specified third party can access the role. For more information, see [How to Use an External ID When Granting Access to Your AWS Resources to a Third Party \(p. 233\)](#).

The following example shows a sample request and response using `AssumeRole`. This example request assumes the `demo` role for the specified duration with the included [session policy \(p. 358\)](#), [session tags \(p. 299\)](#), and [external ID \(p. 233\)](#). The resulting session is named `John-session`.

Example Request

```
https://sts.amazonaws.com/  
?Version=2011-06-15
```

```
&Action=AssumeRole
&RoleSessionName=John-session
&RoleArn=arn:aws::iam::123456789012:role/demo
&Policy=%7B%22Version%22%3A%222012-10-17%22%2C%22Statement%22%3A%5B%7B%22Sid%22%3A
%20%22Stmt1%22%2C%22Effect%22%3A%20%22Allow%22%2C%22Action%22%3A%20%22s3%3A*%22%2C
%22Resource%22%3A%20%22*%22%7D%5D%7D
&DurationSeconds=1800
&Tags.member.1.Key=Project
&Tags.member.1.Value=Pegasus
&Tags.member.2.Key=Cost-Center
&Tags.member.2.Value=12345
&ExternalId=123ABC
&AUTHPARAMS
```

The policy value shown in the preceding example is the URL-encoded version of the following policy:

```
{"Version": "2012-10-17", "Statement": [{"Sid": "Stmt1", "Effect": "Allow", "Action": "s3:*", "Resource": "*"}]}
```

The AUTHPARAMS parameter in the example is a placeholder for your *signature*. A signature is the authentication information that you must include with AWS HTTP API requests. We recommend using the [AWS SDKs](#) to create API requests, and one benefit of doing so is that the SDKs handle request signing for you. If you must create and sign API requests manually, see [Signing AWS Requests By Using Signature Version 4](#) in the *Amazon Web Services General Reference* to learn how to sign a request.

In addition to the temporary security credentials, the response includes the Amazon Resource Name (ARN) for the federated user and the expiration time of the credentials.

Example Response

```
<AssumeRoleResponse xmlns="https://sts.amazonaws.com/doc/2011-06-15/">
<AssumeRoleResult>
<Credentials>
<SessionToken>
AQoDYXdzEPT//////////wEXAMPLEtc764bNrC9SAPBSM22wD0k4x4HIZ8j4FZTwdQW
LWsKWHGBuFqwAeMicRXmxfpSPfIeoIYRqTflfKD8YUuwthAx7mSEI/qkPpKPi/kMcGd
QrmGdeehM4IC1NtBmUpp2wUE8phUZampKsburEDy0KPkyQDYwT7WZ0wq5VSXDvp75YU
9HFv1Rd8Tx6q6fE8YQcHNVXAkjY9q6d+xoOrKwT38xVqr7ZD0u0iPPkUL641IZbqBAz
+scqKmlzm8FDrypNC9Yjc8fPOLn9FX9KSYvKTr4rvx3iSiLTJabiQwj2ICCR/oLxBA==
</SessionToken>
<SecretAccessKey>
wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY
</SecretAccessKey>
<Expiration>2019-07-15T23:28:33.359Z</Expiration>
<AccessKeyId>AKIAIOSFODNN7EXAMPLE</AccessKeyId>
</Credentials>
<AssumedRoleUser>
<Arn>arn:aws:sts::123456789012:assumed-role/demo/John</Arn>
<AssumedRoleId>ARO123EXAMPLE123:John</AssumedRoleId>
</AssumedRoleUser>
<PackedPolicySize>8</PackedPolicySize>
</AssumeRoleResult>
<ResponseMetadata>
<RequestId>c6104cbe-af31-11e0-8154-cbc7ccf896c7</RequestId>
</ResponseMetadata>
</AssumeRoleResponse>
```

Note

An AWS conversion compresses the passed session policies and session tags into a packed binary format that has a separate limit. Your request can fail for this limit even if your plain text meets the other requirements. The `PackedPolicySize` response element indicates by percentage how close the policies and tags for your request are to the upper size limit.

AssumeRoleWithWebIdentity—Federation Through a Web-Based Identity Provider

The `AssumeRoleWithWebIdentity` API operation returns a set of temporary security credentials for federated users who are authenticated through a public identity provider. Examples of public identity providers include Login with Amazon, Facebook, Google, or any OpenID Connect (OIDC)-compatible identity provider. This operation is useful for creating mobile applications or client-based web applications that require access to AWS. Using this operation means that your users do not need their own AWS or IAM identities. For more information, see [About Web Identity Federation \(p. 185\)](#).

Instead of directly calling `AssumeRoleWithWebIdentity`, we recommend that you use Amazon Cognito and the Amazon Cognito credentials provider with the AWS SDKs for mobile development. For more information, see the following:

- [Amazon Cognito Identity](#) in the *AWS Mobile SDK for Android Developer Guide*
- [Amazon Cognito Identity](#) in the *AWS Mobile SDK for iOS Developer Guide*

If you are not using Amazon Cognito, you call the `AssumeRoleWithWebIdentity` action of AWS STS. This is an unsigned call, meaning that the app does not need to have access to any AWS security credentials to make the call. When you make this call, you pass the following information:

- The Amazon Resource Name (ARN) of the role that the app should assume. If your app supports multiple ways for users to sign in, you must define multiple roles, one per identity provider. The call to `AssumeRoleWithWebIdentity` should include the ARN of the role that is specific to the provider through which the user signed in.
- The token that the app gets from the IdP after the app authenticates the user.
- You can configure your IdP to pass attributes into your token as [session tags \(p. 299\)](#).
- (Optional) Duration, which specifies the duration of the temporary security credentials. Use the `DurationSeconds` parameter to specify the duration of the role session from 900 seconds (15 minutes) up to the maximum session duration setting for the role. To learn how to view the maximum value for your role, see [View the Maximum Session Duration Setting for a Role \(p. 255\)](#). If you do not pass this parameter, the temporary credentials expire in one hour. The `DurationSeconds` parameter from this API is separate from the `SessionDuration` HTTP parameter that you use to specify the duration of a console session. Use the `SessionDuration` HTTP parameter in the request to the federation endpoint for a console sign-in token. For more information, see [Enabling Custom Identity Broker Access to the AWS Console \(p. 213\)](#).
- (Optional) Role session name, which is a string value that you can use to identify the session. This value can be captured and logged by CloudTrail to help you distinguish between your role users during an audit.
- (Optional) Inline or managed session policies. These policies limit the permissions from the role's identity-based policy that are assigned to the role session. The resulting session's permissions are the intersection of the role's identity-based policies and the session policies. Session policies cannot be used to grant more permissions than those allowed by the identity-based policy of the role that is being assumed. For more information about role session permissions, see [Session Policies \(p. 358\)](#).

Note

A call to `AssumeRoleWithWebIdentity` is not signed (encrypted). Therefore, you should only include optional session policies if the request is transmitted through a trusted intermediary. In this case, someone could alter the policy to remove the restrictions.

When you call `AssumeRoleWithWebIdentity`, AWS verifies the authenticity of the token. For example, depending on the provider, AWS might make a call to the provider and include the token that the app has passed. Assuming that the identity provider validates the token, AWS returns the following information to you:

- A set of temporary security credentials. These consist of an access key ID, a secret access key, and a session token.
- The role ID and the ARN of the assumed role.
- A `SubjectFromWebIdentityToken` value that contains the unique user ID.

When you have the temporary security credentials, you can use them to make AWS API calls. This is the same process as making an AWS API call with long-term security credentials. The difference is that you must include the session token, which lets AWS verify that the temporary security credentials are valid.

Your app should cache the credentials. As noted, by default the credentials expire after an hour. If you are not using the [AmazonSTSCredentialsProvider](#) operation in the AWS SDK, it's up to you and your app to call `AssumeRoleWithWebIdentity` again. Call this operation to get a new set of temporary security credentials before the old ones expire.

AssumeRoleWithSAML—Federation Through an Enterprise Identity Provider Compatible with SAML 2.0

The `AssumeRoleWithSAML` API operation returns a set of temporary security credentials for federated users who are authenticated by your organization's existing identity system. The users must also use [SAML 2.0 \(Security Assertion Markup Language\)](#) to pass authentication and authorization information to AWS. This API operation is useful in organizations that have integrated their identity systems (such as Windows Active Directory or OpenLDAP) with software that can produce SAML assertions. Such an integration provides information about user identity and permissions (such as Active Directory Federation Services or Shibboleth). For more information, see [About SAML 2.0-based Federation \(p. 190\)](#).

This is an unsigned call, which means that the app does not need to have access to any AWS security credentials in order to make the call. When you make this call, you pass the following information:

- The Amazon Resource Name (ARN) of the role that the app should assume.
- The ARN of the SAML provider created in IAM that describes the identity provider.
- The SAML assertion, encoded in base64, that was provided by the SAML identity provider in its authentication response to the sign-in request from your app.
- You can configure your IdP to pass attributes into your SAML assertion as [session tags \(p. 299\)](#).
- (Optional) Duration, which specifies the duration of the temporary security credentials. Use the `DurationSeconds` parameter to specify the duration of the role session from 900 seconds (15 minutes) up to the maximum session duration setting for the role. To learn how to view the maximum value for your role, see [View the Maximum Session Duration Setting for a Role \(p. 255\)](#). If you do not pass this parameter, the temporary credentials expire in one hour. The `DurationSeconds` parameter from this API is separate from the `SessionDuration` HTTP parameter that you use to specify the duration of a console session. Use the `SessionDuration` HTTP parameter in the request to the federation endpoint for a console sign-in token. For more information, see [Enabling Custom Identity Broker Access to the AWS Console \(p. 213\)](#).
- (Optional) Inline or managed session policies. These policies limit the permissions from the role's identity-based policy that are assigned to the role session. The resulting session's permissions are the intersection of the role's identity-based policies and the session policies. Session policies cannot be used to grant more permissions than those allowed by the identity-based policy of the role that is being assumed. For more information about role session permissions, see [Session Policies \(p. 358\)](#).

When you call `AssumeRoleWithSAML`, AWS verifies the authenticity of the SAML assertion. Assuming that the identity provider validates the assertion, AWS returns the following information to you:

- A set of temporary security credentials. These consist of an access key ID, a secret access key, and a session token.

- The role ID and the ARN of the assumed role.
- An Audience value that contains the value of the Recipient attribute of the SubjectConfirmationData element of the SAML assertion.
- An Issuer value that contains the value of the Issuer element of the SAML assertion.
- A NameQualifier element that contains a hash value built from the Issuer value, the AWS account ID, and the friendly name of the SAML provider. When combined with the Subject element, they can uniquely identify the federated user.
- A Subject element that contains the value of the NameID element in the Subject element of the SAML assertion.
- A SubjectType element that indicates the format of the Subject element. The value can be persistent, transient, or the full Format URI from the Subject and NameID elements used in your SAML assertion. For information about the NameID element's Format attribute, see [Configuring SAML Assertions for the Authentication Response \(p. 206\)](#).

When you have the temporary security credentials, you can use them to make AWS API calls. This is the same process as making an AWS API call with long-term security credentials. The difference is that you must include the session token, which lets AWS verify that the temporary security credentials are valid.

Your app should cache the credentials. By default the credentials expire after an hour. If you are not using the [AmazonSTSCredentialsProvider](#) action in the AWS SDK, it's up to you and your app to call AssumeRoleWithSAML again. Call this operation to get a new set of temporary security credentials before the old ones expire.

GetFederationToken—Federation Through a Custom Identity Broker

The GetFederationToken API operation returns a set of temporary security credentials for federated users. This API differs from AssumeRole in that the default expiration period is substantially longer (12 hours instead of one hour). Additionally, you can use the DurationSeconds parameter to specify a duration for the temporary security credentials to remain valid. The resulting credentials are valid for the specified duration, between 900 seconds (15 minutes) to 129,600 seconds (36 hours). The longer expiration period can help reduce the number of calls to AWS because you do not need to get new credentials as often. For more information, see [Requesting Temporary Security Credentials \(p. 309\)](#).

When you make this request, you use the credentials of a specific IAM user. The permissions for the temporary security credentials are determined by the session policies that you pass when you call GetFederationToken. The resulting session permissions are the intersection of the IAM user policies and the session policies that you pass. Session policies cannot be used to grant more permissions than those allowed by the identity-based policy of the IAM user that is requesting federation. For more information about role session permissions, see [Session Policies \(p. 358\)](#).

When you use the temporary credentials that are returned by the GetFederationToken operation, the session's principal tags include the user's tags and the passed session tags. For more information about session tags, see [Passing Session Tags in AWS STS \(p. 299\)](#).

The GetFederationToken call returns temporary security credentials that consist of the security token, access key, secret key, and expiration. You can use GetFederationToken if you want to manage permissions inside your organization (for example, using the proxy application to assign permissions). To view a sample application that uses GetFederationToken, go to [Identity Federation Sample Application for an Active Directory Use Case](#) in the AWS Sample Code & Libraries.

The following example shows a sample request and response that uses GetFederationToken. This example request federates the calling user for the specified duration with the [session policy \(p. 358\)](#) ARN and [session tags \(p. 299\)](#). The resulting session is named Jane-session.

Example Request

```
https://sts.amazonaws.com/  
?Version=2011-06-15  
&Action=GetFederationToken  
&Name=Jane-session  
&PolicyArns.member.1.arn==arn%3Aaws%3Aiam%3A123456789012%3Apolicy%2FRole1policy  
&DurationSeconds=1800  
&Tags.member.1.Key=Project  
&Tags.member.1.Value=Pegasus  
&Tags.member.2.Key=Cost-Center  
&Tags.member.2.Value=12345  
&AUTHPARAMS
```

The policy ARN shown in the preceding example includes the following URL-encoded ARN:

```
arn:aws:iam::123456789012:policy/Role1policy
```

Also, note that the &AUTHPARAMS parameter in the example is meant as a placeholder for the authentication information. This is the *signature*, which you must include with AWS HTTP API requests. We recommend using the [AWS SDKs](#) to create API requests, and one benefit of doing so is that the SDKs handle request signing for you. If you must create and sign API requests manually, go to [Signing AWS Requests By Using Signature Version 4](#) in the *Amazon Web Services General Reference* to learn how to sign a request.

In addition to the temporary security credentials, the response includes the Amazon Resource Name (ARN) for the federated user and the expiration time of the credentials.

Example Response

```
<GetFederationTokenResponse xmlns="https://sts.amazonaws.com/doc/2011-06-15/">  
<GetFederationTokenResult>  
<Credentials>  
<SessionToken>  
AQoDYXzdEPt//////////wEXAMPLEtc764bNrC9SAPBSM22wD0k4x4HIZ8j4FZTwdQW  
LWsKWHGBuFgwAeMicRXmxfpSPFleoiYRqTflfKD8YUuwhAx7mSEI/qkPpKPi/kMcGd  
QrmGdeehM4IC1NtBmUpp2wUE8phUZampKsburEDy0KPkyQDYwT7WZ0wq5VSxDvp75YU  
9HFv1Rd8Tx6q6fE8YQcHNVXAkiv9q6d+xo0rKwT38xVqr7ZD0u0iPPkUL641IZbqBAz  
+scqKmlzm8FDdrypNC9Yjc8fPOLn9FX9KSYvKTr4rvx3iSiLTJabiQwj2ICCEXAMPLE==  
</SessionToken>  
<SecretAccessKey>  
wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY  
</SecretAccessKey>  
<Expiration>2019-04-15T23:28:33.359Z</Expiration>  
<AccessKeyId>AKIAIOSFODNN7EXAMPLE;</AccessKeyId>  
</Credentials>  
<FederatedUser>  
<Arn>arn:aws:sts::123456789012:federated-user/Jean</Arn>  
<FederatedUserId>123456789012:Jean</FederatedUserId>  
</FederatedUser>  
<PackedPolicySize>4</PackedPolicySize>  
</GetFederationTokenResult>  
<ResponseMetadata>  
<RequestId>c6104cbe-af31-11e0-8154-cbc7ccf896c7</RequestId>  
</ResponseMetadata>  
</GetFederationTokenResponse>
```

Note

An AWS conversion compresses the passed session policies and session tags into a packed binary format that has a separate limit. Your request can fail for this limit even if your plain text meets the other requirements. The `PackedPolicySize` response element indicates by percentage how close the policies and tags for your request are to the upper size limit.

AWS recommends that you grant permissions at the resource level (for example, you attach a resource-based policy to an Amazon S3 bucket), you can omit the `Policy` parameter. However, if you do not include a policy for the federated user, the temporary security credentials will not grant any permissions. In this case, you *must* use resource policies to grant the federated user access to your AWS resources.

For example, assume your AWS account number is 111122223333, and you have an Amazon S3 bucket that you want to allow Susan to access. Susan's temporary security credentials don't include a policy for the bucket. In that case, you would need to ensure that the bucket has a policy with an ARN that matches Susan's ARN, such as `arn:aws:sts::111122223333:federated-user/Susan`.

GetSessionToken—Temporary Credentials for Users in Untrusted Environments

The `GetSessionToken` API operation returns a set of temporary security credentials to an existing IAM user. This is useful for providing enhanced security, such as allowing AWS requests only when MFA is enabled for the IAM user. Because the credentials are temporary, they provide enhanced security when you have an IAM user who accesses your resources through a less secure environment. Examples of less secure environments include a mobile device or web browser. For more information, see [Requesting Temporary Security Credentials \(p. 309\)](#) or [GetSessionToken](#) in the *AWS Security Token Service API Reference*.

By default, temporary security credentials for an IAM user are valid for a maximum of 12 hours. But you can request a duration as short as 15 minutes or as long as 36 hours using the `DurationSeconds` parameter. For security reasons, a token for an AWS account root user is restricted to a duration of one hour.

`GetSessionToken` returns temporary security credentials consisting of a security token, an access key ID, and a secret access key. The following example shows a sample request and response using `GetSessionToken`. The response also includes the expiration time of the temporary security credentials.

Example Request

```
https://sts.amazonaws.com/  
?Version=2011-06-15  
&Action=GetSessionToken  
&DurationSeconds=1800  
&AUTHPARAMS
```

The `AUTHPARAMS` parameter in the example is a placeholder for your *signature*. A signature is the authentication information that you must include with AWS HTTP API requests. We recommend using the [AWS SDKs](#) to create API requests, and one benefit of doing so is that the SDKs handle request signing for you. If you must create and sign API requests manually, go to [Signing AWS Requests By Using Signature Version 4](#) in the *Amazon Web Services General Reference* to learn how to sign a request.

Example Response

```
<GetSessionTokenResponse xmlns="https://sts.amazonaws.com/doc/2011-06-15/">  
<GetSessionTokenResult>  
<Credentials>  
<SessionToken>  
AQoEXAMPLE4aoAH0gNCAPyJxz4BlCFFxWNE1OPTgk5TthT+FvwqnKwRcOIfrrRh3c/L  
To6UDdyJwOovEPVlxCrUtdnniCEXAMPLE/IvU1dYUg2RVAJBanLiHb4IgRmpRV3z  
rkuWJOgQs8IZZaIv2BXIa2R40lgkBN9bkUDNCJiBeb/AXlzBBko7b15fjrBs2+cTQtp  
Z3CYWFXG8C5zqx37wnOE49mR1/+OtkIKG07fAE  
</SessionToken>  
<SecretAccessKey>  
wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY  
</SecretAccessKey>
```

```

<Expiration>2011-07-11T19:55:29.611Z</Expiration>
<AccessKeyId>AKIAIOSFODNN7EXAMPLE</AccessKeyId>
</Credentials>
</GetSessionTokenResult>
<ResponseMetadata>
<RequestId>58c5dbae-abef-11e0-8cf8-09039844ac7d</RequestId>
</ResponseMetadata>
</GetSessionTokenResponse>

```

Optionally, the `GetSessionToken` request can include `SerialNumber` and `TokenCode` values for AWS multi-factor authentication (MFA) verification. If the provided values are valid, AWS STS provides temporary security credentials that include the state of MFA authentication. The temporary security credentials can then be used to access the MFA-protected API operations or AWS websites for as long as the MFA authentication is valid.

The following example shows a `GetSessionToken` request that includes an MFA verification code and device serial number.

```

https://sts.amazonaws.com/
?Version=2011-06-15
&Action=GetSessionToken
&DurationSeconds=7200
&SerialNumber=YourMFADeviceSerialNumber
&TokenCode=123456
&AUTHPARAMS

```

Note

The call to AWS STS can be to the global endpoint or to any of the Regional endpoints that you activate your AWS account. For more information, see the [AWS STS section of Regions and Endpoints](#).

The `AUTHPARAMS` parameter in the example is a placeholder for your *signature*. A signature is the authentication information that you must include with AWS HTTP API requests. We recommend using the [AWS SDKs](#) to create API requests, and one benefit of doing so is that the SDKs handle request signing for you. If you must create and sign API requests manually, see [Signing AWS Requests By Using Signature Version 4](#) in the *Amazon Web Services General Reference* to learn how to sign a request.

Comparing the AWS STS API Operations

The following table compares features of the API operations in AWS STS that return temporary security credentials. To learn about the different methods you can use to request temporary security credentials by assuming a role, see [Using IAM Roles \(p. 253\)](#). To learn about the different AWS STS API operations that allow you to pass session tags, see [Passing Session Tags in AWS STS \(p. 299\)](#).

Comparing Your API Options

AWS STS API	Who Can Call	Credential Lifetime (Min Max Default)	MFA Support ¹	Session Policy Support ²	Restrictions on Resulting Temporary Credentials
<code>AssumeRole</code>	IAM user or IAM role with existing temporary security credentials	15 m Maximum session duration setting ³ 1 hr	Yes	Yes	Cannot call <code>GetFederationToken</code> or <code>GetSessionToken</code> .

AWS STS API	Who Can Call	Credential Lifetime (Min Max Default)	MFA Support ¹	Session Policy Support ²	Restrictions on Resulting Temporary Credentials
AssumeRole	Any user; caller must pass a SAML authentication response that indicates authentication from a known identity provider	15 m Maximum session duration setting ³ 1 hr	No	Yes	Cannot call GetFederationToken or GetSessionToken.
AssumeRoleWithWebIdentity	Any user; caller must pass a web identity token that indicates authentication from a known identity provider	15 m Maximum session duration setting ³ 1 hr	No	Yes	Cannot call GetFederationToken or GetSessionToken.
GetFederationToken	IAM user or AWS account root user	IAM user: 15 m 36 hr 12 hr Root user: 15 m 1 hr 1 hr	No	Yes	Cannot call IAM operations using the AWS CLI or AWS API. Cannot call AWS STS operations except GetCallerIdentity. ⁴ SSO to console is allowed. ⁵
GetSessionToken	IAM user or AWS account root user	IAM user: 15 m 36 hr 12 hr Root user: 15 m 1 hr 1 hr	Yes	No	Cannot call IAM API operations unless MFA information is included with the request. Cannot call AWS STS API operations except AssumeRole or GetCallerIdentity. SSO to console is not allowed. ⁶

¹ MFA support. You can include information about a multi-factor authentication (MFA) device when you call the AssumeRole and GetSessionToken API operations. This ensures that the temporary security credentials that result from the API call can be used only by users who are authenticated with an MFA device. For more information, see [Configuring MFA-Protected API Access \(p. 145\)](#).

² Session policy support. Session policies are policies that you pass as a parameter when you programmatically create a temporary session for a role or federated user. This policy limits the permissions from the role or user's identity-based policy that are assigned to the session. The resulting session's permissions are the intersection of the entity's identity-based policies and the session policies. Session policies cannot be used to grant more permissions than those allowed by the identity-based policy of the role that is being assumed. For more information about role session permissions, see [Session Policies \(p. 358\)](#).

³ Maximum session duration setting. Use the DurationSeconds parameter to specify the duration of your role session from 900 seconds (15 minutes) up to the maximum session duration setting for the role. To learn how to view the maximum value for your role, see [View the Maximum Session Duration Setting for a Role \(p. 255\)](#).

⁴ **GetCallerIdentity.** No permissions are required to perform this operation. If an administrator adds a policy to your IAM user or role that explicitly denies access to the `sts:GetCallerIdentity` action, you can still perform this operation. Permissions are not required because the same information is returned when an IAM user or role is denied access. To view an example response, see [I Am Not Authorized to Perform: iam>DeleteVirtualMFADevice \(p. 547\)](#).

⁵ **Single sign-on (SSO) to the console.** To support SSO, AWS lets you call a federation endpoint (`https://signin.aws.amazon.com/federation`) and pass temporary security credentials. The endpoint returns a token that you can use to construct a URL that signs a user directly into the console without requiring a password. For more information, see [Enabling SAML 2.0 Federated Users to Access the AWS Management Console \(p. 211\)](#) and [How to Enable Cross-Account Access to the AWS Management Console](#) in the AWS Security Blog.

⁶ After you retrieve your temporary credentials, you can't access the AWS Management Console by passing the credentials to the federation single sign-on endpoint. For more information, see [Enabling Custom Identity Broker Access to the AWS Console \(p. 213\)](#).

Using Temporary Credentials With AWS Resources

You can use temporary security credentials to make programmatic requests for AWS resources using the AWS CLI or AWS API (using the [AWS SDKs](#)). The temporary credentials provide the same permissions that you have with use long-term security credentials such as IAM user credentials. However, there are a few differences:

- When you make a call using temporary security credentials, the call must include a session token, which is returned along with those temporary credentials. AWS uses the session token to validate the temporary security credentials.
- The temporary credentials expire after a specified interval. After the credentials expire, any calls that you make with those credentials will fail, so you must get a new set of credentials.
- When you use temporary credentials to make a request, your principal might include a set of tags. These tags come from session tags and tags that are attached to the role that you assume. For more information about session tags, see [Passing Session Tags in AWS STS \(p. 299\)](#).

If you are using the [AWS SDKs](#), the [AWS Command Line Interface](#) (AWS CLI), or the [Tools for Windows PowerShell](#), the way to get and use temporary security credentials differs with the context. If you are running code, AWS CLI, or Tools for Windows PowerShell commands inside an EC2 instance, you can take advantage of roles for Amazon EC2. Otherwise, you can call an [AWS STS API](#) to get the temporary credentials, and then use them explicitly to make calls to AWS services.

Note

You can use AWS Security Token Service (AWS STS) to create and provide trusted users with temporary security credentials that can control access to your AWS resources. For more information about AWS STS, see [Temporary Security Credentials \(p. 307\)](#). AWS STS is a global service that has a default endpoint at `https://sts.amazonaws.com`. This endpoint is in the US East (Ohio) Region, although credentials that you get from this and other endpoints are valid globally. These credentials work with services and resources in any Region. You can also choose to make AWS STS API calls to endpoints in any of the supported Regions. This can reduce latency by making the requests from servers in a Region that is geographically closer to you. No matter which Region your credentials come from, they work globally. For more information, see [Managing AWS STS in an AWS Region \(p. 333\)](#).

Contents

- [Using Temporary Credentials in Amazon EC2 Instances \(p. 320\)](#)
- [Using Temporary Security Credentials with the AWS SDKs \(p. 320\)](#)
- [Using Temporary Security Credentials with the AWS CLI \(p. 320\)](#)
- [Using Temporary Security Credentials with API Operations \(p. 321\)](#)

- [More Information \(p. 321\)](#)

Using Temporary Credentials in Amazon EC2 Instances

If you want to run AWS CLI commands or code inside an EC2 instance, the recommended way to get credentials is to use [roles for Amazon EC2](#). You create an IAM role that specifies the permissions that you want to grant to applications that run on the EC2 instances. When you launch the instance, you associate the role with the instance.

Applications, AWS CLI, and Tools for Windows PowerShell commands that run on the instance can then get automatic temporary security credentials from the instance metadata. You do not have to explicitly get the temporary security credentials. The AWS SDKs, AWS CLI, and Tools for Windows PowerShell automatically get the credentials from the EC2 instance metadata service and use them. The temporary credentials have the permissions that you define for the role that is associated with the instance.

For more information and for examples, see the following:

- [Using IAM Roles to Grant Access to AWS Resources on Amazon Elastic Compute Cloud](#) — AWS SDK for Java
- [Granting Access Using an IAM Role](#) — AWS SDK for .NET
- [Creating a Role](#) — AWS SDK for Ruby

Using Temporary Security Credentials with the AWS SDKs

To use temporary security credentials in code, you programmatically call an AWS STS API like `AssumeRole` and extract the resulting credentials and session token. You then use those values as credentials for subsequent calls to AWS. The following example shows pseudocode for how to use temporary security credentials if you're using an AWS SDK:

```
assumeRoleResult = AssumeRole(role-arn);
tempCredentials = new SessionAWSCredentials(
    assumeRoleResult.AccessKeyId,
    assumeRoleResult.SecretAccessKey,
    assumeRoleResult.SessionToken);
s3Request = CreateAmazonS3Client(tempCredentials);
```

For an example written in Python (using the [AWS SDK for Python \(Boto\)](#)), see [Switching to an IAM Role \(AWS API\) \(p. 268\)](#). This example shows how to call `AssumeRole` to get temporary security credentials and then use those credentials to make a call to Amazon S3.

For details about how to call `AssumeRole`, `GetFederationToken`, and other API operations, see the [AWS Security Token Service API Reference](#). For information on getting the temporary security credentials and session token from the result, see the documentation for the SDK that you're working with. You can find the documentation for all the AWS SDKs on the main [AWS documentation page](#), in the **SDKs and Toolkits** section.

You must make sure that you get a new set of credentials before the old ones expire. In some SDKs, you can use a provider that manages the process of refreshing credentials for you; check the documentation for the SDK you're using.

Using Temporary Security Credentials with the AWS CLI

You can use temporary security credentials with the AWS CLI. This can be useful for testing policies.

Using the [AWS CLI](#), you can call an [AWS STS API](#) like `AssumeRole` or `GetFederationToken` and then capture the resulting output. The following example shows a call to `AssumeRole` that sends the output

to a file. In the example, the `profile` parameter is assumed to be a profile in the AWS CLI configuration file. It is also assumed to reference credentials for an IAM user who has permissions to assume the role.

```
$ aws sts assume-role --role-arn arn:aws:iam::123456789012:role/role-name --role-session-name "RoleSession1" --profile IAM-user-name > assume-role-output.txt
```

When the command is finished, you can extract the access key ID, secret access key, and session token from wherever you've routed it. You can do this either manually or by using a script. You can then assign these values to environment variables.

When you run AWS CLI commands, the AWS CLI looks for credentials in a specific order—first in environment variables and then in the configuration file. Therefore, after you've put the temporary credentials into environment variables, the AWS CLI uses those credentials by default. (If you specify a `profile` parameter in the command, the AWS CLI skips the environment variables. Instead, the AWS CLI looks in the configuration file, which lets you override the credentials in the environment variables if you need to.)

The following example shows how you might set the environment variables for temporary security credentials and then call an AWS CLI command. Because no `profile` parameter is included in the AWS CLI command, the AWS CLI looks for credentials first in environment variables and therefore uses the temporary credentials.

Linux

```
$ export AWS_ACCESS_KEY_ID=AKIAI44QH8DHBEEXAMPLE
$ export AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
$ export AWS_SESSION_TOKEN=AQoDYXdzEJr...<remainder of security token>
$ aws ec2 describe-instances --region us-west-1
```

Windows

```
C:\> SET AWS_ACCESS_KEY_ID=AKIAI44QH8DHBEEXAMPLE
C:\> SET AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
C:\> SET AWS_SESSION_TOKEN=AQoDYXdzEJr...<remainder of token>
C:\> aws ec2 describe-instances --region us-west-1
```

Using Temporary Security Credentials with API Operations

If you're making direct HTTPS API requests to AWS, you can sign those requests with the temporary security credentials that you get from the AWS Security Token Service (AWS STS). To do this, you use the access key ID and secret access key that you receive from AWS STS. You use the access key ID and secret access key the same way you would use long-term credentials to sign a request. You also add to your API request the session token that you receive from AWS STS. You add the session token to an HTTP header or to a query string parameter named `X-Amz-Security-Token`. You add the session token to the HTTP header or the query string parameter, but not both. For more information about signing HTTPS API requests, see [Signing AWS API Requests](#) in the [AWS General Reference](#).

More Information

For more information about using AWS STS with other AWS services, see the following links:

- **Amazon S3.** See [Making Requests Using IAM User Temporary Credentials](#) or [Making Requests Using Federated User Temporary Credentials](#) in the *Amazon Simple Storage Service Developer Guide*.
- **Amazon SNS.** See [Using Temporary Security Credentials](#) in the *Amazon Simple Notification Service Developer Guide*.
- **Amazon SQS.** See [Using Temporary Security Credentials](#) in the *Amazon Simple Queue Service Developer Guide*.

- **Amazon SimpleDB.** See [Using Temporary Security Credentials](#) in the *Amazon SimpleDB Developer Guide*.

Controlling Permissions for Temporary Security Credentials

You can use AWS Security Token Service (AWS STS) to create and provide trusted users with temporary security credentials that can control access to your AWS resources. For more information about AWS STS, see [Temporary Security Credentials \(p. 307\)](#). After AWS STS issues temporary security credentials, they are valid through the expiration period and cannot be revoked. However, the permissions assigned to temporary security credentials are evaluated each time a request is made that uses the credentials, so you can achieve the effect of revoking the credentials by changing their access rights after they have been issued.

The following topics assume you have a working knowledge of AWS permissions and policies. For more information on these topics, see [Access Management \(p. 355\)](#).

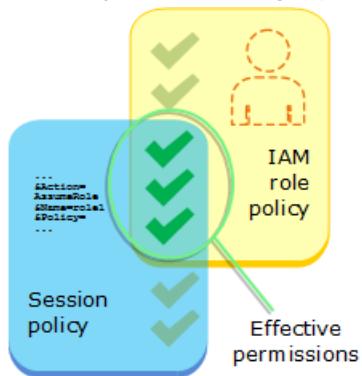
Topics

- [Permissions for AssumeRole, AssumeRoleWithSAML, and AssumeRoleWithWebIdentity \(p. 322\)](#)
- [Permissions for GetFederationToken \(p. 324\)](#)
- [Permissions for GetSessionToken \(p. 327\)](#)
- [Disabling Permissions for Temporary Security Credentials \(p. 329\)](#)
- [Granting Permissions to Create Temporary Security Credentials \(p. 331\)](#)

Permissions for AssumeRole, AssumeRoleWithSAML, and AssumeRoleWithWebIdentity

The permissions policy of the role that is being assumed determines the permissions for the temporary security credentials that are returned by `AssumeRole`, `AssumeRoleWithSAML`, and `AssumeRoleWithWebIdentity`. You define these permissions when you create or update the role.

Optionally, you can pass inline or managed [session policies \(p. 358\)](#) as parameters of the `AssumeRole`, `AssumeRoleWithSAML`, or `AssumeRoleWithWebIdentity` API operations. Session policies limit the permissions for the role's temporary credential session. The resulting session's permissions are the intersection of the role's identity-based policy and the session policies. You can use the role's temporary credentials in subsequent AWS API calls to access resources in the account that owns the role. You cannot use session policies to grant more permissions than those allowed by the identity-based policy of the role that is being assumed. To learn more about how AWS determines the effective permissions of a role, see [Policy Evaluation Logic \(p. 636\)](#).



The policies that are attached to the credentials that made the original call to `AssumeRole` are not evaluated by AWS when making the "allow" or "deny" authorization decision. The user temporarily gives up its original permissions in favor of the permissions assigned by the assumed role. In the case of the `AssumeRoleWithSAML` and `AssumeRoleWithWebIdentity` API operations, there are no policies to evaluate because the caller of the API is not an AWS identity.

Example: Assigning Permissions Using AssumeRole

You can use the `AssumeRole` API operation with different kinds of policies. Here are a few examples.

Role Permissions Policy

In this example, you call the `AssumeRole` API operation without specifying the session policy in the optional `Policy` parameter. The permissions assigned to the temporary credentials are determined by the permissions policy of the role being assumed. The following example permissions policy grants the role permission to list all objects that are contained in an S3 bucket named `productionapp`. It also allows the role to get, put, and delete objects within that bucket.

Example Role Permissions Policy

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3>ListBucket",
      "Resource": "arn:aws:s3:::productionapp"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject",
        "s3>DeleteObject"
      ],
      "Resource": "arn:aws:s3:::productionapp/*"
    }
  ]
}
```

Session Policy Passed as a Parameter

Imagine that you want to allow a user to assume the same role as in the previous example. But in this case you want the role session to have permission only to get and put objects in the `productionapp` S3 bucket. You do not want to allow them to delete objects. One way to accomplish this is to create a new role and specify the desired permissions in that role's permissions policy. Another way to accomplish this is to call the `AssumeRole` API and include session policies in the optional `Policy` parameter as part of the API operation. The resulting session's permissions are the intersection of the role's identity-based policies and the session policies. Session policies cannot be used to grant more permissions than those allowed by the identity-based policy of the role that is being assumed. For more information about role session permissions, see [Session Policies \(p. 358\)](#).

After you retrieve the new session's temporary credentials, you can pass them to the user that you want to have those permissions.

For example, imagine that the following policy is passed as a parameter of the API call. The person using the session has permissions to perform only these actions:

- List all objects in the `productionapp` bucket.

- Get and put objects in the productionapp bucket.

In the following session policy, the `s3>DeleteObject` permission is filtered out and the assumed session is not granted the `s3>DeleteObject` permission. The policy sets the maximum permissions for the role session so that it overrides any existing permissions policies on the role.

Example Session Policy Passed with `AssumeRole` API call

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3>ListBucket",
      "Resource": "arn:aws:s3:::productionapp"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": "arn:aws:s3:::productionapp/*"
    }
  ]
}
```

Resource-Based Policy

Some AWS resources support resource-based policies, and these policies provide another mechanism to define permissions that affect temporary security credentials. Only a few resources, like Amazon S3 buckets, Amazon SNS topics, and Amazon SQS queues support resource-based policies. The following example expands on the previous examples, using an S3 bucket named `productionapp`. The following policy is attached to the bucket.

When you attach the following resource-based policy to the `productionapp` bucket, *all* users are denied permission to delete objects from the bucket. (See the `Principal` element in the policy.) This includes all assumed role users, even though the role permissions policy grants the `DeleteObject` permission. An explicit `Deny` statement always takes precedence over an `Allow` statement.

Example Bucket Policy

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Principal": {"AWS": "*"},
      "Effect": "Deny",
      "Action": "s3>DeleteObject",
      "Resource": "arn:aws:s3:::productionapp/*"
    }
  ]
}
```

For more information about how multiple policy types are combined and evaluated by AWS, see [Policy Evaluation Logic \(p. 636\)](#).

Permissions for `GetFederationToken`

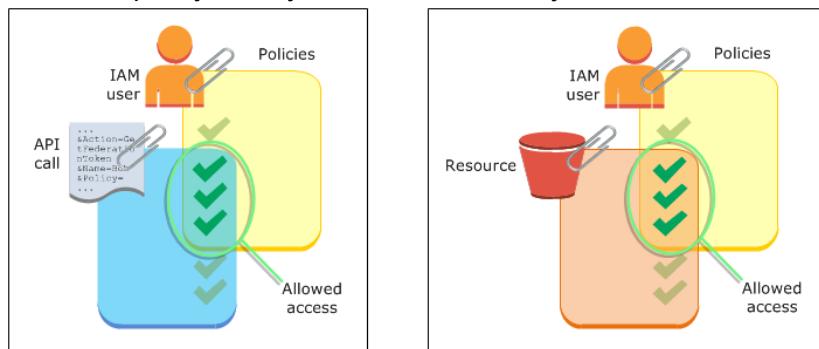
The `GetFederationToken` operation is called by an IAM user and returns temporary credentials for that user. This operation *federates* the user. The permissions assigned a federated user are defined in one of two places:

- The session policies passed as a parameter of the `GetFederationToken` API call. (This is most common.)
- A resource-based policy that explicitly names the federated user in the `Principal` element of the policy. (This is less common.)

Session policies are advanced policies that you pass as parameters when you programmatically create a temporary session. When you create a federated user session and pass session policies, the resulting session's permissions are the intersection of the IAM user's identity-based policy and the session policies. You cannot use the session policy to grant more permissions than those allowed by the identity-based policy of the user that is being federated.

In most cases if you do not pass a policy with the `GetFederationToken` API call, the resulting temporary security credentials have no permissions. However, a resource-based policy can provide additional permissions for the session. You can access a resource with a resource-based policy that specifies your session as the allowed principal.

The following figures show a visual representation of how the policies interact to determine permissions for the temporary security credentials returned by a call to `GetFederationToken`.



Example: Assigning Permissions Using `GetFederationToken`

You can use the `GetFederationToken` API action with different kinds of policies. Here are a few examples.

Policy Attached to the IAM User

In this example, you have a browser-based client application that relies on two backend web services. One backend service is your own authentication server that uses your own identity system to authenticate the client application. The other backend service is an AWS service that provides some of the client application's functionality. The client application is authenticated by your server, and your server creates or retrieves the appropriate permissions policy. Your server then calls the `GetFederationToken` API to obtain temporary security credentials, and returns those credentials to the client application. The client application can then make requests directly to the AWS service with the temporary security credentials. This architecture allows the client application to make AWS requests without embedding long-term AWS credentials.

Your authentication server calls the `GetFederationToken` API with the long-term security credentials of an IAM user named `token-app`. But the long-term IAM user credentials remain on your server and are never distributed to the client. The following example policy is attached to the `token-app` IAM user and defines the broadest set of permissions that your federated users (clients) will need. Note that the `sts:GetFederationToken` permission is required for your authentication service to obtain temporary security credentials for the federated users.

Note

AWS provides a sample Java application to serve this purpose, which you can download here: [Token Vending Machine for Identity Registration - Sample Java Web Application](#).

Example Policy Attached to IAM User token-app that Calls `GetFederationToken`

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "sts:GetFederationToken",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "dynamodb>ListTables",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "sns:ReceiveMessage",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "s3>ListBucket",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "sns>ListSubscriptions",
      "Resource": "*"
    }
  ]
}
```

The preceding policy grants several permissions to the IAM user. However, this policy alone doesn't grant any permissions to the federated user. If this IAM user calls `GetFederationToken` and does not pass a policy as a parameter of the API call, the resulting federated user has no effective permissions.

Session Policy Passed as Parameter

The most common way to ensure that the federated user is assigned appropriate permission is to pass session policies in the `GetFederationToken` API call. Expanding on the previous example, imagine that `GetFederationToken` is called with the credentials of the IAM user token-app. Then imagine that the following session policy is passed as a parameter of the API call. The resulting federated user has permission to list the contents of the Amazon S3 bucket named `productionapp`. The user can't perform the Amazon S3 `GetObject`, `PutObject`, and `DeleteObject` actions on items in the `productionapp` bucket.

The federated user is assigned these permissions because the permissions are the intersection of the IAM user policies and the session policies that you pass.

The federated user could not perform actions in Amazon SNS, Amazon SQS, Amazon DynamoDB, or in any S3 bucket except `productionapp`. These actions are denied even though those permissions are granted to the IAM user that is associated with the `GetFederationToken` call.

Example Session Policy Passed as Parameter of `GetFederationToken` API call

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["s3>ListBucket"]
    }
  ]
}
```

```

        "Resource": ["arn:aws:s3:::productionapp"]
    },
    {
        "Effect": "Allow",
        "Action": [
            "s3:GetObject",
            "s3:PutObject",
            "s3:DeleteObject"
        ],
        "Resource": ["arn:aws:s3:::productionapp/*"]
    }
]
}

```

Resource-Based Policies

Some AWS resources support resource-based policies, and these policies provide another mechanism to grant permissions directly to a federated user. Only some AWS services support resource-based policies. For example, Amazon S3 has buckets, Amazon SNS has topics, and Amazon SQS has queues that you can attach policies to. For a list of all services that support resource-based policies, see [AWS Services That Work with IAM \(p. 586\)](#) and review the "Resource-based policies" column of the tables. You can use resource-based policies to assign permissions directly to a federated user. Do this by specifying the Amazon Resource Name (ARN) of the federated user in the `Principal` element of the resource-based policy. The following example illustrates this and expands on the previous examples, using an S3 bucket named `productionapp`.

The following resource-based policy is attached to the bucket. This bucket policy allows a federated user named Carol to access the bucket. When the example policy described earlier is attached to the `token-app` IAM user, the federated user named Carol has permission to perform the `s3:GetObject`, `s3:PutObject`, and `s3:DeleteObject` actions on the bucket named `productionapp`. This is true even when no session policy is passed as a parameter of the `GetFederationToken` API call. That's because in this case the federated user named Carol has been explicitly granted permissions by the following resource-based policy.

Remember, a federated user is granted permissions only when those permissions are explicitly granted to both the IAM user **and** the federated user. Permissions can be granted to the federated user by the session policy passed as a parameter of the `GetFederationToken` API call. They can also be granted by a resource-based policy that explicitly names the federated user in the `Principal` element of the policy, as in the following example.

Example Bucket Policy that Allows Access to Federated User

```

{
    "Version": "2012-10-17",
    "Statement": {
        "Principal": {"AWS": "arn:aws:sts::ACCOUNT-ID-WITHOUT-HYPHENS:federated-user/Carol"},
        "Effect": "Allow",
        "Action": [
            "s3:GetObject",
            "s3:PutObject",
            "s3:DeleteObject"
        ],
        "Resource": ["arn:aws:s3:::productionapp/*"]
    }
}

```

Permissions for GetSessionToken

The primary occasion for calling the `GetSessionToken` API operation or the `get-session-token` CLI command is when a user must be authenticated with multi-factor authentication (MFA). It is possible to

write a policy that allows certain actions only when those actions are requested by a user who has been authenticated with MFA. In order to successfully pass the MFA authorization check, a user must first call `GetSessionToken` and include the optional `SerialNumber` and `TokenCode` parameters. If the user is successfully authenticated with an MFA device, the credentials returned by the `GetSessionToken` API operation include the MFA context. This context indicates that the user is authenticated with MFA and is authorized for API operations that require MFA authentication.

Permissions Required for `GetSessionToken`

No permissions are required for a user to get a session token. The purpose of the `GetSessionToken` operation is to authenticate the user using MFA. You cannot use policies to control authentication operations.

To grant permissions to perform most AWS operations, you add the action with the same name to a policy. For example, to create a user, you must use the `CreateUser` API operation, the `create-user` CLI command, or the AWS Management Console. To perform these operations, you must have a policy that allows you to access the `CreateUser` action.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": "iam:CreateUser",  
            "Resource": "*"  
        }  
    ]  
}
```

You can include the `GetSessionToken` action in your policies, but it has no effect on a user's ability to perform the `GetSessionToken` operation.

Permissions Granted by `GetSessionToken`

If `GetSessionToken` is called with the credentials of an IAM user, the temporary security credentials have the same permissions as the IAM user. Similarly, if `GetSessionToken` is called with AWS account root user credentials, the temporary security credentials have root user permissions.

Note

We recommend that you do not call `GetSessionToken` with root user credentials. Instead, follow our [best practices \(p. 60\)](#) and create IAM users with the permissions they need. Then use these IAM users for everyday interaction with AWS.

The temporary credentials that you get when you call `GetSessionToken` have the following capabilities and limitations:

- You can use the credentials to access the AWS Management Console by passing the credentials to the federation single sign-on endpoint at <https://signin.aws.amazon.com/federation>. For more information, see [Enabling Custom Identity Broker Access to the AWS Console \(p. 213\)](#).
- You **cannot** use the credentials to call IAM or AWS STS API operations. You **can** use them to call API operations for other AWS services.

Compare this API operation and its limitations and capability with the other API operations that create temporary security credentials at [Comparing the AWS STS API Operations \(p. 317\)](#)

For more information about MFA-protected API access using `GetSessionToken`, see [Configuring MFA-Protected API Access \(p. 145\)](#).

Disabling Permissions for Temporary Security Credentials

Temporary security credentials are valid until they expire, and they cannot be revoked. However, because permissions are evaluated each time an AWS request is made using the credentials, you can achieve the effect of revoking the credentials by changing the permissions for the credentials even after they have been issued. If you remove all permissions from the temporary security credentials, subsequent AWS requests that use those credentials will fail. The mechanisms for changing or removing the permissions assigned to temporary security credentials are explained in the following sections.

Note

When you update existing policy permissions, or when you apply a new policy to a user or a resource, it may take a few minutes for policy updates to take effect.

Topics

- [Denying Access to the Creator of the Temporary Security Credentials \(p. 329\)](#)
- [Denying Access to Temporary Security Credentials by Name \(p. 330\)](#)
- [Denying Access to Temporary Security Credentials Issued Before a Specific Time \(p. 331\)](#)

Denying Access to the Creator of the Temporary Security Credentials

To change or remove the permissions assigned to temporary security credentials, you can change or remove the permissions that are associated with the creator of the credentials. The creator of the credentials is determined by the AWS STS API that was used to obtain the credentials. The mechanisms for changing or removing the permissions associated with this creator are explained in the following sections.

Denying Access to Credentials Created by AssumeRole, AssumeRoleWithSAML, or AssumeRoleWithWebIdentity

To change or remove the permissions assigned to the temporary security credentials obtained by calling the `AssumeRole`, `AssumeRoleWithSAML`, or `AssumeRoleWithWebIdentity` API operations, you edit or delete the role permission policy that defines the permissions for the assumed role. The temporary security credentials obtained by assuming a role can never have more permissions than those defined in the permissions policy of the assumed role, and the permissions assigned to temporary security credentials are evaluated each time they are used to make an AWS request. When you edit or delete the permission policy of a role, the changes affect the permissions of *all* temporary security credentials associated with that role, including credentials that were issued before you changed the role's permissions policy. You can immediately revoke all permissions to a session by following the steps at [Revoking IAM Role Temporary Security Credentials \(p. 277\)](#).

For more information about editing a role permission policy, see [Modifying a Role \(p. 279\)](#).

Denying Access to Credentials Created by GetFederationToken or GetSessionToken

To change or remove the permissions assigned to the temporary security credentials obtained by calling the `GetFederationToken` or `GetSessionToken` API operations, you edit or delete the policies that are attached to the IAM user whose credentials were used to call `GetFederationToken` or `GetSessionToken`. The temporary security credentials that were obtained by calling `GetFederationToken` or `GetSessionToken` can never have more permissions than the IAM user whose credentials were used to obtain them. In addition the permissions assigned to temporary security credentials are evaluated each time they are used to make an AWS request. It is important to note that when you edit or delete the permissions of an IAM user, the changes affect the IAM user as well as all temporary security credentials created by that user.

Important

You cannot change the permissions for an AWS account root user. Likewise, you cannot change the permissions for the temporary security credentials that were created by calling `GetFederationToken` or `GetSessionToken` while signed in as the root user. For this reason,

we recommend that you do not call `GetFederationToken` or `GetSessionToken` as a root user.

For information about how to change or remove the policies associated with the IAM user whose credentials were used to call `GetFederationToken` or `GetSessionToken`, see [Managing IAM Policies \(p. 442\)](#).

Denying Access to Temporary Security Credentials by Name

You can deny access to temporary security credentials without affecting the permissions of the IAM user or role that created the credentials. You do this by specifying the Amazon Resource Name (ARN) of the temporary security credentials in the `Principal` element of a resource-based policy. (Only some AWS services support resource-based policies.)

Denying Access to Federated Users

For example, imagine you have an IAM user named `token-app` whose credentials are used to call `GetFederationToken`. The `GetFederationToken` API call resulted in temporary security credentials associated with a federated user named Bob (the federated user's name is taken from the `Name` parameter of the API call). To deny federated user Bob's access to an S3 bucket called `EXAMPLE-BUCKET`, you attach the following example bucket policy to `EXAMPLE-BUCKET`. It is important to note that this affects the federated user's Amazon S3 permissions only—any other permissions granted to the federated user remain intact.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {"Principal": {"AWS": "arn:aws:sts::ACCOUNT-ID-WITHOUT-HYPHENS:federated-user/Bob"},  
         "Effect": "Deny",  
         "Action": "s3:*",  
         "Resource": "arn:aws:s3:::EXAMPLE-BUCKET"}  
    ]  
}
```

You can specify the ARN of the IAM user whose credentials were used to call `GetFederationToken` in the `Principal` element of the bucket policy, instead of specifying the federated user. In that case, the `Principal` element of the preceding policy would look like this:

```
"Principal": {"AWS": "arn:aws:iam::ACCOUNT-ID-WITHOUT-HYPHENS:user/token-app"}
```

It is important to note that specifying the ARN of IAM user `token-app` in the policy will result in denying access to *all* federated users created by `token-app`, not only the federated user named Bob.

Denying Access to Assumed Role Users

It is also possible to specify the ARN of the temporary security credentials that were created by assuming a role. The difference is the syntax used in the `Principal` element of the resource-based policy. For example, a user assumes a role called `Accounting-Role` and specifies a `RoleSessionName` of `Mary` (`RoleSessionName` is a parameter of the `AssumeRole` API call). To deny access to the temporary security credentials that resulted from this API call, the `Principal` element of the resource-based policy would look like this:

```
"Principal": {"AWS": "arn:aws:sts::ACCOUNT-ID-WITHOUT-HYPHENS:assumed-role/Accounting-Role/Mary"}
```

You can also specify the ARN of the IAM role in the `Principal` element of a resourced-based policy, as in the following example. In this case, the policy will result in denying access to *all* temporary security credentials associated with the role named `Accounting-Role`.

```
"Principal": {"AWS": "arn:aws:iam::ACCOUNT-ID-WITHOUT-HYPHENS:role/Accounting-Role"}
```

Denying Access to Temporary Security Credentials Issued Before a Specific Time

It is possible to deny access only to temporary security credentials that were created before a specific time and date. You do this by specifying a value for the `aws:TokenIssueTime` key in the `Condition` element of a policy. The following policy shows an example. You attach a policy similar to the following example to the IAM user that created the temporary security credentials. The policy denies all permissions but only when the value of `aws:TokenIssueTime` is earlier than the specified date and time. The value of `aws:TokenIssueTime` corresponds to the exact time at which the temporary security credentials were created. The `aws:TokenIssueTime` value is only present in the context of AWS requests that are signed with temporary security credentials, so the `Deny` statement in the policy will not affect requests that are signed with the long-term credentials of the IAM user.

The following policy can also be attached to a role. In that case, the policy affects only the temporary security credentials that were created by the role before the specified date and time. If the credentials were created by the role after the specified date and time, the `Condition` element in the policy is evaluated to false, so the `Deny` statement has no effect.

Example Policy that Denies All Permissions to Temporary Credentials by Issue Time

```
{  
    "Version": "2012-10-17",  
    "Statement": {  
        "Effect": "Deny",  
        "Action": "*",  
        "Resource": "*",  
        "Condition": {"DateLessThan": {"aws:TokenIssueTime": "2014-05-07T23:47:00Z"}}  
    }  
}
```

Valid users whose sessions are revoked in this way must acquire temporary credentials for a new session to continue working. Note that the AWS CLI caches credentials until they expire. To force the CLI to delete and refresh cached credentials that are no longer valid, run one of the following commands:

Linux, MacOS, or Unix

```
$ rm -r ~/.aws/cli/cache
```

Windows

```
C:\> del /s /q %UserProfile%\.aws\cli\cache
```

Granting Permissions to Create Temporary Security Credentials

By default, IAM users do not have permission to create temporary security credentials for federated users and roles. You must use a policy to provide your users with these permissions. Although you can grant permissions directly to a user, we strongly recommend that you grant permissions to a group. This makes management of the permissions much easier. When someone no longer needs to perform the tasks associated with the permissions, you simply remove them from the group. If someone else needs to perform that task, add them to the group to grant the permissions.

To grant an IAM group permission to create temporary security credentials for federated users or roles, you attach a policy that grants one or both of the following privileges:

- For federated users to access an IAM role, grant access to AWS STS `AssumeRole`.
- For federated users that don't need a role, grant access to AWS STS `GetFederationToken`.

For more information about the differences between the `AssumeRole` and `GetFederationToken` API operations, see [Requesting Temporary Security Credentials \(p. 309\)](#).

IAM users can also call `GetSessionToken` to create temporary security credentials. No permissions are required for a user to call `GetSessionToken`. The purpose of this operation is to authenticate the user using MFA. You cannot use policies to control authentication. This means that you cannot prevent IAM users from calling `GetSessionToken` to create temporary credentials.

Example : A policy that grants permission to assume a role

The following example policy grants permission to call `AssumeRole` for the `UpdateApp` role in AWS account 123123123123. When `AssumeRole` is used, the user (or application) that creates the security credentials on behalf of a federated user cannot delegate any permissions that are not already specified in the role permission policy.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": "sts:AssumeRole",  
            "Resource": "arn:aws:iam::123123123123:role/UpdateAPP"  
        }  
    ]  
}
```

Example : A policy that grants permission to create temporary security credentials for a federated user

The following example policy grants permission to access `GetFederationToken`.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": "sts:GetFederationToken",  
            "Resource": "*"  
        }  
    ]  
}
```

Important

When you give IAM users permission to create temporary security credentials for federated users with `GetFederationToken`, be aware that this permits those users to delegate their own permissions. For more information about delegating permissions across IAM users and AWS accounts, see [Examples of Policies for Delegating Access \(p. 250\)](#). For more information about controlling permissions in temporary security credentials, see [Controlling Permissions for Temporary Security Credentials \(p. 322\)](#).

Example : A policy that grants a user limited permission to create temporary security credentials for federated users

When you let an IAM user call `GetFederationToken`, it is a best practice to restrict the permissions that the IAM user can delegate. For example, the following policy shows how to let an IAM user create temporary security credentials only for federated users whose names start with *Manager*.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": "sts:GetFederationToken",  
            "Resource": ["arn:aws:sts::123456789012:federated-user/Manager*"]  
        }  
    ]  
}
```

}

Managing AWS STS in an AWS Region

By default, the AWS Security Token Service (AWS STS) is available as a global service, and all AWS STS requests go to a single endpoint at <https://sts.amazonaws.com>. AWS recommends using Regional AWS STS endpoints instead of the global endpoint to reduce latency, build in redundancy, and increase session token validity.

- **Reduce latency** – By making your AWS STS calls to an endpoint that is geographically closer to your services and applications, you can access AWS STS services with lower latency and better response times.
- **Build in redundancy** – You can add code to your application that switches your AWS STS API calls to a different Region. This ensures that if the first Region stops responding, your application continues to operate. This redundancy is not automatic; you must build the functionality into your code.
- **Increase session token validity** – Session tokens from Regional AWS STS endpoints are valid in all AWS Regions. Session tokens from the global STS endpoint are valid only in AWS Regions that are enabled by default. If you intend to enable a new Region for your account, you can use session tokens from Regional STS endpoints. If you choose to use the global endpoint, you must change the Region compatibility of STS session tokens for the global endpoint. Doing so ensures that tokens are valid in all AWS Regions.

Managing Global Endpoint Session Tokens

Most AWS Regions are enabled for operations in all AWS services by default. Those Regions are automatically activated for use with AWS STS. Some Regions, such as Asia Pacific (Hong Kong), must be manually enabled. To learn more about enabling and disabling AWS Regions, see [Managing AWS Regions](#) in the [AWS General Reference](#). When you enable these AWS Regions, they are automatically activated for use with AWS STS. You cannot activate the STS endpoint for a Region that is disabled. Tokens that are valid in all AWS Regions include more characters than tokens that are valid in Regions that are enabled by default. Changing this setting might affect existing systems where you temporarily store tokens.

You can change this setting using the AWS Management Console, AWS CLI, or AWS API.

To change the Region compatibility of session tokens for the global endpoint (console)

1. Sign in as a root user or an IAM user with permissions to perform IAM administration tasks. To change the compatibility of session tokens, you must have a policy that allows the `iam:SetSecurityTokenServicePreferences` action.
2. Open the [IAM console](#). In the navigation pane, choose **Account settings**.
3. If necessary, expand the **Security Token Service (STS)** section. In the first table next to **Global endpoint**, the **Region compatibility of session tokens** column indicates **Valid only in AWS Regions enabled by default**. Choose **Change**.
4. In the **Change region compatibility of session tokens for global endpoint** dialog box, select **Valid in all AWS Regions**. Then choose **Save changes**.

Note

Tokens that are valid in all AWS Regions include more characters than tokens that are valid in Regions that are enabled by default. Changing this setting might affect existing systems where you temporarily store tokens.

To change the Region compatibility of session tokens for the global endpoint (AWS CLI)

Set the security token version. Version 1 tokens are valid only in AWS Regions that are available by default. These tokens do not work in manually enabled Regions, such as Asia Pacific (Hong Kong).

Version 2 tokens are valid in all Regions. However, version 2 tokens include more characters and might affect systems where you temporarily store tokens.

- [aws iam set-security-token-service-preferences](#)

To change the Region compatibility of session tokens for the global endpoint (AWS API)

Set the security token version. Version 1 tokens are valid only in AWS Regions that are available by default. These tokens do not work in manually enabled Regions, such as Asia Pacific (Hong Kong). Version 2 tokens are valid in all Regions. However, version 2 tokens include more characters and might affect systems where you temporarily store tokens.

- [SetSecurityTokenServicePreferences](#)

Activating and Deactivating AWS STS in an AWS Region

When you activate STS endpoints for a Region, AWS STS can issue temporary credentials to users and roles in your account that make an AWS STS request. Those credentials can then be used in any Region that is enabled by default or is manually enabled. You must activate the Region in the account where the temporary credentials are generated. It does not matter whether a user is signed into the same account or a different account when they make the request.

For example, imagine a user in account A wants to send an `sts:AssumeRole` API request to the STS Regional endpoint `https://sts.us-west-2.amazonaws.com`. The request is for temporary credentials for the role named `Developer` in account B. Because the request is to create credentials for an entity in account B, account B must activate the `us-west-2` Region. Users from account A (or any other account) can call the `us-west-2` endpoint to request credentials for account B whether or not the Region is activated in their accounts.

Note

Active Regions are available to everyone that uses temporary credentials in that account. To control which IAM users or roles can access the Region, use the [aws:RequestedRegion](#) (p. 671) condition key in your permissions policies.

To activate or deactivate AWS STS in a Region that is enabled by default (console)

1. Sign in as a root user or an IAM user with permissions to perform IAM administration tasks.
2. Open the [IAM console](#) and in the navigation pane choose [Account settings](#).
3. If necessary, expand the **Security Token Service (STS)** list, find the Region that you want to activate, and then choose **Activate** or **Deactivate**. Some Regions are not enabled by default, such as the Asia Pacific Hong Kong Region. In that case, when you manually enable the Region, STS is activated automatically. From that point forward, AWS STS is always active for these Regions and it cannot be deactivated. To learn how to manually enable a Region, see [Managing AWS Regions](#) in the [AWS General Reference](#).

Writing Code to Use AWS STS Regions

After you activate a Region, you can direct AWS STS API calls to that Region. The following Java code snippet demonstrates how to configure an `AWSSecurityTokenServiceClient` object to make requests to the Europe (Ireland) (`eu-west-1`) Region with the `setEndpoint` method.

```
EndpointConfiguration regionEndpointConfig = new EndpointConfiguration("https://sts.eu-west-1.amazonaws.com", "eu-west-1");
AWSSecurityTokenService stsRegionalClient = AWSSecurityTokenServiceClientBuilder.standard()
    .withCredentials(credentials)
    .withEndpointConfiguration(regionEndpointConfig)
```

```
.build();
```

AWS STS recommends that you use both the `setRegion` and `setEndpoint` methods to make calls to a Regional endpoint. You can use the `setRegion` method alone for manually enabled Regions, such as Asia Pacific (Hong Kong). In this case, the calls are directed to the STS Regional endpoint. To learn how to manually enable a Region, see [Managing AWS Regions](#) in the *AWS General Reference*. If you use the `setRegion` method alone for Regions enabled by default, the calls are directed to the global endpoint of <https://sts.amazonaws.com>.

In the example, the first line instantiates an `AWSecurityTokenServiceClient` object called `stsClient`. The second line configures the `stsClient` object by calling its `setEndpoint` method and passing the URL of the endpoint as the only parameter. All API calls that use this `stsClient` object are now directed to the specified endpoint.

For all other language and programming environment combinations, refer to the [documentation for the relevant SDK](#).

Region Endpoints

The following table lists the Regions and their endpoints. It indicates which ones are activated by default and which ones you can activate or deactivate.

Region Name	Endpoint	Active by Default	Can Be Activated/Deactivated
--Global--	sts.amazonaws.com	Yes	No
US East (Ohio)	sts.us-east-2.amazonaws.com	Yes	Yes
US East (N. Virginia)	sts.us-east-1.amazonaws.com	Yes	No
US West (N. California)	sts.us-west-1.amazonaws.com	Yes	Yes
US West (Oregon)	sts.us-west-2.amazonaws.com	Yes	Yes
Canada (Central)	sts.ca-central-1.amazonaws.com	Yes	Yes
Asia Pacific (Tokyo)	sts.ap-northeast-1.amazonaws.com	Yes	Yes
Asia Pacific (Seoul)	sts.ap-northeast-2.amazonaws.com	Yes	Yes
Asia Pacific (Mumbai)	sts.ap-south-1.amazonaws.com	Yes	Yes
Asia Pacific (Singapore)	sts.ap-southeast-1.amazonaws.com	Yes	Yes
Asia Pacific (Sydney)	sts.ap-southeast-2.amazonaws.com	Yes	Yes
Asia Pacific (Hong Kong)	sts.ap-east-1.amazonaws.com	Not until the Region is enabled	No
Middle East (Bahrain)	sts.me-south-1.amazonaws.com	Not until the Region is enabled	No
Europe (Frankfurt)	sts.eu-central-1.amazonaws.com	Yes	Yes

Region Name	Endpoint	Active by Default	Can Be Activated/Deactivated
Europe (Ireland)	sts.eu-west-1.amazonaws.com	Yes	Yes
Europe (London)	sts.eu-west-2.amazonaws.com	Yes	Yes
Europe (Paris)	sts.eu-west-3.amazonaws.com	Yes	Yes
Europe (Stockholm)	sts.eu-north-1.amazonaws.com	Yes	Yes
South America (São Paulo)	sts.sa-east-1.amazonaws.com	Yes	Yes

Note

Calls to Regional endpoints, such as `us-east-2.amazonaws.com`, are logged in AWS CloudTrail the same as any call to a Regional service. Calls to the global endpoint, `sts.amazonaws.com`, are logged as calls to a global service. For more information, see [Logging IAM and AWS STS API Calls with AWS CloudTrail \(p. 341\)](#).

Using AWS STS Interface VPC Endpoints

If you use Amazon Virtual Private Cloud (Amazon VPC) to host your AWS resources, you can establish a private connection between your VPC and AWS STS. You can use this connection to enable AWS STS to communicate with your resources on your VPC without going through the public internet.

Amazon VPC is an AWS service that you can use to launch AWS resources in a virtual network that you define. With a VPC, you have control over your network settings, such as the IP address range, subnets, route tables, and network gateways. To connect your VPC to AWS STS, you define an *interface VPC endpoint* for AWS STS. The endpoint provides reliable, scalable connectivity to AWS STS without requiring an internet gateway, network address translation (NAT) instance, or VPN connection. For more information, see [What Is Amazon VPC?](#) in the *Amazon VPC User Guide*.

Interface VPC endpoints are powered by AWS PrivateLink, an AWS technology that enables private communication between AWS services using an elastic network interface with private IP addresses. For more information, see [AWS PrivateLink for AWS Services](#).

The following steps are for users of Amazon VPC. For more information, see [Getting Started with Amazon VPC](#) in the *Amazon VPC User Guide*.

Availability

AWS STS currently supports VPC endpoints in the following Regions:

- US East (Ohio)
- US East (N. Virginia)
- US West (N. California)
- US West (Oregon)
- Asia Pacific (Hong Kong)
- Middle East (Bahrain)
- Asia Pacific (Mumbai)
- Asia Pacific (Osaka-Local)
- Asia Pacific (Seoul)
- Asia Pacific (Singapore)

- Asia Pacific (Sydney)
- Asia Pacific (Tokyo)
- Canada (Central)
- Europe (Frankfurt)
- Europe (Ireland)
- Europe (London)
- Europe (Paris)
- Europe (Stockholm)
- South America (São Paulo)

Create a VPC for AWS STS

To start using AWS STS with your VPC, create an interface VPC endpoint for AWS STS. For more information, see [Creating an Interface Endpoint](#) in the *Amazon VPC User Guide*.

After you create the VPC endpoint, you must use the matching regional endpoint to send your AWS STS requests. AWS STS recommends that you use both the `setRegion` and `setEndpoint` methods to make calls to a Regional endpoint. You can use the `setRegion` method alone for manually enabled Regions, such as Asia Pacific (Hong Kong). In this case, the calls are directed to the STS Regional endpoint. To learn how to manually enable a Region, see [Managing AWS Regions](#) in the *AWS General Reference*. If you use the `setRegion` method alone for Regions enabled by default, the calls are directed to the global endpoint of <https://sts.amazonaws.com>.

When you use regional endpoints, AWS STS calls other AWS services using either public endpoints or private interface VPC endpoints, whichever are in use. For example, assume that you have created an interface VPC endpoint for AWS STS and have already requested temporary credentials from AWS STS from resources that are located in your VPC. In that case, these credentials begin flowing through the interface VPC endpoint by default. For more information about making Regional requests using AWS STS, see [Managing AWS STS in an AWS Region \(p. 333\)](#).

Sample Applications That Use Temporary Credentials

You can use AWS Security Token Service (AWS STS) to create and provide trusted users with temporary security credentials that can control access to your AWS resources. For more information about AWS STS, see [Temporary Security Credentials \(p. 307\)](#). To see how you can use AWS STS to manage temporary security credentials, you can download the following sample applications that implement complete example scenarios:

- [Identity Federation Sample Application for an Active Directory Use Case](#). Demonstrates how to use permissions that are tied to a user defined in Active Directory (.NET/C#) to issue temporary security credentials for accessing Amazon S3 files and buckets.
- [AWS Management Console Federation Proxy Sample Use Case](#). Demonstrates how to create a custom federation proxy that enables single sign-on (SSO) so that existing Active Directory users can sign into the AWS Management Console (.NET/C#).
- [Integrate Shibboleth with AWS Identity and Access Management](#). Shows how to use [Shibboleth](#) and [SAML \(p. 190\)](#) to provide users with single sign-on (SSO) access to the AWS Management Console.

Samples for Web Identity Federation

The following sample applications illustrate how to use web identity federation with providers like Login with Amazon, Amazon Cognito, Facebook, or Google. You can trade authentication from these providers for temporary AWS security credentials to access AWS services.

- [Amazon Cognito Tutorials](#) – We recommend that you use Amazon Cognito with the AWS SDKs for mobile development. Amazon Cognito is the simplest way to manage identity for mobile apps, and it provides additional features like synchronization and cross-device identity. For more information about Amazon Cognito, see [Amazon Cognito Identity](#) in the *AWS Mobile SDK for Android Developer Guide* and [Authenticate Users with Amazon Cognito Identity](#) in the *AWS Mobile SDK for iOS Developer Guide*.
- [Web Identity Federation Playground](#). This website provides an interactive demonstration of [web identity federation \(p. 185\)](#) and the `AssumeRoleWithWebIdentity` API.
- [Build and Deploy a Federated Web Identity Application with AWS Elastic Beanstalk and Login with Amazon](#). This blog post describes how to use `AssumeRoleWithWebIdentity` to obtain temporary security credentials through web identity federation and Login with Amazon. It also explains how to use those credentials in a Python web application that runs on Elastic Beanstalk to make calls to AWS.

Additional Resources for Temporary Security Credentials

The following scenarios and applications can guide you in using temporary security credentials:

- [About Web Identity Federation \(p. 185\)](#). This section discusses how to configure IAM roles when you use web identity federation and the `AssumeRoleWithWebIdentity` API.
- [Configuring MFA-Protected API Access \(p. 145\)](#). This topic explains how to use roles to require multi-factor authentication (MFA) to protect sensitive API actions in your account.
- [Token Vending Machine for Identity Registration](#). This sample Java web application uses the `GetFederationToken` API to serve temporary security credentials to remote clients.

For more information on policies and permissions in AWS see the following topics:

- [Access Management \(p. 355\)](#)
- [Policy Evaluation Logic \(p. 636\)](#).
- [Managing Access Permissions to Your Amazon S3 Resources in Amazon Simple Storage Service Developer Guide](#).
- To learn whether principals in accounts outside of your zone of trust (trusted organization, OU, or account) have access to assume your roles, see [What is IAM Access Analyzer?](#).

The AWS Account Root User

When you first create an Amazon Web Services (AWS) account, you begin with a single sign-in identity that has complete access to all AWS services and resources in the account. This identity is called the AWS account *root user* and is accessed by signing in with the email address and password that you used to create the account.

Important

We strongly recommend that you do not use the root user for your everyday tasks, even the administrative ones. Instead, adhere to the [best practice of using the root user only to create your first IAM user \(p. 61\)](#). Then securely lock away the root user credentials and use them to perform only a few account and service management tasks. To view the tasks that require you to sign in as the root user, see [AWS Tasks That Require Root User](#). For a tutorial on how to set up an administrator for daily use, see [Creating Your First IAM Admin User and Group \(p. 20\)](#).

You can create, rotate, disable, or delete access keys (access key IDs and secret access keys) for your AWS account root user. You can also change your root user password. Anyone who has root user credentials for your AWS account has unrestricted access to all the resources in your account, including billing information.

When you create access keys, you create the access key ID and secret access key as a set. During access key creation, AWS gives you one opportunity to view and download the secret access key part of the access key. If you don't download it or if you lose it, you can delete the access key and then create a new one. You can create root user access keys with the [IAM console](#), AWS CLI, or AWS API.

A newly created access key has the status of *active*, which means that you can use the access key for CLI and API calls. You are [limited to two access keys](#) for each IAM user, which is useful when you want to [rotate the access keys](#). You can also assign up to two access keys to the root user. When you disable an access key, you can't use it for API calls, and inactive keys do count toward your limit. You can create or delete an access key any time. However, when you delete an access key, it's gone forever and can't be retrieved.

You can change the email address and password on the [Security Credentials](#) page. You can also choose **Forgot password?** on the AWS sign-in page to reset your password.

Topics

- [Enable MFA on the AWS Account Root User \(p. 339\)](#)
- [Creating Access Keys for the Root User \(p. 339\)](#)
- [Deleting Access Keys from the Root User \(p. 340\)](#)
- [Changing the Root User's Password \(p. 341\)](#)

Enable MFA on the AWS Account Root User

If you continue to use the root user credentials, we recommend that you follow the security best practice to enable multi-factor authentication (MFA) for your account. Because your root user can perform sensitive operations in your account, adding an additional layer of authentication helps you to better secure your account. Multiple types of MFA are available. For more information about enabling MFA, see the following:

- [Enable a Virtual MFA Device for Your AWS Account Root User \(Console\) \(p. 123\)](#)
- [Enable a Hardware MFA Device for the AWS Account Root User \(Console\) \(p. 133\)](#)

Creating Access Keys for the Root User

You can use the AWS Management Console or AWS programming tools to create access keys for the root user.

To create an access key for the AWS account root user (console)

1. Use your AWS account email address and password to sign in to the [AWS Management Console](#) as the AWS account root user.

Note
If you previously signed in to the console with [IAM user](#) credentials, your browser might remember this preference and open your account-specific sign-in page. You cannot use the IAM user sign-in page to sign in with your AWS account root user credentials. If you see the IAM user sign-in page, choose **Sign-in using root user credentials** near the bottom of the page to return to the main sign-in page. From there, you can enter your AWS account email address and password.
2. Choose your account name in the navigation bar, and then choose **My Security Credentials**.
3. If you see a warning about accessing the security credentials for your AWS account, choose **Continue to Security Credentials**.
4. Expand the **Access keys (access key ID and secret access key)** section.

5. Choose **Create New Access Key**. If this feature is disabled, then you must delete one of the existing access keys before you can create a new key. For more information, see [IAM Entity Object Limits](#) in the *IAM User Guide*.

A warning explains that you have only this one opportunity to view or download the secret access key. It cannot be retrieved later.

- If you choose **Show Access Key**, you can copy the access key ID and secret key from your browser window and paste it somewhere else.
- If you choose **Download Key File**, you receive a file named `rootkey.csv` that contains the access key ID and the secret key. Save the file somewhere safe.

6. When you no longer use the access key [we recommend that you delete it \(p. 66\)](#), or at least mark it inactive by choosing **Make Inactive** so that it cannot be misused.

To create an access key for the root user (AWS CLI or AWS API)

Use one of the following:

- AWS CLI: [aws iam create-access-key](#)
- AWS API: [CreateAccessKey](#)

Deleting Access Keys from the Root User

You can use the AWS Management Console or various programming tools to delete access keys for the root user.

To delete an access key from the AWS account root user (console)

1. Use your AWS account email address and password to sign in to the [AWS Management Console](#) as the root user.

Note

If you previously signed in to the console with [IAM user \(p. 84\)](#) credentials, your browser might remember this preference and open your account-specific sign-in page. You cannot use the IAM user sign-in page to sign in with your AWS account root user credentials. If you see the IAM user sign-in page, choose **Sign-in using root account credentials** near the bottom of the page to return to the main sign-in page. From there, you can type your AWS account email address and password.

2. Choose your account name in the navigation bar, and then choose **My Security Credentials**.
3. If you see a warning about accessing the security credentials for your AWS account, choose **Continue to Security Credentials**.
4. Expand the **Access keys (access key ID and secret access key)** section.
5. Find the access key that you want to delete, and then, under the **Actions** column, choose **Delete**.

Note

You can mark an access key as inactive instead of deleting it. This enables you to resume use of it in the future without having to change either the key ID or secret key. While it is inactive, any attempts to use it in requests to the AWS API fail with the status of access denied.

To delete an access key for the root user (AWS CLI or AWS API)

Use one of the following:

- AWS CLI: [aws iam delete-access-key](#)

- AWS API: [DeleteAccessKey](#)

Changing the Root User's Password

For information about changing the root user's password, see [Changing the AWS Account Root User Password \(p. 100\)](#). To change the root user, you must log in using the root user credentials. To view the tasks that require you to sign in as the root user, see [AWS Tasks that Require Root User](#)

Logging IAM and AWS STS API Calls with AWS CloudTrail

IAM and AWS STS are integrated with AWS CloudTrail, a service that provides a record of actions taken by an IAM user or role. CloudTrail captures all API calls for IAM and AWS STS as events, including calls from the console and from API calls. If you create a trail, you can enable continuous delivery of CloudTrail events to an Amazon S3 bucket. If you don't configure a trail, you can still view the most recent events in the CloudTrail console in **Event history**. You can use CloudTrail to get information about the request that was made to IAM or AWS STS. For example, you can view 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](#).

Topics

- [IAM and AWS STS Information in CloudTrail \(p. 341\)](#)
- [Logging IAM and AWS STS API Requests \(p. 342\)](#)
- [Logging API Requests to Other AWS Services \(p. 342\)](#)
- [Logging Regional Sign-in Events \(p. 342\)](#)
- [Logging User Sign-in Events \(p. 344\)](#)
- [Logging Sign-in Events for Temporary Credentials \(p. 345\)](#)
- [Example IAM API Events in CloudTrail Log \(p. 345\)](#)
- [Example AWS STS API Events in CloudTrail Log \(p. 346\)](#)
- [Example Sign-In Events in CloudTrail Log \(p. 352\)](#)

IAM and AWS STS Information in CloudTrail

CloudTrail is enabled on your AWS account when you create the account. When activity occurs in IAM or AWS STS, 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 IAM and AWS STS, 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 IAM and AWS STS actions are logged by CloudTrail and are documented in the [IAM API Reference](#) and the [AWS Security Token Service API Reference](#).

Logging IAM and AWS STS API Requests

CloudTrail logs all authenticated API requests (made with credentials) to IAM and AWS STS API operations. CloudTrail also logs nonauthenticated requests to the AWS STS actions, `AssumeRoleWithSAML` and `AssumeRoleWithWebIdentity`, and logs information provided by the identity provider. You can use this information to map calls made by a federated user with an assumed role back to the originating external federated caller. In the case of `AssumeRole`, you can map calls back to the originating AWS service or to the account of the originating user. The `userIdentity` section of the JSON data in the CloudTrail log entry contains the information that you need to map the `AssumeRole*` request with a specific federated user. For more information, see [CloudTrail userIdentity Element](#) in the [AWS CloudTrail User Guide](#).

For example, calls to the IAM `CreateUser`, `DeleteRole`, `ListGroups`, and other API operations are all logged by CloudTrail.

Examples for this type of log entry are presented later in this topic.

Important

If you activate AWS STS endpoints in Regions other than the default global endpoint, then you must also turn on CloudTrail logging in those Regions. This is necessary to record any AWS STS API calls that are made in those Regions. For more information, see [Turning On CloudTrail in Additional Regions](#) in the AWS CloudTrail User Guide.

Logging API Requests to Other AWS Services

Authenticated requests to other AWS service API operations are logged by CloudTrail, and these log entries contain information about who generated the request.

For example, assume that you made a request to list Amazon EC2 instances or create an CodeDeploy deployment group. Details about the person or service that made the request are contained in the log entry for that request. This information helps you determine whether the request was made by the AWS account root user, an IAM user, a role, or another AWS service.

For more details about the user identity information in CloudTrail log entries, see [userIdentity Element](#) in the [AWS CloudTrail User Guide](#).

Logging Regional Sign-in Events

If you enable CloudTrail to log sign-in events to your logs, you need to be aware of how CloudTrail chooses where to log the events.

- If your users sign in directly to a console, they are redirected to either a global or a Regional sign-in endpoint. The endpoint depends on whether the selected service console supports Regions. For example, the main console home page supports Regions. If you sign in to `https://alias.signin.aws.amazon.com/console`, you are redirected to a Regional sign-in endpoint such as `https://us-east-2.signin.aws.amazon.com`. This redirection creates a Regional CloudTrail log entry in the user's Region's log.

On the other hand, the Amazon S3 console does not support Regions, so if you sign in to `https://alias.signin.aws.amazon.com/console/s3`, AWS redirects you to the global sign-in endpoint at `https://signin.aws.amazon.com`. This redirection creates a global CloudTrail log entry.

- You can manually request a certain Regional sign-in endpoint by signing in to the Region-enabled main console home page using a URL like <https://alias.signin.aws.amazon.com/console?region=ap-southeast-1>. In this case, AWS redirects you to the ap-southeast-1 Regional sign-in endpoint and results in a Regional CloudTrail log event.

Whether the sign-in event is considered Regional or global depends on the console the user signs into and how the user constructs the sign-in URL.

- Is the service console Regionalized? If so, then the sign-in request is automatically redirected to a Regional sign-in endpoint and the event is logged in that Region's CloudTrail log. For example, if you sign in to <https://alias.signin.aws.amazon.com/console>, which is Regionalized, you are redirected to a sign-in endpoint in your Region, such as <https://us-east-2.signin.aws.amazon.com>. The event is logged in that Region's log.

However, some services are not Regionalized yet. For example, the Amazon S3 service is *not* currently Regionalized. If you sign in to <https://alias.signin.aws.amazon.com/console/s3>, you are redirected to the global sign-in endpoint at <https://signin.aws.amazon.com>. This redirection creates an event in your global log.

- You can also manually request a specific Regional sign-in endpoint by using a URL such as <https://alias.signin.aws.amazon.com/console?region=ap-southeast-1>. This URL redirects to the ap-southeast-1 Regional sign-in endpoint. This redirection results in an event in the Regional log.

Preventing Duplicate Regional Log Entries

CloudTrail creates separate trails in each Region. These trails include information for events that occur in those Regions, plus global events and events that are not region-specific. Examples include IAM API calls, AWS STS calls to the global endpoint, and AWS sign-in events. For example, assume that you have two trails, each in a different Region. If you then create a new IAM user, the `CreateUser` event is added to the log files in both Regions, creating a duplicate log entry.

AWS Security Token Service (STS) is a global service with a single global endpoint at <https://sts.amazonaws.com>. Calls to this endpoint are logged as calls to a global service. However, because this endpoint is physically located in the US East (N. Virginia) Region, your logs list us-east-1 as the event Region. CloudTrail does not write these logs to the US East (Ohio) Region unless you choose to include global service logs in that Region. AWS STS also allows calls to Regional endpoints, such as `sts.eu-central-1.amazonaws.com`. CloudTrail writes calls to all Regional endpoints to their respective Regions. For example, calls to `sts.us-east-2.amazonaws.com` are published to the US East (Ohio) Region. Calls to `sts.eu-central-1.amazonaws.com` are published in the Europe (Frankfurt) Region logs.

For more information about multiple Regions and AWS STS, see [Managing AWS STS in an AWS Region \(p. 333\)](#).

The following table lists the Regions and how CloudTrail logs AWS STS requests in each Region. The "Location" column indicates which logs CloudTrail writes to. "Global" means that the event is logged in any Region for which you choose to include global service logs in that Region. "Region" means that the event is logged only in the Region where the endpoint is located. The last column indicates how the request's Region is identified in the log entry.

Region name	Region identity in CloudTrail log	Endpoint	Location of CloudTrail logs
n/a - global	us-east-1	<code>sts.amazonaws.com</code>	Global
US East (Ohio)	us-east-2	<code>sts.us-east-2.amazonaws.com</code>	Region

Region name	Region identity in CloudTrail log	Endpoint	Location of CloudTrail logs
US East (N. Virginia)	us-east-1	sts.us-east-1.amazonaws.com	Region
US West (N. California)	us-west-1	sts.us-west-1.amazonaws.com	Region
US West (Oregon)	us-west-2	sts.us-west-2.amazonaws.com	Region
Canada (Central)	ca-central-1	sts.ca-central-1.amazonaws.com	Region
Europe (Frankfurt)	eu-central-1	sts.eu-central-1.amazonaws.com	Region
Europe (Ireland)	eu-west-1	sts.eu-west-1.amazonaws.com	Region
Europe (London)	eu-west-2	sts.eu-west-2.amazonaws.com	Region
Asia Pacific (Tokyo)	ap-northeast-1	sts.ap-northeast-1.amazonaws.com	Region
Asia Pacific (Seoul)	ap-northeast-2	sts.ap-northeast-2.amazonaws.com	Region
Asia Pacific (Mumbai)	ap-south-1	sts.ap-south-1.amazonaws.com	Region
Asia Pacific (Singapore)	ap-southeast-1	sts.ap-southeast-1.amazonaws.com	Region
Asia Pacific (Sydney)	ap-southeast-2	sts.ap-southeast-2.amazonaws.com	Region
South America (São Paulo)	sa-east-1	sts.sa-east-1.amazonaws.com	Region

When you configure CloudTrail to aggregate trail information from multiple Regions in your account into a single Amazon S3 bucket, IAM events are duplicated in the logs. In other words, the trail for each Region writes the same IAM event to the aggregated log. To prevent this duplication, you can include global events selectively. A typical approach is to enable global events in one trail. Then disable global events in all other trails that write to the same Amazon S3 bucket. That way only one set of global events is written.

For more information, see [Aggregating Logs](#) in the *AWS CloudTrail User Guide*.

Logging User Sign-in Events

CloudTrail logs sign-in events to the AWS Management Console, the AWS discussion forums, and AWS Marketplace. CloudTrail logs successful and failed sign-in attempts for IAM users and federated users.

For AWS account root users, only successful sign-in events are logged. Unsuccessful sign-in events by the root user are *not* logged by CloudTrail.

As a security best practice, AWS does not log the entered IAM user name text when the sign-in failure is caused by *an incorrect user name*. The user name text is masked by the value `HIDDEN_DUE_TO_SECURITY_REASONS`. For an example of this, see [Example Sign-in Failure Event Caused by Incorrect User Name \(p. 353\)](#), later in this topic. The user name text is obscured because such failures might be caused by user errors. Logging these errors could expose potentially sensitive information. For example:

- You accidentally type your password in the user name box.
- You choose the link for one AWS account's sign-in page, but then type the account number for a different one.
- You forget which account you are signing in to and accidentally type the account name of your personal email account, your bank sign-in identifier, or some other private ID.

Logging Sign-in Events for Temporary Credentials

When a principal requests temporary credentials, the principal type determines how CloudTrail logs the event. This can be complicated when a principal assumes a role in another account. There are multiple API calls to perform operations related to role cross-account operations. First, the principal calls an AWS STS API to retrieve the temporary credentials. That operation is logged in the calling account and the account where the AWS STS operation is performed. Then the principal then uses the role to perform other API calls in the assumed role's account,

The following table shows how CloudTrail logs different information for each of the API calls that generate temporary credentials.

Principal Type	STS API	User Identity in CloudTrail Log for Caller's Account	User Identity in CloudTrail Log for the Assumed Role's Account	User Identity in CloudTrail Log for the Role's Subsequent API Calls
AWS account root user credentials	GetSessionToken	Root user identity	Role owner account is same as calling account	Root user identity
IAM user	GetSessionToken	IAM user identity	Role owner account is same as calling account	IAM user identity
IAM user	GetFederationToken	IAM user identity	Role owner account is same as calling account	IAM user identity
IAM user	AssumeRole	IAM user identity	Account number and principal ID (if a user), or AWS service principal	Role identity only (no user)
Externally authenticated user	AssumeRoleWithSAMl/a		SAML user identity	Role identity only (no user)
Externally authenticated user	AssumeRoleWithWebIdentity		OIDC/Web user identity	Role identity only (no user)

Example IAM API Events in CloudTrail Log

CloudTrail log files contain events that are formatted using JSON. An API event represents a single API request and includes information about the principal, the requested action, any parameters, and the date and time of the action.

Example IAM API Event in CloudTrail Log File

The following example shows a CloudTrail log entry for a request made for the IAM `GetUserPolicy` action.

```
{  
    "eventVersion": "1.05",  
    "userIdentity": {  
        "type": "IAMUser",  
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",  
        "arn": "arn:aws:iam::444455556666:user/JaneDoe",  
        "accountId": "444455556666",  
        "accessKeyId": "AKIAI44QH8DHBEXAMPLE",  
        "userName": "JaneDoe",  
        "sessionContext": {  
            "attributes": {  
                "mfaAuthenticated": "false",  
                "creationDate": "2014-07-15T21:39:40Z"  
            }  
        },  
        "invokedBy": "signin.amazonaws.com"  
    },  
    "eventTime": "2014-07-15T21:40:14Z",  
    "eventSource": "iam.amazonaws.com",  
    "eventName": " GetUserPolicy",  
    "awsRegion": "us-east-2",  
    "sourceIPAddress": "signin.amazonaws.com",  
    "userAgent": "signin.amazonaws.com",  
    "requestParameters": {  
        "userName": "JaneDoe",  
        "policyName": "ReadOnlyAccess-JaneDoe-201407151307"  
    },  
    "responseElements": null,  
    "requestID": "9EXAMPLE-0c68-11e4-a24e-d5e16EXAMPLE",  
    "eventID": "cEXAMPLE-127e-4632-980d-505a4EXAMPLE"  
}
```

From this event information, you can determine that the request was made to get a user policy named `ReadOnlyAccess-JaneDoe-201407151307` for user `JaneDoe`, as specified in the `requestParameters` element. You can also see that the request was made by an IAM user named `JaneDoe` on July 15, 2014 at 9:40 PM (UTC). In this case, the request originated in the AWS Management Console, as you can tell from the `userAgent` element.

Example AWS STS API Events in CloudTrail Log

CloudTrail log files contain events that are formatted using JSON. An API event represents a single API request and includes information about the principal, the requested action, any parameters, and the date and time of the action.

Example Cross-Account AWS STS API Events in CloudTrail Log Files

The IAM user named `JohnDoe` in account `777788889999` calls the AWS STS `AssumeRole` action to assume the role `EC2-dev` in account `111122223333`.

```
{  
    "eventVersion": "1.05",  
    "userIdentity": {  
        "type": "IAMUser",  
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",  
        "arn": "arn:aws:iam::777788889999:user/JohnDoe",  
        "accountId": "777788889999",  
        "accessKeyId": "AKIAI44QH8DHBEXAMPLE",  
        "userName": "JohnDoe",  
        "sessionContext": {  
            "attributes": {  
                "mfaAuthenticated": "false",  
                "creationDate": "2014-07-15T21:39:40Z"  
            }  
        },  
        "invokedBy": "signin.amazonaws.com"  
    },  
    "eventTime": "2014-07-15T21:40:14Z",  
    "eventSource": "sts.amazonaws.com",  
    "eventName": " AssumeRole",  
    "awsRegion": "us-east-2",  
    "sourceIPAddress": "signin.amazonaws.com",  
    "userAgent": "signin.amazonaws.com",  
    "requestParameters": {  
        "roleArn": "arn:aws:iam::111122223333:role/EC2-dev",  
        "durationSeconds": 3600  
    },  
    "responseElements": {  
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",  
        "roles": [ {  
            "roleName": "EC2-dev",  
            "principalArn": "arn:aws:iam::111122223333:role/EC2-dev",  
            "sessionToken": "WgVtP...XzJL",  
            "durationSeconds": 3600, "externalId": null  
        } ]  
    },  
    "requestID": "9EXAMPLE-0c68-11e4-a24e-d5e16EXAMPLE",  
    "eventID": "cEXAMPLE-127e-4632-980d-505a4EXAMPLE"  
}
```

```

"principalId": "AIDAQRSTUVWXYZEXAMPLE",
"arn": "arn:aws:iam::777788889999:user/JohnDoe",
"accountId": "777788889999",
"accessKeyId": "AKIAQRSTUVWXYZEXAMPLE",
"userName": "JohnDoe"
},
"eventTime": "2014-07-18T15:07:39Z",
"eventSource": "sts.amazonaws.com",
"eventName": "AssumeRole",
"awsRegion": "us-east-2",
"sourceIPAddress": "192.0.2.101",
"userAgent": "aws-cli/1.11.10 Python/2.7.8 Linux/3.2.45-0.6.wd.865.49.315.metal1.x86_64
botocore/1.4.67",
"requestParameters": {
    "roleArn": "arn:aws:iam::111122223333:role/EC2-dev",
    "roleSessionName": "JohnDoe-EC2-dev"
    "serialNumber": "arn:aws:iam::777788889999:mfa"
},
"responseElements": {
    "credentials": {
        "sessionToken": "<encoded session token blob>",
        "accessKeyId": "AKIAQRSTUVWXYZEXAMPLE",
        "expiration": "Jul 18, 2014 4:07:39 PM"
    },
    "assumedRoleUser": {
        "assumedRoleId": "AIDAQRSTUVWXYZEXAMPLE:JohnDoe-EC2-dev",
        "arn": "arn:aws:sts::111122223333:assumed-role/EC2-dev/JohnDoe-EC2-dev"
    }
},
"resources": [
    {
        "ARN": "arn:aws:iam::111122223333:role/EC2-dev",
        "accountId": "111122223333",
        "type": "AWS::IAM::Role"
    }
],
"requestID": "4EXAMPLE-0e8d-11e4-96e4-e55c0EXAMPLE",
"sharedEventID": "bEXAMPLE-efea-4a70-b951-19a88EXAMPLE",
"eventID": "dEXAMPLE-ac7f-466c-a608-4ac8dEXAMPLE",
"eventType": "AwsApiCall",
"recipientAccountId": "111122223333"
}
}

```

The second example shows the assumed role account's (111122223333) CloudTrail log entry for the same request.

```

{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "AWSAccount",
        "principalId": "AIDAQRSTUVWXYZEXAMPLE",
        "accountId": "777788889999"
    },
    "eventTime": "2014-07-18T15:07:39Z",
    "eventSource": "sts.amazonaws.com",
    "eventName": "AssumeRole",
    "awsRegion": "us-east-2",
    "sourceIPAddress": "192.0.2.101",
    "userAgent": "aws-cli/1.11.10 Python/2.7.8 Linux/3.2.45-0.6.wd.865.49.315.metal1.x86_64
botocore/1.4.67",
    "requestParameters": {
        "roleArn": "arn:aws:iam:: 111122223333:role/EC2-dev",
        "roleSessionName": "JohnDoe-EC2-dev",
        "serialNumber": "arn:aws:iam::777788889999:mfa"
    }
}

```

```

},
"responseElements": {
    "credentials": {
        "sessionToken": "<encoded session token blob>",
        "accessKeyId": "AKIAQRSTUVWXYZEXAMPLE",
        "expiration": "Jul 18, 2014 4:07:39 PM"
    },
    "assumedRoleUser": {
        "assumedRoleId": "AIDAQRSTUVWXYZEXAMPLE:JohnDoe-EC2-dev",
        "arn": "arn:aws:sts::111122223333:assumed-role/EC2-dev/JohnDoe-EC2-dev"
    }
},
"requestID": "4EXAMPLE-0e8d-11e4-96e4-e55c0EXAMPLE",
"sharedEventID": "bEXAMPLE-efea-4a70-b951-19a88EXAMPLE",
"eventID": "dEXAMPLE-ac7f-466c-a608-4ac8dEXAMPLE"
}
}

```

Example AWS STS Role Chaining API Event in CloudTrail Log File

The following example shows a CloudTrail log entry for a request made by John Doe in account 111111111111. John previously used his JohnDoe user to assume the JohnRole1 role. For this request, he uses the credentials from that role to assume the JonRole2 role. This is known as [role chaining \(p. 177\)](#). John passes two [session tags \(p. 299\)](#) into the request. He sets those two tags as transitive. The request inherits the Department tag as transitive because John set it as transitive when he assumed JohnRole1. For more information about transitive keys in role chains, see [Chaining Roles with Session Tags \(p. 305\)](#).

```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROAIN5ATK5U7KEXAMPLE:JohnRole1",
        "arn": "arn:aws:sts::111111111111:assumed-role/JohnDoe/JohnRole1",
        "accountId": "111111111111",
        "accessKeyId": "AKIAI44QH8DHBEEXAMPLE",
        "sessionContext": {
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2019-10-02T21:50:54Z"
            },
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAIN5ATK5U7KEXAMPLE",
                "arn": "arn:aws:iam::111111111111:role/JohnRole1",
                "accountId": "111111111111",
                "userName": "JohnDoe"
            }
        }
    },
    "eventTime": "2019-10-02T22:12:29Z",
    "eventSource": "sts.amazonaws.com",
    "eventName": "AssumeRole",
    "awsRegion": "us-east-2",
    "sourceIPAddress": "123.145.67.89",
    "userAgent": "aws-cli/1.16.248 Python/3.4.7
Linux/4.9.184-0.1.ac.235.83.329.metal1.x86_64 botocore/1.12.239",
    "requestParameters": {
        "incomingTransitiveTags": {
            "Department": "Engineering"
        },
        "tags": [
            {
                "value": "johndoe@example.com",

```

```

        "key": "Email"
    },
    {
        "value": "12345",
        "key": "CostCenter"
    }
],
"roleArn": "arn:aws:iam::111111111111:role/JohnRole2",
"roleSessionName": "Role2WithTags",
"transitiveTagKeys": [
    "Email",
    "CostCenter"
],
"durationSeconds": 3600
},
"responseElements": {
    "credentials": {
        "accessKeyId": "ASIAWHOJDLGPOEXAMPLE",
        "expiration": "Oct 2, 2019 11:12:29 PM",
        "sessionToken": "AgoJb3JpZ2luX2VjEB4aCXVzLXd1c3QtMSJHMEXAMPLETOKEN
+//rJb8Lo30mFc5M1hFCEbubZvEj0wHB/mDMwIgSEE9gk/Zjr09tZV7F1HDTMhmEXAMPLETOKEN/iEJ/
rkqngII9///////////
ARABGgw0MjgzMDc4NjM5NjYiDLZjZFKwP4qxQG5sFCryASO4UPz5qE97wPPH1eLMvs7CgSDBSWfonmRTcfokm2FN1+hWUdQQH6adjbb
+C+WKFZb701eiv9J5La2EXAMPLETOKEN/c7S51ro1WUJ0q3Cxuo+8HUoSxVhQHM7zF7mWWLhXLEQ52ivL
+F6q5dpXu4aTFedpMfnJa8JtkWwG9x1Axj0Ypy2ok8v5unpQGWych1vwdvj6ez1Dm8Xg1+qIzXILiEXAMPLETOKEN/
vQGqu8H+nxp3kabcrOvTFTvxX6vsc8OGwUfHzAfYGEEXAMPLETOKEN/
L6v1yMM3B1OwFOrQBno1HEjf1oNI8RnQiMNFdUOtWYj7HUZIOCZmjfn8PPHq77N7GJ191zvIZKQA0Owcjg
+mc78zHCj8y0siY8C96paEXAMPLETOKEN/
E3cpksxWdgs91HRzJWSCjN2+r2LTGjYhyPqcmFzzo2mCE7mBNEXAMPLETOKEN/oJy
+2o83YNW5tOidmczgDzJZ4UKR84yGYOMfSnF4XcEJrDgAJ3OJFwmTctQICALSwLEXAMPLETOKEN"
    },
    "assumedRoleUser": {
        "assumedRoleId": "AROAIFR7WHDTSOYQYHFUE:Role2WithTags",
        "arn": "arn:aws:sts::111111111111:assumed-role/test-role/Role2WithTags"
    }
},
"requestID": "b96b0e4e-e561-11e9-8b3f-7b396EXAMPLE",
"eventID": "1917948f-3042-46ec-98e2-62865EXAMPLE",
"resources": [
    {
        "ARN": "arn:aws:iam::111122223333:role/JohnRole2",
        "accountId": "111111111111",
        "type": "AWS::IAM::Role"
    }
],
"eventType": "AwsApiCall",
"recipientAccountId": "111111111111"
}

```

Example AWS Service AWS STS API Event in CloudTrail Log File

The following example shows a CloudTrail log entry for a request made by an AWS service calling another service API using permissions from a service role. It shows the CloudTrail log entry for the request made in account 777788889999.

```
{
"eventVersion": "1.04",
"userIdentity": {
    "type": "AssumedRole",
    "principalId": "AIDAQRSTUVWXYZEXAMPLE:devdsk",
    "arn": "arn:aws:sts::777788889999:assumed-role/AssumeNothing/devdsk",
    "accountId": "777788889999",
    "accessKeyId": "AKIAQRSTUVWXYZEXAMPLE",
    "sessionContext": {

```

```

    "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2016-11-14T17:25:26Z"
    },
    "sessionIssuer": {
        "type": "Role",
        "principalId": "AIDAQRSTUVWXYZEXAMPLE",
        "arn": "arn:aws:iam::777788889999:role/AssumeNothing",
        "accountId": "777788889999",
        "userName": "AssumeNothing"
    }
},
"eventTime": "2016-11-14T17:25:45Z",
"eventSource": "s3.amazonaws.com",
"eventName": "DeleteBucket",
"awsRegion": "us-east-2",
"sourceIPAddress": "192.0.2.1",
"userAgent": "[aws-cli/1.11.10 Python/2.7.8 Linux/3.2.45-0.6.wd.865.49.315.metal1.x86_64 botocore/1.4.67]",
"requestParameters": {
    "bucketName": "my-test-bucket-cross-account"
},
"responseElements": null,
"requestID": "EXAMPLE463D56D4C",
"eventID": "dEXAMPLE-265a-41e0-9352-4401bEXAMPLE",
"eventType": "AwsApiCall",
"recipientAccountId": "777788889999"
}

```

Example SAML AWS STS API Event in CloudTrail Log File

The following example shows a CloudTrail log entry for a request made for the AWS STS AssumeRoleWithSAML action. The request includes the SAML attributes CostCenter and Project that are passed through the SAML assertion as [session tags \(p. 299\)](#). Those tags are set as transitive so that they [persist in role chaining scenarios \(p. 305\)](#).

```

{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "SAMLUser",
        "principalId": "SampleUkh1i4+ExampLexL/jEvs=:SamlExample",
        "userName": "SamlExample",
        "identityProvider": "bdGOnTesti4+ExampLexL/jEvs="
    },
    "eventTime": "2019-11-01T19:14:36Z",
    "eventSource": "sts.amazonaws.com",
    "eventName": "AssumeRoleWithSAML",
    "awsRegion": "us-east-2",
    "sourceIPAddress": "192.0.2.101",
    "userAgent": "aws-cli/1.16.263 Python/3.4.7
Linux/4.9.184-0.1.ac.235.83.329.metal1.x86_64 botocore/1.12.253",
    "requestParameters": {
        "SAMLAssertionID": "_c0046cEXAMPLEb9d4b8eEXAMPLE2619aEXAMPLE",
        "roleSessionName": "MyAssignedRoleSessionName",
        "principalTags": {
            "CostCenter": "987654",
            "Project": "Unicorn",
            "Department": "Engineering"
        },
        "transitiveTagKeys": [
            "CostCenter",
            "Project"
        ]
    }
}

```

```

        ],
        "durationSeconds": 3600,
        "roleArn": "arn:aws:iam::444455556666:role/SAMLTestRoleShibboleth",
        "principalArn": "arn:aws:iam::444455556666:saml-provider/Shibboleth"
    },
    "responseElements": {
        "subjectType": "transient",
        "issuer": "https://server.example.com/idp/shibboleth",
        "credentials": {
            "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
            "expiration": "Mar 23, 2016 2:39:57 AM",
            "sessionToken": "<encoded session token blob>"
        },
        "nameQualifier": "bdGOnTesti4+ExamplexL/jEvs=",
        "assumedRoleUser": {
            "assumedRoleId": "AROAD35QRSTUVWEXAMPLE:MyAssignedRoleSessionName",
            "arn": "arn:aws:sts::444455556666:assumed-role/SAMLTestRoleShibboleth/
MyAssignedRoleSessionName"
        },
        "subject": "SamlExample",
        "audience": "https://signin.aws.amazon.com/saml"
    },
    "resources": [
        {
            "ARN": "arn:aws:iam::444455556666:role/SAMLTestRoleShibboleth",
            "accountId": "444455556666",
            "type": "AWS::IAM::Role"
        },
        {
            "ARN": "arn:aws:iam::444455556666:saml-provider/test-saml-provider",
            "accountId": "444455556666",
            "type": "AWS::IAM::SAMLProvider"
        }
    ],
    "requestID": "6EXAMPLE-e595-11e5-b2c7-c974fEXAMPLE",
    "eventID": "dEXAMPLE-265a-41e0-9352-4401bEXAMPLE",
    "eventType": "AwsApiCall",
    "recipientAccountId": "444455556666"
}

```

Example Web Identity AWS STS API Event in CloudTrail Log File

The following example shows a CloudTrail log entry for a request made for the AWS STS `AssumeRoleWithWebIdentity` action. The request includes the attributes `CostCenter` and `Project` that are passed through the identity provider token as [session tags \(p. 299\)](#). Those tags are set as transitive so that they [persist in role chaining scenarios \(p. 305\)](#).

```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "WebIdentityUser",
        "principalId": "accounts.google.com:<id-of-application>.apps.googleusercontent.com:<id-
of-user>",
        "userName": "<id of user>",
        "identityProvider": "accounts.google.com"
    },
    "eventTime": "2016-03-23T01:39:51Z",
    "eventSource": "sts.amazonaws.com",
    "eventName": "AssumeRoleWithWebIdentity",
    "awsRegion": "us-east-2",
    "sourceIPAddress": "192.0.2.101",
    "userAgent": "aws-cli/1.3.23 Python/2.7.6 Linux/2.6.18-164.el5",
    "requestParameters": {

```

```

    "durationSeconds": 3600,
    "roleArn": "arn:aws:iam::444455556666:role/FederatedWebIdentityRole",
    "roleSessionName": "MyAssignedRoleSessionName"
        "principalTags": {
            "CostCenter": "24680",
            "Project": "Pegasus"
        },
        "transitiveTagKeys": [
            "CostCenter",
            "Project"
        ],
    },
    "responseElements": {
        "provider": "accounts.google.com",
        "subjectFromWebIdentityToken": "<id of user>",
        "audience": "<id of application>.apps.googleusercontent.com",
        "credentials": {
            "accessKeyId": "ASIAQRSTUVWRAOEXAMPLE",
            "expiration": "Mar 23, 2016 2:39:51 AM",
            "sessionToken": "<encoded session token blob>"
        },
        "assumedRoleUser": {
            "assumedRoleId": "AROACQRSTUVWRAOEXAMPLE:MyAssignedRoleSessionName",
            "arn": "arn:aws:sts::444455556666:assumed-role/FederatedWebIdentityRole/MyAssignedRoleSessionName"
        }
    },
    "resources": [
        {
            "ARN": "arn:aws:iam::444455556666:role/FederatedWebIdentityRole",
            "accountId": "444455556666",
            "type": "AWS::IAM::Role"
        }
    ],
    "requestID": "6EXAMPLE-e595-11e5-b2c7-c974fEXAMPLE",
    "eventID": "bEXAMPLE-0b30-4246-b28c-e3da3EXAMPLE",
    "eventType": "AwsApiCall",
    "recipientAccountId": "444455556666"
}

```

Example Sign-In Events in CloudTrail Log

CloudTrail log files contain events that are formatted using JSON. A sign-in event represents a single sign-in request and includes information about the sign-in principal, the Region, and the date and time of the action.

Example Sign-in Success Event in CloudTrail Log File

The following example shows a CloudTrail log entry for a successful sign-in event.

```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::111122223333:user/JohnDoe",
        "accountId": "111122223333",
        "userName": "JohnDoe"
    },
    "eventTime": "2014-07-16T15:49:27Z",
    "eventSource": "signin.amazonaws.com",
    "eventName": "ConsoleLogin",
    "awsRegion": "us-east-2",
}
```

```
"sourceIPAddress": "192.0.2.110",
"userAgent": "Mozilla/5.0 (Windows NT 6.1; WOW64; rv:24.0) Gecko/20100101 Firefox/24.0",
"requestParameters": null,
"responseElements": {
    "ConsoleLogin": "Success"
},
"additionalEventData": {
    "MobileVersion": "No",
    "LoginTo": "https://console.aws.amazon.com/s3/",
    "MFAUsed": "No"
},
"eventID": "3fcfb182-98f8-4744-bd45-10a395ab61cb"
}
```

For more details about the information contained in CloudTrail log files, see [CloudTrail Event Reference](#) in the [AWS CloudTrail User Guide](#).

Example Sign-in Failure Event in CloudTrail Log File

The following example shows a CloudTrail log entry for a failed sign-in event.

```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::111122223333:user/JaneDoe",
        "accountId": "111122223333",
        "userName": "JaneDoe"
    },
    "eventTime": "2014-07-08T17:35:27Z",
    "eventSource": "signin.amazonaws.com",
    "eventName": "ConsoleLogin",
    "awsRegion": "us-east-2",
    "sourceIPAddress": "192.0.2.100",
    "userAgent": "Mozilla/5.0 (Windows NT 6.1; WOW64; rv:24.0) Gecko/20100101 Firefox/24.0",
    "errorMessage": "Failed authentication",
    "requestParameters": null,
    "responseElements": {
        "ConsoleLogin": "Failure"
    },
    "additionalEventData": {
        "MobileVersion": "No",
        "LoginTo": "https://console.aws.amazon.com/sns",
        "MFAUsed": "No"
    },
    "eventID": "11ea990b-4678-4bcd-8fbe-62509088b7cf"
}
```

From this information, you can determine that the sign-in attempt was made by an IAM user named JaneDoe, as shown in the `userIdentity` element. You can also see that the sign-in attempt failed, as shown in the `responseElements` element. You can see that JaneDoe tried to sign in to the Amazon SNS console at 5:35 PM (UTC) on July 8, 2014.

Example Sign-in Failure Event Caused by Incorrect User Name

The following example shows a CloudTrail log entry for an unsuccessful sign-in event caused by the user entering an incorrect user name. AWS masks the `userName` text with `HIDDEN_DUE_TO_SECURITY_REASON` to help prevent exposing potentially sensitive information.

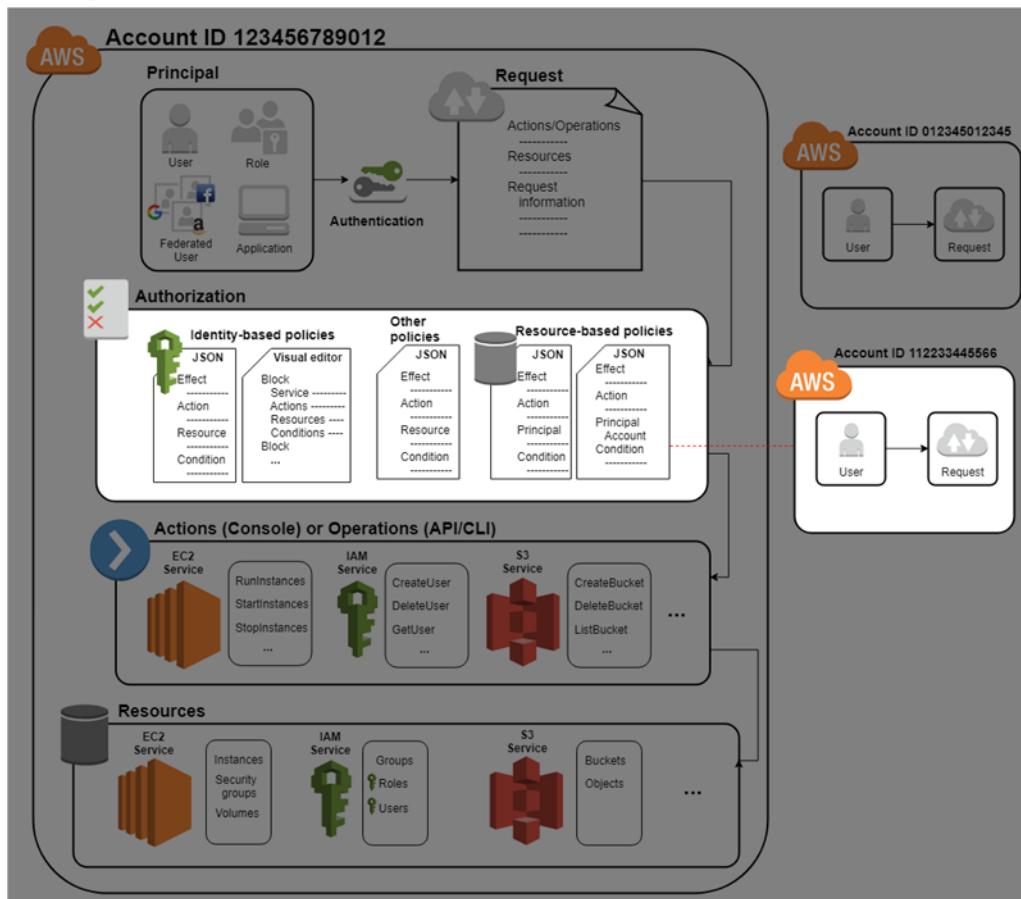
```
{
```

```
"eventVersion": "1.05",
"userIdentity": {
    "type": "IAMUser",
    "accountId": "123456789012",
    "accessKeyId": "",
    "userName": "HIDDEN_DUE_TO_SECURITY_REASONS"
},
"eventTime": "2015-03-31T22:20:42Z",
"eventSource": "signin.amazonaws.com",
"eventName": "ConsoleLogin",
"awsRegion": "us-east-2",
"sourceIPAddress": "192.0.2.101",
"userAgent": "Mozilla/5.0 (Windows NT 6.1; WOW64; rv:24.0) Gecko/20100101 Firefox/24.0",
"errorMessage": "No username found in supplied account",
"requestParameters": null,
"responseElements": {
    "ConsoleLogin": "Failure"
},
"additionalEventData": {
    "LoginTo": "https://console.aws.amazon.com/console/home?state=hashArgs%23&isauthcode=true",
    "MobileVersion": "No",
    "MFAUsed": "No"
},
"eventID": "a7654656-0417-45c6-9386-ea8231385051",
"eventType": "AwsConsoleSignin",
"recipientAccountId": "123456789012"
}
```

Access Management

AWS Identity and Access Management (IAM) is a web service that helps you securely control access to AWS resources. When a [principal \(p. 5\)](#) makes a request in AWS, the AWS enforcement code checks whether the principal is authenticated (signed in) and authorized (has permissions). You manage access in AWS by creating policies and attaching them to IAM identities or AWS resources. Policies are JSON documents in AWS that, when attached to an identity or resource, define their permissions. For more information about policy types and uses, see [Policies and Permissions \(p. 356\)](#).

For details about the rest of the authentication and authorization process, see [Understanding How IAM Works \(p. 3\)](#).



During authorization, the AWS enforcement code uses values from the [request context \(p. 5\)](#) to check for matching policies and determine whether to allow or deny the request.

AWS checks each policy that applies to the context of the request. If a single policy denies the request, AWS denies the entire request and stops evaluating policies. This is called an *explicit deny*. Because requests are *denied by default*, IAM authorizes your request only if every part of your request is allowed by the applicable policies. The [evaluation logic \(p. 636\)](#) for a request within a single account follows these rules:

- By default, all requests are implicitly denied. (Alternatively, by default, the AWS account root user has full access.)

- An explicit allow in an identity-based or resource-based policy overrides this default.
- If a permissions boundary, Organizations SCP, or session policy is present, it might override the allow with an implicit deny.
- An explicit deny in any policy overrides any allows.

After your request has been authenticated and authorized, AWS approves the request. If you need to make a request in a different account, a policy in the other account must allow you to access the resource. In addition, the IAM entity that you use to make the request must have an identity-based policy that allows the request.

Access Management Resources

For more information about permissions and about creating policies, see the following resources:

The following entries in the AWS Security Blog cover common ways to write policies for access to Amazon S3 buckets and objects.

- [Writing IAM Policies: How to Grant Access to an Amazon S3 Bucket](#)
- [Writing IAM policies: Grant Access to User-Specific Folders in an Amazon S3 Bucket](#)
- [IAM Policies and Bucket Policies and ACLs! Oh My! \(Controlling Access to S3 Resources\)](#)
- [A Primer on RDS Resource-Level Permissions](#)
- [Demystifying EC2 Resource-Level Permissions](#)

Policies and Permissions

You manage access in AWS by creating policies and attaching them to IAM identities (users, groups of users, or roles) or AWS resources. A policy is an object in AWS that, when associated with an identity or resource, defines their permissions. AWS evaluates these policies when an IAM principal (user or role) makes a request. Permissions in the policies determine whether the request is allowed or denied. Most policies are stored in AWS as JSON documents. AWS supports six types of policies: identity-based policies, resource-based policies, permissions boundaries, Organizations SCPs, ACLs, and session policies.

IAM policies define permissions for an action regardless of the method that you use to perform the operation. For example, if a policy allows the [GetUser](#) action, then a user with that policy can get user information from the AWS Management Console, the AWS CLI, or the AWS API. When you create an IAM user, you can choose to allow console or programmatic access. If console access is allowed, the IAM user can sign in to the console using a user name and password. Or if programmatic access is allowed, the user can use access keys to work with the CLI or API.

Policy Types

The following policy types, listed in order of frequency, are available for use in AWS. For more details, see the sections below for each policy type.

- **Identity-based policies (p. 357)** – Attach managed and inline policies to IAM identities (users, groups to which users belong, or roles). Identity-based policies grant permissions to an identity.
- **Resource-based policies (p. 357)** – Attach inline policies to resources. The most common examples of resource-based policies are Amazon S3 bucket policies and IAM role trust policies. Resource-based policies grant permissions to the principal that is specified in the policy. Principals can be in the same account as the resource or in other accounts.

- **Permissions boundaries (p. 358)** – Use a managed policy as the permissions boundary for an IAM entity (user or role). That policy defines the maximum permissions that the identity-based policies can grant to an entity, but does not grant permissions. Permissions boundaries do not define the maximum permissions that a resource-based policy can grant to an entity.
- **Organizations SCPs (p. 358)** – Use an AWS Organizations service control policy (SCP) to define the maximum permissions for account members of an organization or organizational unit (OU). SCPs limit permissions that identity-based policies or resource-based policies grant to entities (users or roles) within the account, but do not grant permissions.
- **Access control lists (ACLs) (p. 358)** – Use ACLs to control which principals in other accounts can access the resource to which the ACL is attached. ACLs are similar to resource-based policies, although they are the only policy type that does not use the JSON policy document structure. ACLs are cross-account permissions policies that grant permissions to the specified principal. ACLs cannot grant permissions to entities within the same account.
- **Session policies (p. 358)** – Pass advanced session policies when you use the AWS CLI or AWS API to assume a role or a federated user. Session policies limit the permissions that the role or user's identity-based policies grant to the session. Session policies limit permissions for a created session, but do not grant permissions. For more information, see [Session Policies](#).

Identity-Based Policies

Identity-based policies are JSON permissions policy documents that you can attach to an identity (user, group of users, or role). These policies control what actions an entity (user or role) can perform, on which resources, and under what conditions. Identity-based policies can be further categorized:

- **Managed policies** – Standalone identity-based policies that you can attach to multiple users, groups, and roles in your AWS account. You can use two types of managed policies:
 - **AWS managed policies** – Managed policies that are created and managed by AWS. If you are new to using policies, we recommend that you start by using AWS managed policies.
 - **Customer managed policies** – Managed policies that you create and manage in your AWS account. Customer managed policies provide more precise control over your policies than AWS managed policies. You can create and edit an IAM policy in the visual editor or by creating the JSON policy document directly. For more information, see [Creating IAM Policies \(p. 443\)](#) and [Editing IAM Policies \(p. 468\)](#).
- **Inline policies** – Policies that you create and manage and that are embedded directly into a single user, group, or role. In most cases, we don't recommend using inline policies.

To learn how to choose between a managed policy or an inline policy, see [Managed Policies and Inline Policies \(p. 364\)](#).

Resource-Based Policies

Resource-based policies are JSON policy documents that you attach to a resource such as an Amazon S3 bucket. These policies grant the specified principal permission to perform specific actions on that resource and defines under what conditions this applies. Resource-based policies are inline policies. There are no managed resource-based policies.

To enable cross-account access, you can specify an entire account or IAM entities in another account as the principal in a resource-based policy. Adding a cross-account principal to a resource-based policy is only half of establishing the trust relationship. When the principal and the resource are in separate AWS accounts, you must also use an identity-based policy to grant the principal access to the resource. However, if a resource-based policy grants access to a principal in the same account, no additional identity-based policy is required.

The IAM service supports only one type of resource-based policy called a role *trust policy*, which is attached to an IAM role. An IAM role is both an identity and a resource that supports resource-based

policies. For that reason, you must attach both a trust policy and an identity-based policy to an IAM role. Trust policies define which principal entities (accounts, users, roles, and federated users) can assume the role. To learn how IAM roles are different from other resource-based policies, see [How IAM Roles Differ from Resource-based Policies \(p. 292\)](#).

To see which other services support resource-based policies, see [AWS Services That Work with IAM \(p. 586\)](#). To learn more about resource-based policies, see [Identity-Based Policies and Resource-Based Policies \(p. 379\)](#). To learn whether principals in accounts outside of your zone of trust (trusted organization, OU, or account) have access to assume your roles, see [What is IAM Access Analyzer?](#).

IAM 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. When you set a permissions boundary for an entity, the entity can perform only the actions that are allowed by both its identity-based policies and its permissions boundaries. Resource-based policies that specify the user or role as the principal 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 \(p. 370\)](#).

Service Control Policies (SCPs)

AWS Organizations is a service for grouping and centrally managing the 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. SCPs are JSON policies that specify the maximum permissions for an organization or organizational unit (OU). The SCP limits permissions for entities in member accounts, including each AWS account root user. An explicit deny in any of these policies overrides the allow.

For more information about Organizations and SCPs, see [How SCPs Work in the AWS Organizations User Guide](#).

Access Control Lists (ACLs)

Access control lists (ACLs) are service policies that allow you to control which principals in another account can access a resource. ACLs cannot be used to control access for a principal within the same account. ACLs are similar to resource-based policies, although they are the only policy type that does 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](#).

Session Policies

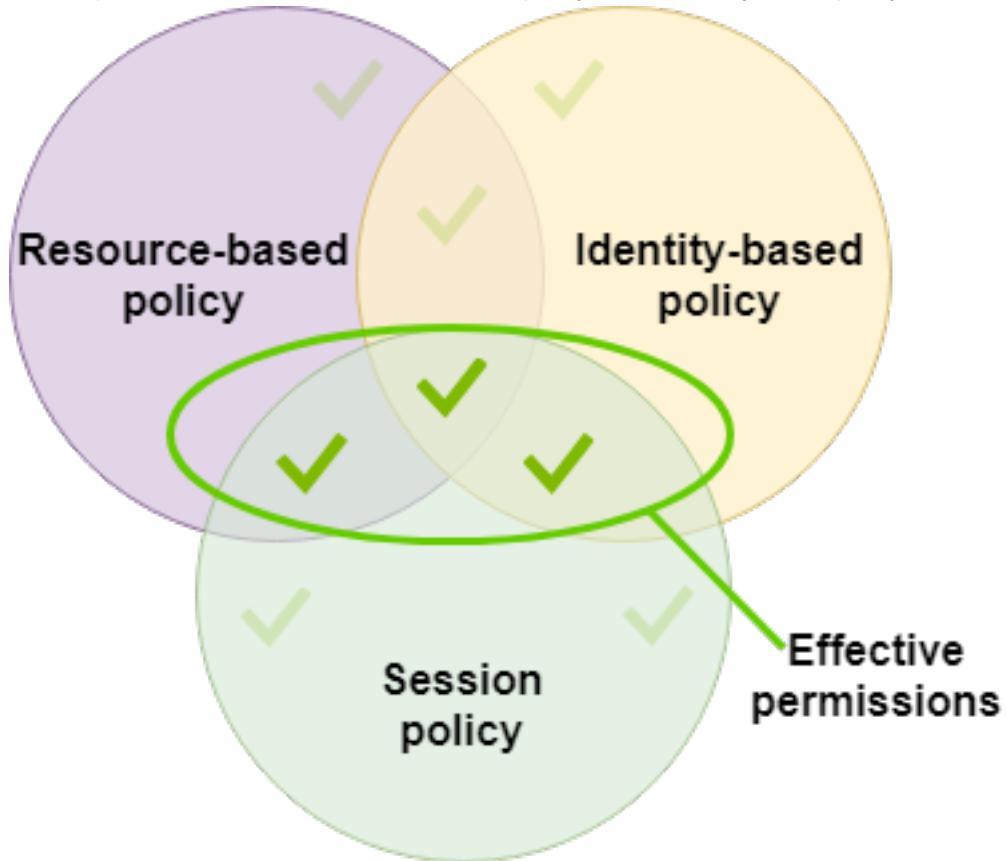
Session policies are advanced policies that you pass in a parameter when you programmatically create a temporary session for a role or federated user. The permissions for a session are the intersection of the identity-based policies for the IAM entity (user or role) used to create the session and the session policies. Permissions can also come from a resource-based policy. An explicit deny in any of these policies overrides the allow.

You can create role session and pass session policies programmatically using the `AssumeRole`, `AssumeRoleWithSAML`, or `AssumeRoleWithWebIdentity` API operations. You can pass a single JSON inline session policy document using the `Policy` parameter. You can use the `PolicyArns` parameter to specify up to 10 managed session policies. For more information about creating a role session, see [Requesting Temporary Security Credentials \(p. 309\)](#).

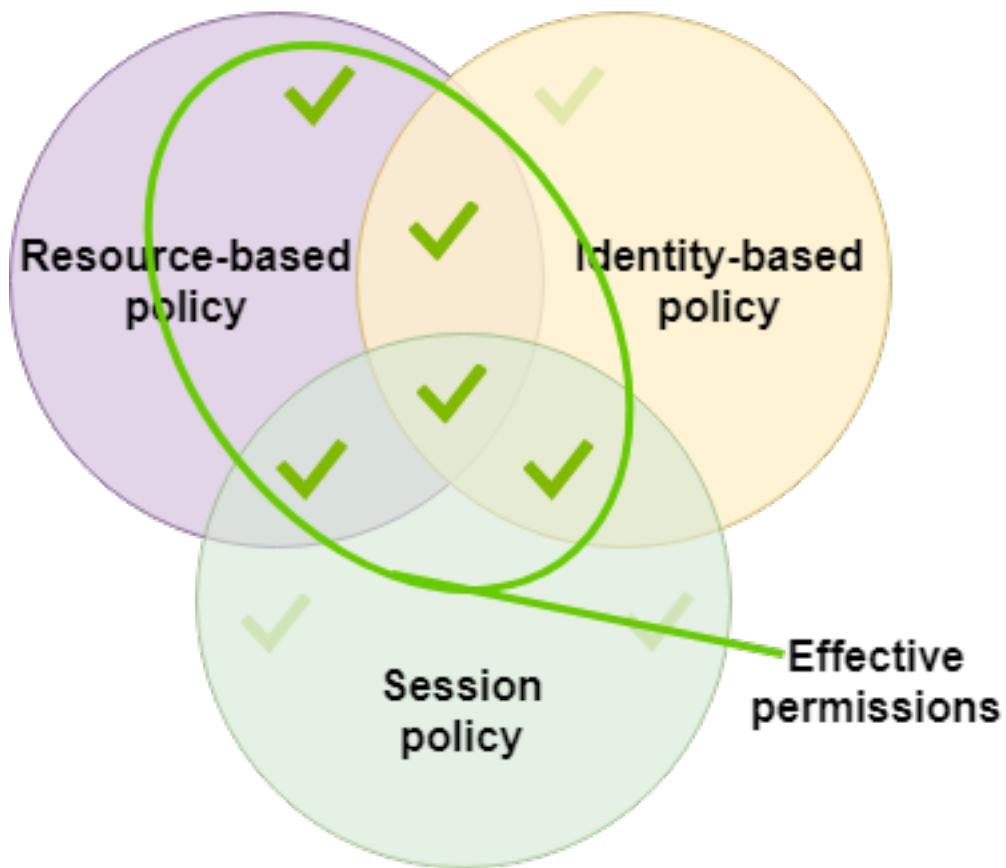
When you create a federated user session, you use an IAM user's access keys to programmatically call the `GetFederationToken` API operation. You must also pass session policies. The resulting session's

permissions are the intersection of the IAM user's identity-based policy and the session policy. For more information about creating a federated user session, see [GetFederationToken—Federation Through a Custom Identity Broker \(p. 314\)](#).

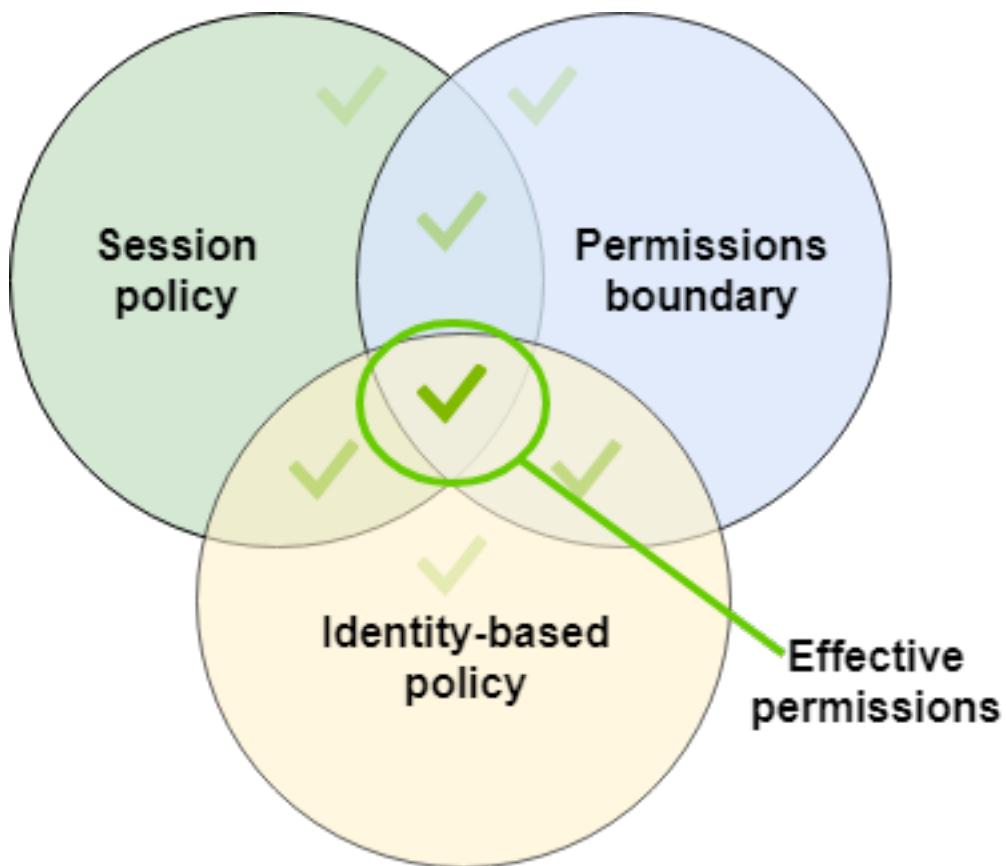
A resource-based policy can specify the ARN of the user or role as a principal. In that case, the permissions from the resource-based policy are added to the role or user's identity-based policy before the session is created. The session policy limits the total permissions granted by the resource-based policy and the identity-based policy. The resulting session's permissions are the intersection of the session policies and either the resource-based policy or the identity-based policy.



A resource-based policy can specify the ARN of the session as a principal. In that case, the permissions from the resource-based policy are added after the session is created. The resource-based policy permissions are not limited by the session policy. The resulting session has all the permissions of the resource-based policy *plus* the intersection of the identity-based policy and the session policy.



A permissions boundary can set the maximum permissions for a user or role that is used to create a session. In that case, the resulting session's permissions are the intersection of the session policy, the permissions boundary, and the identity-based policy. However, a permissions boundary does not limit permissions granted by a resource-based policy that specifies the ARN of the resulting session.



Policies and the Root User

The AWS account root user is affected by some policy types but not others. You cannot attach identity-based policies to the root user, and you cannot set the permissions boundary for the root user. However, you can specify the root user as the principal in a resource-based policy or an ACL. As a member of an account, the root user is affected by any SCPs for the account.

Overview of JSON Policies

Most policies are stored in AWS as JSON documents. Identity-based policies and policies used to set permissions boundaries are JSON policy documents that you attach to a user or role. Resource-based policies are JSON policy documents that you attach to a resource. SCPs are JSON policy documents with restricted syntax that you attach to an AWS Organizations organizational unit (OU). ACLs are also attached to a resource, but you must use a different syntax. Session policies are JSON policies that you provide when you assume a role or federated user session.

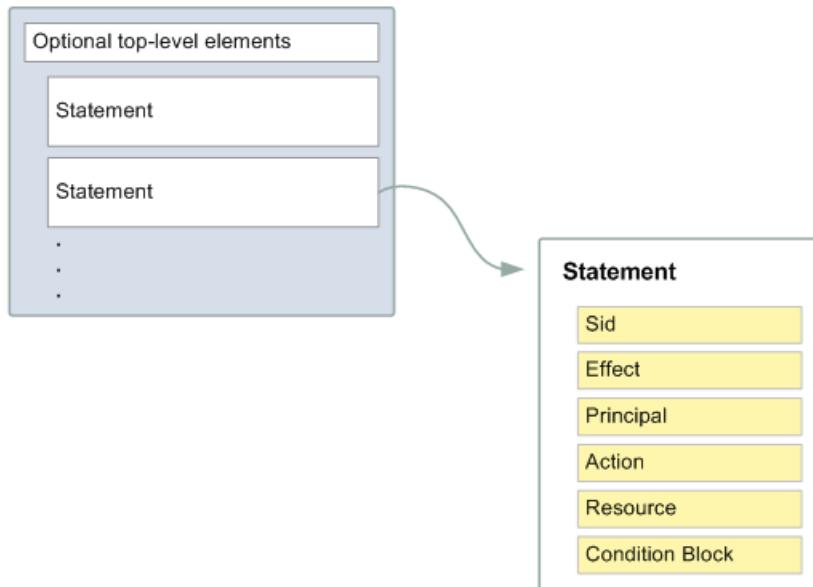
It is not necessary for you to understand the JSON syntax. You can use the visual editor in the AWS Management Console to create and edit customer managed policies without ever using JSON. However, if you use inline policies for groups or complex policies, you must still create and edit those policies in the JSON editor using the console. For more information about using the visual editor, see [Creating IAM Policies \(p. 443\)](#) and [Editing IAM Policies \(p. 468\)](#).

JSON Policy Document Structure

As illustrated in the following figure, a JSON policy document includes these elements:

- Optional policy-wide information at the top of the document
- One or more individual statements

Each statement includes information about a single permission. If a policy includes multiple statements, AWS applies a logical OR across the statements when evaluating them. If multiple policies apply to a request, AWS applies a logical OR across all of those policies when evaluating them.



The information in a statement is contained within a series of elements.

- **Version** – Specify the version of the policy language that you want to use. As a best practice, use the latest 2012-10-17 version.
- **Statement** – Use this main policy element as a container for the following elements. You can include more than one statement in a policy.
- **Sid** (Optional) – Include an optional statement ID to differentiate between your statements.
- **Effect** – Use Allow or Deny to indicate whether the policy allows or denies access.
- **Principal** (Required in only some circumstances) – If you create a resource-based policy, you must indicate the account, user, role, or federated user to which you would like to allow or deny access. If you are creating an IAM permissions policy to attach to a user or role, you cannot include this element. The principal is implied as that user or role.
- **Action** – Include a list of actions that the policy allows or denies.
- **Resource** (Required in only some circumstances) – If you create an IAM permissions policy, you must specify a list of resources to which the actions apply. If you create a resource-based policy, this element is optional. If you do not include this element, then the resource to which the action applies is the resource to which the policy is attached.
- **Condition** (Optional) – Specify the circumstances under which the policy grants permission.

To learn about these and other more advanced policy elements, see [IAM JSON Policy Elements Reference \(p. 598\)](#).

Multiple Statements and Multiple Policies

If you want to define more than one permission for an entity (user, group, or role), you can use multiple statements in a single policy. You can also attach multiple policies. If you try to define multiple permissions in a single statement, your policy might not grant the access that you expect. As a best practice, break up policies by resource type.

Because of the [limited size of policies \(p. 581\)](#), it might be necessary to use multiple policies for more complex permissions. It's also a good idea to create functional groupings of permissions in a separate customer managed policy. For example, Create one policy for IAM user management, one for self-management, and another policy for S3 bucket management. Regardless of the combination of multiple statements and multiple policies, AWS [evaluates \(p. 636\)](#) your policies the same way.

For example, the following policy has three statements, each of which defines a separate set of permissions within a single account. The statements define the following:

- The first statement, with an `Sid` (Statement ID) of `FirstStatement`, lets the user with the attached policy change their own password. The `Resource` element in this statement is `"*"` (which means "all resources"). But in practice, the `ChangePassword` API operation (or equivalent `change-password` CLI command) affects only the password for the user who makes the request.
- The second statement lets the user list all the Amazon S3 buckets in their AWS account. The `Resource` element in this statement is `"*"` (which means "all resources"). But because policies don't grant access to resources in other accounts, the user can list only the buckets in their own AWS account.
- The third statement lets the user list and retrieve any object that is in a bucket named `confidential-data`, but only when the user is authenticated with multi-factor authentication (MFA). The `Condition` element in the policy enforces the MFA authentication.

When a policy statement contains a `Condition` element, the statement is only in effect when the `Condition` element evaluates to true. In this case, the `Condition` evaluates to true when the user is MFA-authenticated. If the user is not MFA-authenticated, this `Condition` evaluates to false. In that case, the third statement in this policy does not apply and the user does not have access to the `confidential-data` bucket.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "FirstStatement",  
            "Effect": "Allow",  
            "Action": ["iam:ChangePassword"],  
            "Resource": "*"  
        },  
        {  
            "Sid": "SecondStatement",  
            "Effect": "Allow",  
            "Action": "s3>ListAllMyBuckets",  
            "Resource": "*"  
        },  
        {  
            "Sid": "ThirdStatement",  
            "Effect": "Allow",  
            "Action": [  
                "s3>List*",  
                "s3:Get*"  
            ],  
            "Resource": [  
                "arn:aws:s3:::confidential-data",  
                "arn:aws:s3:::confidential-data/*"  
            ]  
        }  
    ]  
}
```

```
        ],
        "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
    }
}
```

Examples of JSON Policy Syntax

The following identity-based policy allows the implied principal to list a single Amazon S3 bucket named `example_bucket`:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3>ListBucket",
            "Resource": "arn:aws:s3:::example_bucket"
        }
    ]
}
```

The following resource-based policy can be attached to an Amazon S3 bucket. The policy allows members of a specific AWS account to perform any Amazon S3 actions in the bucket named `mybucket`. It allows any action that can be performed on a bucket or the objects within it. (Because the policy grants trust only to the account, individual users in the account must still be granted permissions for the specified Amazon S3 actions.)

```
{
    "Version": "2012-10-17",
    "Id": "S3-Account-Permissions",
    "Statement": [
        {
            "Sid": "1",
            "Effect": "Allow",
            "Principal": {"AWS": ["arn:aws:iam::ACCOUNT-ID-WITHOUT-HYPHENS:root"]},
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::mybucket",
                "arn:aws:s3:::mybucket/*"
            ]
        }
    ]
}
```

To view example policies for common scenarios, see [Example IAM Identity-Based Policies \(p. 394\)](#).

Managed Policies and Inline Policies

When you need to set the permissions for an identity in IAM, you must decide whether to use an AWS managed policy, a customer managed policy, or an inline policy. The following sections provide more information about each of the types of identity-based policies and when to use them.

Topics

- [AWS Managed Policies \(p. 365\)](#)
- [Customer Managed Policies \(p. 366\)](#)
- [Inline Policies \(p. 367\)](#)
- [Choosing Between Managed Policies and Inline Policies \(p. 368\)](#)
- [Deprecated AWS Managed Policies \(p. 369\)](#)

AWS Managed Policies

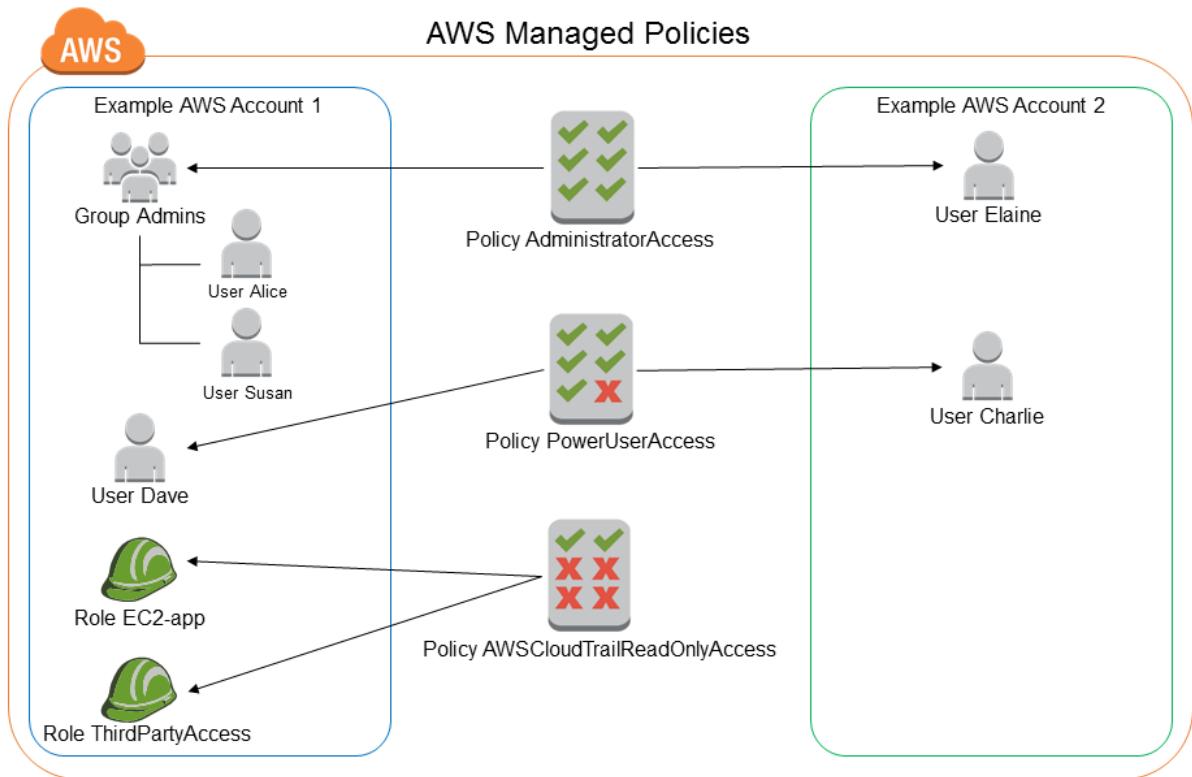
An *AWS managed policy* is a standalone policy that is created and administered by AWS. *Standalone policy* means that the policy has its own Amazon Resource Name (ARN) that includes the policy name. For example, `arn:aws:iam::aws:policy/IAMReadOnlyAccess` is an AWS managed policy. For more information about ARNs, see [Amazon Resource Names \(ARNs\) and AWS Service Namespaces](#).

AWS managed policies are designed to provide permissions for many common use cases. Full access AWS managed policies such as [AmazonDynamoDBFullAccess](#) and [IAMFullAccess](#) define permissions for service administrators by granting full access to a service. Power-user AWS managed policies such as [AWSCodeCommitPowerUser](#) and [AWSKeyManagementServicePowerUser](#) are designed for power users. Partial-access AWS managed policies such as [AmazonMobileAnalyticsWriteOnlyAccess](#) and [AmazonEC2ReadOnlyAccess](#) provide specific levels of access to AWS services without allowing permissions management permissions. 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.

One particularly useful category of AWS managed policies are those designed for job functions. These policies align closely to commonly used job functions in the IT industry. The intent is to make granting permissions for these common job functions easy. One key advantage of using job function policies is that they are maintained and updated by AWS as new services and API operations are introduced. For example, the [AdministratorAccess](#) job function provides full access and permissions delegation to every service and resource in AWS. We recommend that this policy is used only for the account administrator. For power users that require full access to every service except limited access to IAM and Organizations, use the [PowerUserAccess](#) job function. For a list and descriptions of the job function policies, see [AWS Managed Policies for Job Functions \(p. 656\)](#).

You cannot change the permissions defined in AWS managed policies. AWS occasionally updates the permissions defined in an AWS managed policy. When AWS does this, the update affects all principal entities (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 calls become available for existing services. For example, the AWS managed policy called **ReadOnlyAccess** provides read-only access to all AWS services and resources. When AWS launches a new service, AWS updates the **ReadOnlyAccess** policy to add read-only permissions for the new service. The updated permissions are applied to all principal entities that the policy is attached to.

The following diagram illustrates AWS managed policies. The diagram shows three AWS managed policies: **AdministratorAccess**, **PowerUserAccess**, and **AWSCloudTrailReadOnlyAccess**. Notice that a single AWS managed policy can be attached to principal entities in different AWS accounts, and to different principal entities in a single AWS account.



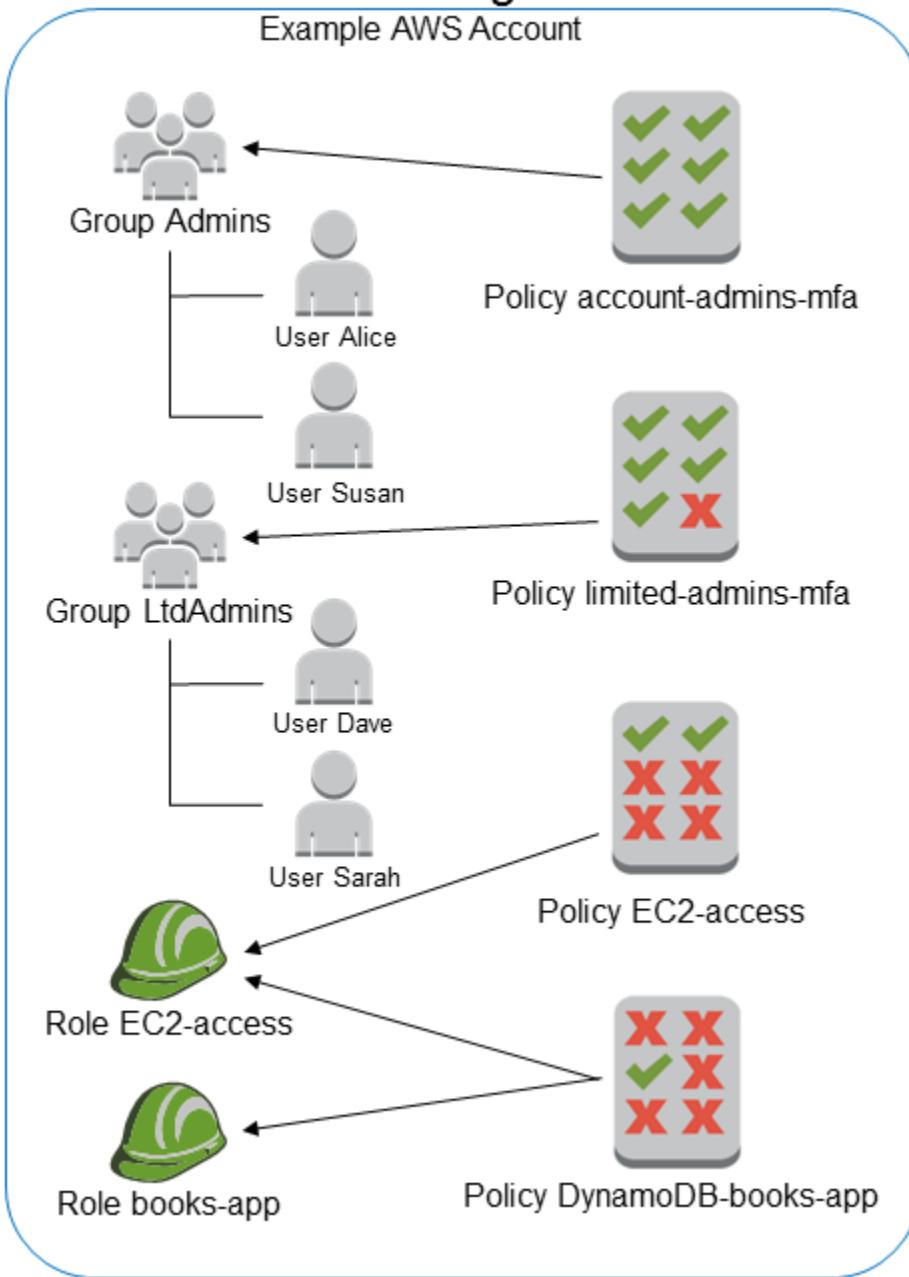
Customer Managed Policies

You can create standalone policies that you administer in your own AWS account, which we refer to as *customer managed policies*. You can then attach the policies to multiple principal entities in your AWS account. When you attach a policy to a principal entity, you give the entity the permissions that are defined in the policy.

A great way to create a customer managed policy is to start by copying an existing AWS managed policy. That way you know that the policy is correct at the beginning and all you need to do is customize it to your environment.

The following diagram illustrates customer managed policies. Each policy is an entity in IAM with its own [Amazon Resource Name \(ARN\)](#) that includes the policy name. Notice that the same policy can be attached to multiple principal entities—for example, the same **DynamoDB-books-app** policy is attached to two different IAM roles.

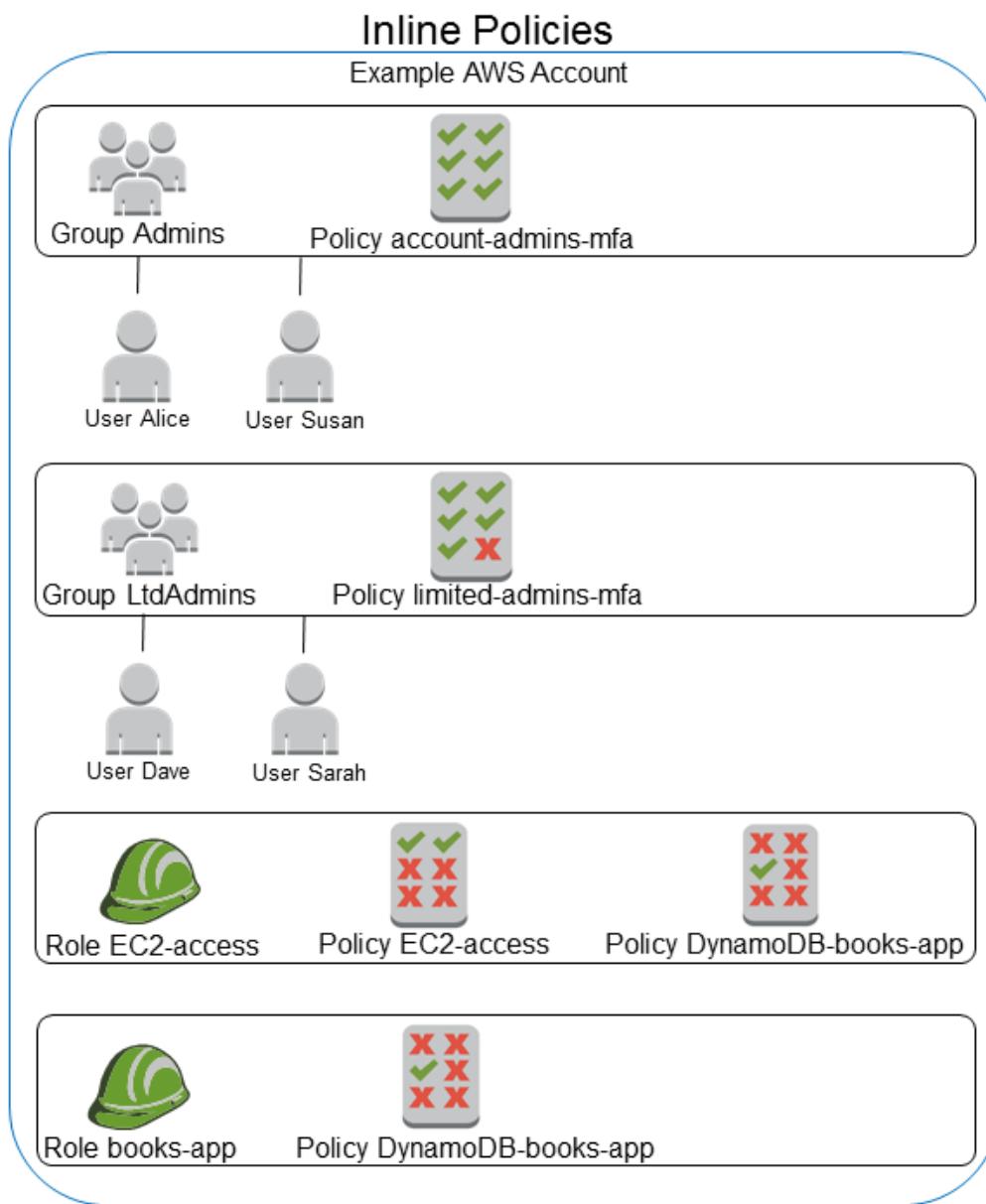
Customer Managed Policies



Inline Policies

An inline policy is a policy that's embedded in an IAM identity (a user, group, or role). That is, the policy is an inherent part of the identity. You can create a policy and embed it in a identity, either when you create the identity or later.

The following diagram illustrates inline policies. Each policy is an inherent part of the user, group, or role. Notice that two roles include the same policy (the **DynamoDB-books-app** policy), but they are not sharing a single policy; each role has its own copy of the policy.



Choosing Between Managed Policies and Inline Policies

The different types of policies are for different use cases. In most cases, we recommend that you use managed policies instead of inline policies.

Managed policies provide the following features:

Reusability

A single managed policy can be attached to multiple principal entities (users, groups, and roles). In effect, you can create a library of policies that define permissions that are useful for your AWS account, and then attach these policies to principal entities as needed.

Central change management

When you change a managed policy, the change is applied to all principal entities that the policy is attached to. For example, if you want to add permission for a new AWS API, you can update the managed policy to add the permission. (If you're using an AWS managed policy, AWS updates to the policy.) When the policy is updated, the changes are applied to all principal entities that the policy is attached to. In contrast, to change an inline policy you must individually edit each identity that contains the policy. For example, if a group and a role both contain the same inline policy, you must individually edit both principal entities in order to change that policy.

Versioning and rolling back

When you change a customer managed policy, the changed policy doesn't overwrite the existing policy. Instead, IAM creates a new version of the managed policy. IAM stores up to five versions of your customer managed policies. You can use policy versions to revert a policy to an earlier version if you need to.

A policy version is different from a `Version` policy element. The `Version` policy element is used within a policy and defines the version of the policy language. To learn more about policy versions, see [the section called "Versioning IAM Policies" \(p. 465\)](#). To learn more about the `Version` policy element see [IAM JSON Policy Elements: Version \(p. 598\)](#).

Delegating permissions management

You can allow users in your AWS account to attach and detach policies while maintaining control over the permissions defined in those policies. In effect, you can designate some users as full administrators—that is, administrators that can create, update, and delete policies. You can then designate other users as limited administrators. That is, administrators that can attach policies to other principal entities, but only the policies that you have allowed them to attach.

For more information about delegating permissions management, see [Controlling Access to Policies \(p. 385\)](#).

Automatic updates for AWS managed policies

AWS maintains AWS managed policies and updates them when necessary (for example, to add permissions for new AWS services), without you having to make changes. The updates are automatically applied to the principal entities that you have attached the AWS managed policy to.

Using Inline Policies

Inline policies are useful if you want to maintain a strict one-to-one relationship between a policy and the identity that it's applied to. For example, you want to be sure that the permissions in a policy are not inadvertently assigned to an identity other than the one they're intended for. When you use an inline policy, the permissions in the policy cannot be inadvertently attached to the wrong identity. In addition, when you use the AWS Management Console to delete that identity, the policies embedded in the identity are deleted as well. That's because they are part of the principal entity.

Deprecated AWS Managed Policies

To simplify the assignment of permissions, AWS provides [managed policies \(p. 364\)](#)—predefined policies that are ready to be attached to your IAM users, groups, and roles.

Sometimes AWS needs to add a new permission to an existing policy, such as when a new service is introduced. Adding a new permission to an existing policy does not disrupt or remove any feature or ability.

However, AWS might choose to create a *new* policy when the needed changes could impact customers if they were applied to an existing policy. For example, removing permissions from an existing policy could

break the permissions of any IAM entity or application that depended upon it, potentially disrupting a critical operation.

Therefore, when such a change is required, AWS creates a completely new policy with the required changes and makes it available to customers. The old policy is then marked *deprecated*. A deprecated managed policy appears with a warning icon next to it in the **Policies** list in the IAM console.

A deprecated policy has the following characteristics:

- It continues to work for all *currently* attached users, groups, and roles. Nothing breaks.
- It *cannot* be attached to any new users, groups, or roles. If you detach it from a current entity, you cannot reattach it.
- After you detach it from all current entities, it is no longer visible and can no longer be used in any way.

If any user, group, or role requires the policy, you must instead attach the new policy. When you receive notice that a policy is deprecated, we recommend that you immediately plan to attach all users, groups, and roles to the replacement policy and detach them from the deprecated policy. Continuing to use the deprecated policy can carry risks that are mitigated only by switching to the replacement policy.

Permissions Boundaries for IAM Entities

AWS supports *permissions boundaries* for IAM entities (users or roles). A permissions boundary is an advanced feature for using a managed policy to set the maximum permissions that an identity-based policy can grant to an IAM entity. An entity's permissions boundary allows it to perform only the actions that are allowed by both its identity-based policies and its permissions boundaries.

For more information about policy types, see [Policy Types \(p. 356\)](#).

You can use an AWS managed policy or a customer managed policy to set the boundary for an IAM entity (user or role). That policy limits the maximum permissions for the user or role.

For example, assume that the IAM user named ShirleyRodriguez should be allowed to manage only Amazon S3, Amazon CloudWatch, and Amazon EC2. To enforce this rule, you can use the following policy to set the permissions boundary for the ShirleyRodriguez user:

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": [  
                "s3:*",  
                "cloudwatch:*",  
                "ec2:*"  
            ],  
            "Resource": "*"  
        }  
    ]  
}
```

When you use a policy to set the permissions boundary for a user, it limits the user's permissions but does not provide permissions on its own. In this example, the policy sets the maximum permissions of ShirleyRodriguez as all operations in Amazon S3, CloudWatch, and Amazon EC2. Shirley can never perform operations in any other service, including IAM, even if she has a permissions policy that allows it. For example, you can add the following policy to the ShirleyRodriguez user:

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": "iam>CreateUser",  
            "Resource": "*"  
        }  
    ]  
}
```

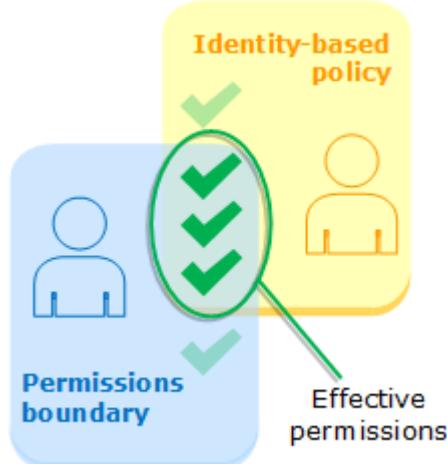
This policy allows creating a user in IAM. If you attach this permissions policy to the ShirleyRodriguez user, and Shirley tries to create a user, the operation fails. It fails because the permissions boundary does not allow the `iam:CreateUser` operation. Given these two policies, Shirley does not have permission to perform any operations in AWS. You must add a different permissions policy to allow actions in other services, such as Amazon S3. Alternatively, you could update the permissions boundary to allow her to create a user in IAM.

Evaluating Effective Permissions with Boundaries

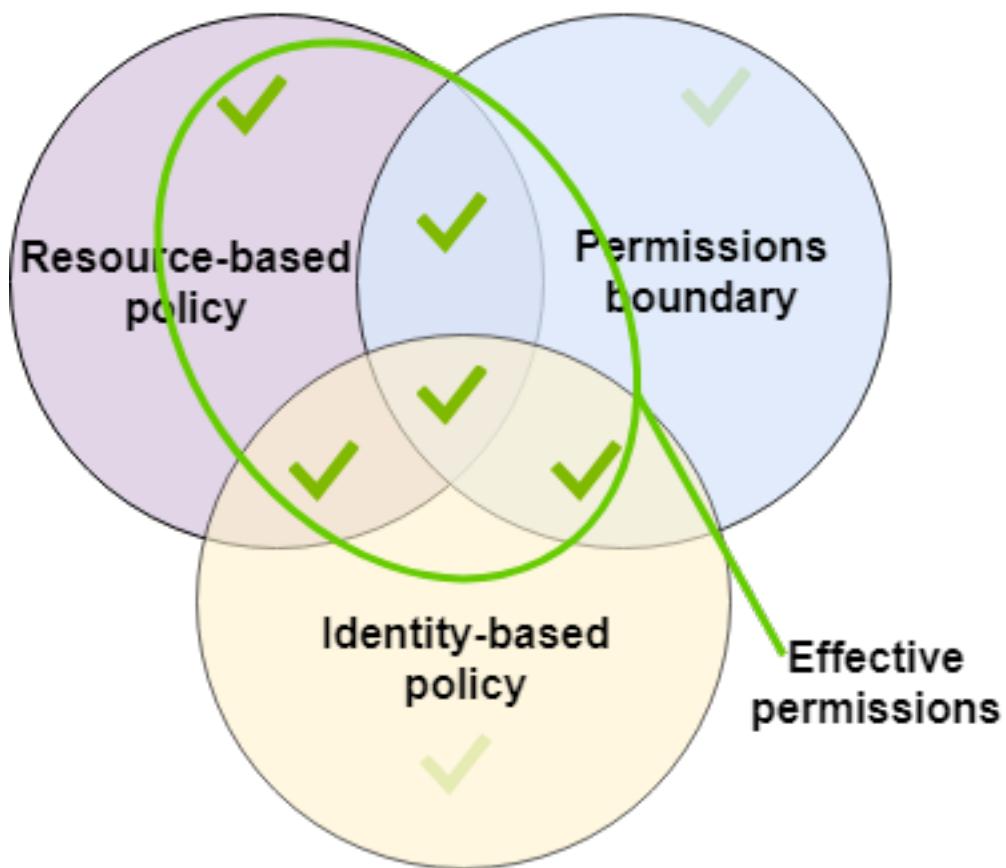
The permissions boundary for an IAM entity (user or role) sets the maximum permissions that the entity can have. This can change the effective permissions for that user or role. The effective permissions for an entity are the permissions that are granted by all the policies that affect the user or role. Within an account, the permissions for an entity can be affected by identity-based policies, resource-based policies, permissions boundaries, Organizations SCPs, or session policies. For more information about the different types of policies, see [Policies and Permissions \(p. 356\)](#).

If any one of these policy types explicitly denies access for an operation, then the request is denied. The permissions granted to an entity by multiple permissions types are more complex. For more details about how AWS evaluates policies, see [Policy Evaluation Logic \(p. 636\)](#).

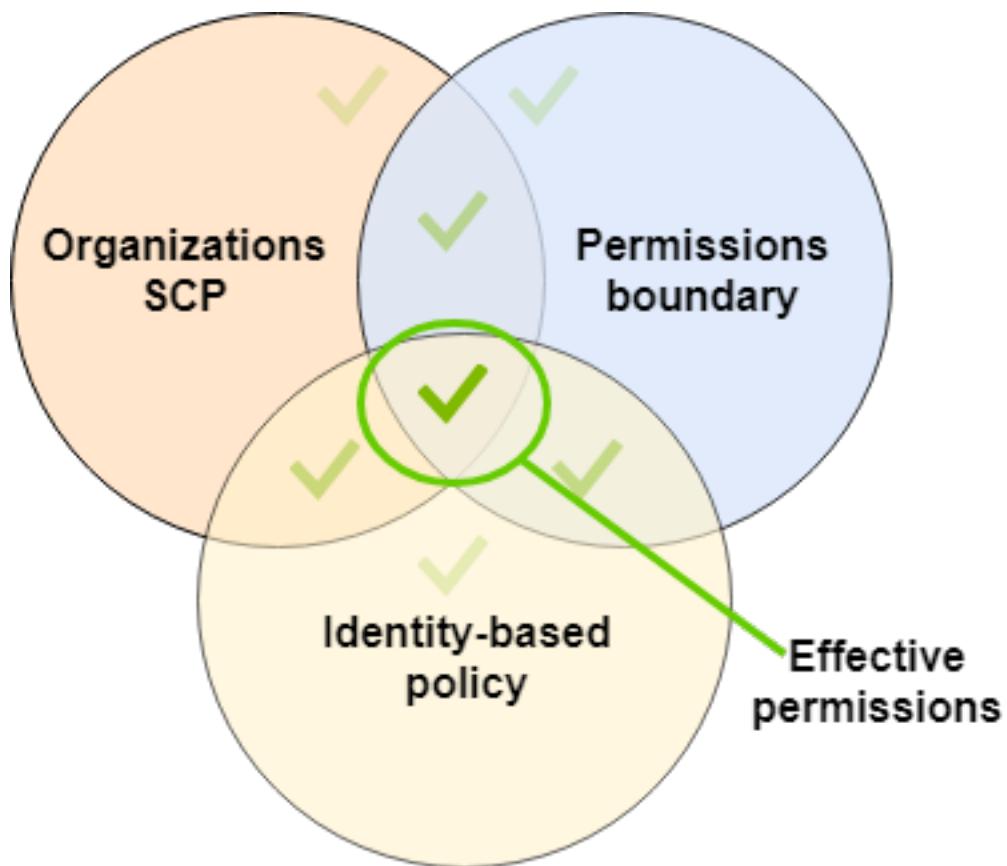
Identity-based policies with boundaries – Identity-based policies are inline or managed policies that are attached to a user, group of users, or role. Identity-based policies grant permission to the entity, and permissions boundaries limit those permissions. The effective permissions are the intersection of both policy types. An explicit deny in either of these policies overrides the allow.



Resource-based policies – Resource-based policies control how the specified principal can access the resource to which the policy is attached. Within an account, an implicit deny in a permissions boundary does not limit the permissions granted by a resource-based policy. Permissions boundaries reduce permissions that are granted to an entity by identity-based policies, and then resource-based policies provide additional permissions to the entity. In this case, the effective permissions are everything that is allowed by the resource-based policy *and* the intersection of the permissions boundary and the identity-based policy. An explicit deny in any of these policies overrides the allow.

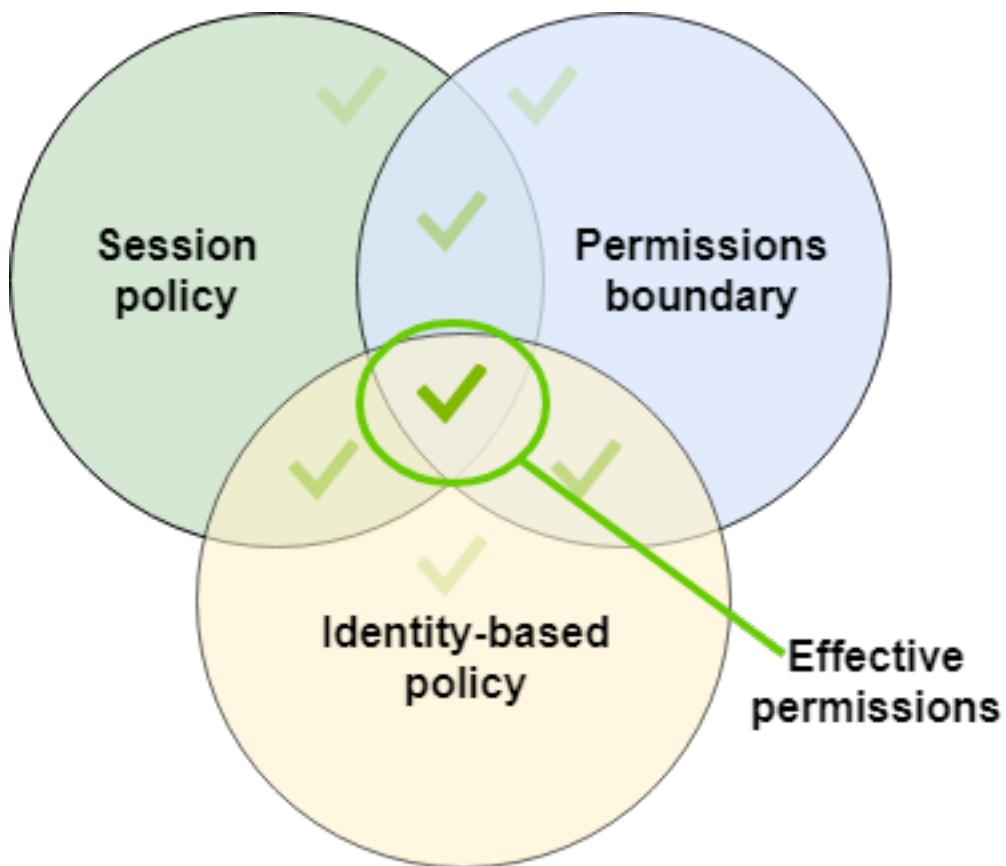


Organizations SCPs – SCPs are applied to an entire AWS account. They limit permissions for every request made by a principal within the account. An IAM entity (user or role) can make a request that is affected by an SCP, a permissions boundary, and an identity-based policy. In this case, the request is allowed only if all three policy types allow it. The effective permissions are the intersection of all three policy types. An explicit deny in any of these policies overrides the allow.



You can learn [whether your account is a member of an organization](#) in AWS Organizations. Organization members might be affected by an SCP. To view this data using the AWS CLI command or AWS API operation, you must have permissions for the `organizations:DescribeOrganization` action for your Organizations entity. You must have additional permissions to perform the operation in the Organizations console. To learn whether an SCP is denying access to a specific request, or to change your effective permissions, contact your AWS Organizations administrator.

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 permissions for a session come from the IAM entity (user or role) used to create the session and from the session policy. The entity's identity-based policy permissions are limited by the session policy and the permissions boundary. The effective permissions for this set of policy types are the intersection of all three policy types. An explicit deny in any of these policies overrides the allow. For more information about session policies, see [Session Policies](#).



Delegating Responsibility to Others Using Permissions Boundaries

You can use permissions boundaries to delegate permissions management tasks, such as user creation, to IAM users in your account. This permits others to perform tasks on your behalf within a specific boundary of permissions.

For example, assume that María is the administrator of the X-Company AWS account. She wants to delegate user creation duties to Zhang. However, she must ensure that Zhang creates users that adhere to the following company rules:

- Users cannot use IAM to create or manage users, groups, roles, or policies.
- Users are denied access to the Amazon S3 logs bucket and cannot access the i-1234567890abcdef0 Amazon EC2 instance.
- Users cannot remove their own boundary policies.

To enforce these rules, María completes the following tasks, for which details are included below:

1. María creates the `xCompanyBoundaries` managed policy to use as a permissions boundary for all new users in the account.
2. María creates the `DelegatedUserBoundary` managed policy and assigns it as the permissions boundary for Zhang. María makes a note of her admin IAM user's ARN and uses it in the policy to prevent Zhang from accessing it.
3. María creates the `DelegatedUserPermissions` managed policy and attaches it as a permissions policy for Zhang.

4. María tells Zhang about his new responsibilities and limitations.

Task 1: María must first create a managed policy to define the boundary for the new users. María will allow Zhang to give users the permissions policies they need, but she wants those users to be restricted. To do this, she creates the following customer managed policy with the name `XCompanyBoundaries`. This policy does the following:

- Allows users full access to several services
- Allows limited self-managing access in the IAM console. This means they can change their password after signing into the console. They can't set their initial password. To allow this, add the `"*LoginProfile"` action to the `AllowManageOwnPasswordAndAccessKeys` statement.
- Denies users access to the Amazon S3 logs bucket or the `i-1234567890abcdef0` Amazon EC2 instance

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ServiceBoundaries",
            "Effect": "Allow",
            "Action": [
                "s3:*",
                "cloudwatch: *",
                "ec2: *",
                "dynamodb: *"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowIAMConsoleForCredentials",
            "Effect": "Allow",
            "Action": [
                "iam>ListUsers",
                "iam:GetAccountPasswordPolicy"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnPasswordAndAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam>*AccessKey*",
                "iam:ChangePassword",
                "iam:GetUser",
                "iam>*ServiceSpecificCredential*",
                "iam>*SigningCertificate*"
            ],
            "Resource": ["arn:aws:iam:::user/${aws:username}"]
        },
        {
            "Sid": "DenyS3Logs",
            "Effect": "Deny",
            "Action": "s3: *",
            "Resource": [
                "arn:aws:s3:::logs",
                "arn:aws:s3:::logs/*"
            ]
        },
        {
            "Sid": "DenyEC2Production",
            "Effect": "Deny",
            "Action": "ec2: *",
            "Resource": [
                "arn:aws:ec2:::instance/i-1234567890abcdef0"
            ]
        }
    ]
}
```

```

        "Action": "ec2:*",
        "Resource": "arn:aws:ec2:***:instance/i-1234567890abcdef0"
    }
]
}

```

Each statement serves a different purpose:

1. The `ServiceBoundaries` statement of this policy allows full access to the specified AWS services. This means that a new user's actions in these services are limited only by the permissions policies that are attached to the user.
2. The `AllowIAMConsoleForCredentials` statement allows access to list all IAM users. This access is necessary to navigate the **Users** page in the AWS Management Console. It also allows viewing the password requirements for the account, which is necessary when changing your own password.
3. The `AllowManageOwnPasswordAndAccessKeys` statement allows the users manage only their own console password and programmatic access keys. This is important if Zhang or another administrator gives a new user a permissions policy with full IAM access. In that case, that user could then change their own or other users' permissions. This statement prevents that from happening.
4. The `DenyS3Logs` statement explicitly denies access to the logs bucket.
5. The `DenyEC2Production` statement explicitly denies access to the `i-1234567890abcdef0` instance.

Task 2: María wants to allow Zhang to create all X-Company users, but only with the `XCompanyBoundaries` permissions boundary. She creates the following customer managed policy named `DelegatedUserBoundary`. This policy defines the maximum permissions that Zhang can have.

```

{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "CreateOrChangeOnlyWithBoundary",
            "Effect": "Allow",
            "Action": [
                "iam:CreateUser",
                "iam:DeleteUserPolicy",
                "iam:AttachUserPolicy",
                "iam:DetachUserPolicy",
                "iam:PutUserPermissionsBoundary",
                "iam:PutUserPolicy"
            ],
            "Resource": "*",
            "Condition": {"StringEquals":
                {"iam:PermissionsBoundary": "arn:aws:iam::123456789012:policy/
XCompanyBoundaries"}}
        },
        {
            "Sid": "CloudWatchAndOtherIAMTasks",
            "Effect": "Allow",
            "Action": [
                "cloudwatch:*",
                "iam:GetUser",
                "iam>ListUsers",
                "iam>DeleteUser",
                "iam:UpdateUser",
                "iam>CreateAccessKey",
                "iam>CreateLoginProfile",
                "iam:GetAccountPasswordPolicy",
                "iam:GetLoginProfile",
                "iam>ListGroups",
                "iam>ListGroupsForUser",
                "iam:ListUsers"
            ]
        }
    ]
}

```

```

        "iam:CreateGroup",
        "iam:GetGroup",
        "iam:DeleteGroup",
        "iam:UpdateGroup"
        "iam:CreatePolicy",
        "iam:DeletePolicy",
        "iam:DeletePolicyVersion",
        "iam:GetPolicy",
        "iam:GetPolicyVersion",
        "iam:GetUserPolicy",
        "iam:GetRolePolicy",
        "iam>ListPolicies",
        "iam>ListPolicyVersions",
        "iam>ListEntitiesForPolicy",
        "iam>ListUserPolicies",
        "iam>ListAttachedUserPolicies",
        "iam>ListRolePolicies",
        "iam>ListAttachedRolePolicies",
        "iam:SetDefaultPolicyVersion",
        "iam:SimulatePrincipalPolicy",
        "iam:SimulateCustomPolicy"
    ],
    "NotResource": "arn:aws:iam::123456789012:user/Maria"
},
{
    "Sid": "NoBoundaryPolicyEdit",
    "Effect": "Deny",
    "Action": [
        "iam:CreatePolicyVersion",
        "iam:DeletePolicy",
        "iam:DeletePolicyVersion",
        "iam:SetDefaultPolicyVersion"
    ],
    "Resource": [
        "arn:aws:iam::123456789012:policy/XCompanyBoundaries",
        "arn:aws:iam::123456789012:policy/DelegatedUserBoundary"
    ]
},
{
    "Sid": "NoBoundaryUserDelete",
    "Effect": "Deny",
    "Action": "iam>DeleteUserPermissionsBoundary",
    "Resource": "*"
}
]
}

```

Each statement serves a different purpose:

1. The `CreateOrChangeOnlyWithBoundary` statement allows Zhang to create IAM users but only if he uses the `XCompanyBoundaries` policy to set the permissions boundary. This statement also allows him to set the permissions boundary for existing users but only using that same policy. Finally, this statement allows Zhang to manage permissions policies for users with this permissions boundary set.
2. The `CloudWatchAndOtherIAMTasks` statement allows Zhang to complete other user, group, and policy management tasks. He has permissions to reset passwords and create access keys for any IAM user not listed in the condition key. This allows him to help users with sign-in issues.
3. The `NoBoundaryPolicyEdit` statement denies Zhang access to update the `XCompanyBoundaries` policy. He is not allowed to change any policy that is used to set the permissions boundary for himself or other users.
4. The `NoBoundaryUserDelete` statement denies Zhang access to delete the permissions boundary for himself or other users.

María then assigns the `DelegatedUserBoundary` policy [as the permissions boundary \(p. 97\)](#) for the Zhang user.

Task 3: Because the permissions boundary limits the maximum permissions, but does not grant access on its own, María must create a permissions policy for Zhang. She creates the following policy named `DelegatedUserPermissions`. This policy defines the operations that Zhang can perform, within the defined boundary.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "IAM",  
            "Effect": "Allow",  
            "Action": "iam:*",  
            "Resource": "*"  
        },  
        {  
            "Sid": "CloudWatchLimited",  
            "Effect": "Allow",  
            "Action": [  
                "cloudwatch:GetDashboard",  
                "cloudwatch:GetMetricData",  
                "cloudwatch>ListDashboards",  
                "cloudwatch:GetMetricStatistics",  
                "cloudwatch>ListMetrics"  
            ],  
            "Resource": "*"  
        },  
        {  
            "Sid": "S3BucketContents",  
            "Effect": "Allow",  
            "Action": "s3>ListBucket",  
            "Resource": "arn:aws:s3:::ZhangBucket"  
        }  
    ]  
}
```

Each statement serves a different purpose:

1. The `IAM` statement of the policy allows Zhang full access to IAM. However, because his permissions boundary allows only some IAM operations, his effective IAM permissions are limited only by his permissions boundary.
2. The `CloudWatchLimited` statement allows Zhang to perform five actions in CloudWatch. His permissions boundary allows all actions in CloudWatch, so his effective CloudWatch permissions are limited only by his permissions policy.
3. The `S3BucketContents` statement allows Zhang to list the `ZhangBucket` Amazon S3 bucket. However, his permissions boundary does not allow any Amazon S3 action, so he cannot perform any S3 operations, regardless of his permissions policy.

Note

Zhang's policies allow him to create a user that can then access Amazon S3 resources that he can't access. By delegating these administrative actions, María effectively trusts Zhang with access to Amazon S3.

María then attaches the `DelegatedUserPermissions` policy as the permissions policy for the Zhang user.

Task 4: She gives Zhang instructions to create a new user. She tells him that he can create new users with any permissions that they need, but he must assign them the `XCompanyBoundaries` policy as a permissions boundary.

Zhang completes the following tasks:

1. Zhang [creates a user \(p. 87\)](#) with the AWS Management Console. He types the user name Nikhil and enables console access for the user. He clears the checkbox next to **Requires password reset**, because the policies above allow Zhang to change his password only after he is signed in to the IAM console.
2. On the **Set permissions** page, Zhang chooses the `IAMFullAccess` and `AmazonS3ReadOnlyAccess` permissions policies that allow Nikhil to do his work.
3. Zhang skips the **Set permissions boundary** section, forgetting María's instructions.
4. Zhang reviews the user details and chooses **Create user**.

The operation fails and access is denied. Zhang's `DelegatedUserBoundary` permissions boundary requires that any user he creates have the `XCompanyBoundaries` policy used as a permissions boundary.

5. Zhang returns to the previous page. In the **Set permissions boundary** section, he chooses the `XCompanyBoundaries` policy.
6. Zhang reviews the user details and chooses **Create user**.

The user is created.

When Nikhil signs in, he has access to IAM and Amazon S3, except those operations that are denied by the permissions boundary. For example, he can change his own password in IAM but can't create another user or edit his policies. Nikhil has read-only access to Amazon S3.

If someone adds a resource-based policy to the `logs` bucket that allows Nikhil to put an object in the bucket, he still cannot access the bucket. The reason is that any actions on the `logs` bucket are explicitly denied by his permissions boundary. An explicit deny in any policy type results in a request being denied. However, if a resource-based policy attached to a Secrets Manager secret allows Nikhil to perform the `secretsmanager:GetSecretValue` action, then Nikhil can retrieve and decrypt the secret. The reason is that Secrets Manager operations are not explicitly denied by his permissions boundary, and implicit denies in permissions boundaries do not limit resource-based policies.

Identity-Based Policies and Resource-Based Policies

A policy is an object in AWS that, when associated with an identity or resource, defines their permissions. When you create a permissions policy to restrict access to a resource, you can choose an *identity-based policy* or a *resource-based policy*.

Identity-based policies are attached to an IAM user, group, or role. These policies let you specify what that identity can do (its permissions). For example, you can attach the policy to the IAM user named John, stating that he is allowed to perform the Amazon EC2 `RunInstances` action. The policy could further state that John is allowed to get items from an Amazon DynamoDB table named `MyCompany`. You can also allow John to manage his own IAM security credentials. Identity-based policies can be [managed or inline \(p. 364\)](#).

Resource-based policies are attached to a resource. For example, you can attach resource-based policies to Amazon S3 buckets, Amazon SQS queues, and AWS Key Management Service encryption keys. For a list of services that support resource-based policies, see [AWS Services That Work with IAM \(p. 586\)](#).

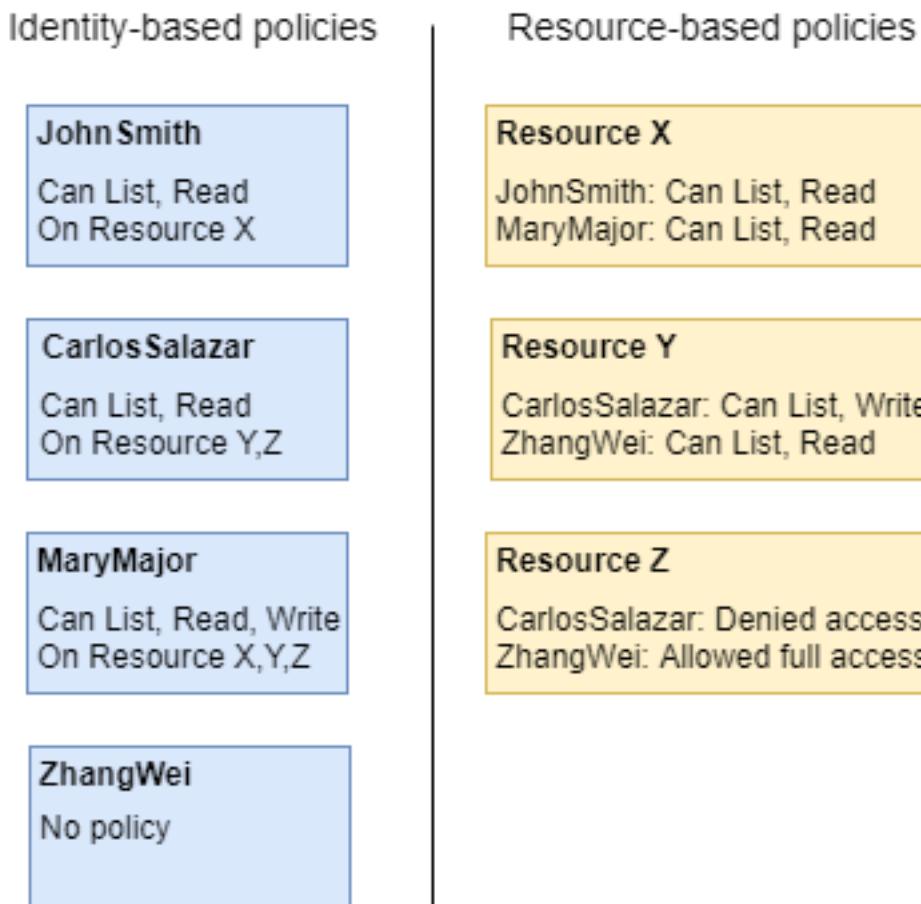
With resource-based policies, you can specify who has access to the resource and what actions they can perform on it. To learn whether principals in accounts outside of your zone of trust (trusted organization, OU, or account) have access to assume your roles, see [What is IAM Access Analyzer?](#). Resource-based policies are inline only, not managed.

Note

Resource-based policies differ from *resource-level* permissions. You can attach resource-based policies directly to a resource, as described in this topic. Resource-level permissions refer to the ability to use ARNs to specify individual resources in a policy. Resource-based policies are supported only by some AWS services. For a list of which services support resource-based policies and resource-level permissions, see [AWS Services That Work with IAM \(p. 586\)](#).

To better understand these concepts, view the following figure. The administrator of the 123456789012 account attached *identity-based policies* to the JohnSmith, CarlosSalazar, and MaryMajor users. Some of the actions in these policies can be performed on specific resources. For example, the user JohnSmith can perform some actions on Resource X. This is a *resource-level permission* in an identity-based policy. The administrator also added *resource-based policies* to Resource X, Resource Y, and Resource Z. Resource-based policies allow you to specify who can access that resource. For example, the resource-based policy on Resource X allows the JohnSmith and MaryMajor users list and read access to the resource.

Account ID: 123456789012



The 123456789012 account example allows the following users to perform the listed actions:

- **JohnSmith** – John can perform list and read actions on Resource X. He is granted this permission by the identity-based policy on his user and the resource-based policy on Resource X.
- **CarlosSalazar** – Carlos can perform list, read, and write actions on Resource Y, but is denied access to Resource Z. The identity-based policy on Carlos allows him to perform list and read actions on Resource Y. The Resource Y resource-based policy also allows him write permissions. However,

although his identity-based policy allows him access to Resource Z, the Resource Z resource-based policy denies that access. An explicit Deny overrides an Allow and his access to Resource Z is denied. For more information, see [Policy Evaluation Logic \(p. 636\)](#).

- **MaryMajor** – Mary can perform list, read, and write operations on Resource X, Resource Y, and Resource Z. Her identity-based policy allows her more actions on more resources than the resource-based policies, but none of them deny access.
- **ZhangWei** – Zhang has full access to Resource Z. Zhang has no identity-based policies, but the Resource Z resource-based policy allows him full access to the resource. Zhang can also perform list and read actions on Resource Y.

Identity-based policies and resource-based policies are both permissions policies and are evaluated together. For a request to which only permissions policies apply, AWS first checks all policies for a Deny. If one exists, then the request is denied. Then AWS checks for each Allow. If at least one policy statement allows the action in the request, the request is allowed. It doesn't matter whether the Allow is in the identity-based policy or the resource-based policy.

Important

This logic applies only when the request is made within a single AWS account. For requests made from one account to another, the requester in Account A must have an identity-based policy that allows them to make a request to the resource in Account B. Also, the resource-based policy in Account B must allow the requester in Account A to access the resource. If policies in both accounts don't allow the operation, the request fails. For more information about using resource-based policies for cross-account access, see [How IAM Roles Differ from Resource-based Policies \(p. 292\)](#).

A user who has specific permissions might request a resource that also has a permissions policy attached to it. In that case, AWS evaluates both sets of permissions when determining whether to grant access to the resource. For information about how policies are evaluated, see [Policy Evaluation Logic \(p. 636\)](#).

Note

Amazon S3 supports identity-based policies and resource-based policies (referred to as *bucket policies*). In addition, Amazon S3 supports a permission mechanism known as an *access control list (ACL)* that is independent of IAM policies and permissions. You can use IAM policies in combination with Amazon S3 ACLs. For more information, see [Access Control](#) in the *Amazon Simple Storage Service Developer Guide*.

Controlling Access Using Policies

You can use a policy to control access to resources within IAM or all of AWS.

To use a [policy \(p. 356\)](#) to control access in AWS, you must understand how AWS grants access. AWS is composed of collections of *resources*. An IAM user is a resource. An Amazon S3 bucket is a resource. When you use the AWS API, the AWS CLI, or the AWS Management Console to perform an operation (such as creating a user), you send a *request* for that operation. Your request specifies an action, a resource, a *principal entity* (user or role), a *principal account*, and any necessary request information. All of this information provides *context*.

AWS then checks that you (the principal) are authenticated (signed in) and authorized (have permission) to perform the specified action on the specified resource. During authorization, AWS checks all the policies that apply to the context of your request. Most policies are stored in AWS as [JSON documents \(p. 361\)](#) and specify the permissions for principal entities. For more information about policy types and uses, see [Policies and Permissions \(p. 356\)](#).

AWS authorizes the request only if each part of your request is allowed by the policies. To view a diagram of this process, see [Understanding How IAM Works \(p. 3\)](#). For details about how AWS determines whether a request is allowed, see [Policy Evaluation Logic \(p. 636\)](#).

When you create an IAM policy, you can control access to the following:

- **Principals (p. 382)** – Control what the person making the request (the principal (p. 5)) is allowed to do.
- **IAM Identities (p. 383)** – Control which IAM identities (groups, users, and roles) can be accessed and how.
- **IAM Policies (p. 385)** – Control who can create, edit, and delete customer managed policies, and who can attach and detach all managed policies.
- **AWS Resources (p. 388)** – Control who has access to resources using an identity-based policy or a resource-based policy.
- **AWS Accounts (p. 389)** – Control whether a request is allowed only for members of a specific account.

Policies let you specify who has access to AWS resources, and what actions they can perform on those resources. Every IAM user starts with no permissions. In other words, by default, users can do nothing, not even view their own access keys. To give a user permission to do something, you can add the permission to the user (that is, attach a policy to the user). Or you can add the user to a group that has the intended permission.

For example, you might grant a user permission to list his or her own access keys. You might also expand that permission and also let each user create, update, and delete their own keys.

When you give permissions to a group, all users in that group get those permissions. For example, you can give the Administrators group permission to perform any of the IAM actions on any of the AWS account resources. Another example: You can give the Managers group permission to describe the AWS account's Amazon EC2 instances.

For information about how to delegate basic permissions to your users, groups, and roles, see [Permissions Required to Access IAM Resources \(p. 517\)](#). For additional examples of policies that illustrate basic permissions, see [Example Policies for Administering IAM Resources \(p. 521\)](#).

Controlling Access for Principals

You can use policies to control what the person making the request (the principal) is allowed to do. To do this, you must attach an identity-based policy to that person's identity (user, group of users, or role). You can also use a [permissions boundary \(p. 370\)](#) to set the maximum permissions that an entity (user or role) can have.

For example, assume that you want the user Zhang Wei to have full access to CloudWatch, Amazon DynamoDB, Amazon EC2, and Amazon S3. You can create two different policies so that you can later break them up if you need one set of permissions for a different user. Or you can put both the permissions together in a single policy, and then attach that policy to the IAM user that is named Zhang Wei. You could also attach a policy to a group to which Zhang belongs, or a role that Zhang can assume. As a result, when Zhang views the contents of an S3 bucket, his requests are allowed. If he tries to create a new IAM user, his request is denied because he doesn't have permission.

You can use a permissions boundary on Zhang to make sure that he is never given access to the CompanyConfidential S3 bucket. To do this, determine the *maximum* permissions that you want Zhang to have. In this case, you control what he does using his permissions policies. Here, you only care that he doesn't access the confidential bucket. So you use the following policy to define Zhang's boundary to allow all AWS actions for Amazon S3 and a few other services but deny access to the CompanyConfidential S3 bucket. Because the permissions boundary does not allow any IAM actions, it prevents Zhang from deleting his (or anyone's) boundary.

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

```
"Statement": [
    {
        "Sid": "SomeServices",
        "Effect": "Allow",
        "Action": [
            "cloudwatch:*",
            "dynamodb:*",
            "ec2:*",
            "s3:)"
        ],
        "Resource": "*"
    },
    {
        "Sid": "NoConfidentialBucket",
        "Effect": "Deny",
        "Action": "s3:*",
        "Resource": [
            "arn:aws:s3:::CompanyConfidential/*",
            "arn:aws:s3:::CompanyConfidential"
        ]
    }
]
```

When you assign a policy like this as a permissions boundary for a user, remember that it does not grant any permissions. It sets the maximum permissions that an identity-based policy can grant to an IAM entity. For more information about permissions boundaries, see [Permissions Boundaries for IAM Entities \(p. 370\)](#).

For detailed information about the procedures mentioned previously, refer to these resources:

- To learn more about creating an IAM policy that you can attach to a principal, see [Creating IAM Policies \(p. 443\)](#).
- To learn how to attach an IAM policy to a principal, see [Adding and Removing IAM Identity Permissions \(p. 457\)](#).
- To see an example policy for granting full access to EC2, see [Amazon EC2: Allows Full EC2 Access Within a Specific Region, Programmatically and in the Console \(p. 416\)](#).
- To allow read-only access to an S3 bucket, use the first two statements of the following example policy: [Amazon S3: Allows Read and Write Access to Objects in an S3 Bucket, Programmatically and in the Console \(p. 441\)](#).
- To see an example policy for allowing users to set or rotate their credentials, such as their console password, their programmatic access keys, and their MFA devices, see [AWS: Allows MFA-Authenticated IAM Users to Manage Their Own Credentials on the My Security Credentials Page \(p. 398\)](#).

Controlling Access to Identities

You can use IAM policies to control what your users can do to an identity by creating a policy that you attach to all users through a group. To do this, create a policy that limits what can be done to an identity, or who can access it.

For example, you can create a group named **AllUsers**, and then attach that group to all users. When you create the group, you might give all your users access to rotate their credentials as described in the previous section. You can then create a policy that denies access to change the group unless the user name is included in the condition of the policy. But that part of the policy only denies access to anyone except those users listed. You also have to include permissions to allow all the group management actions for everyone in the group. Finally, you attach this policy to the group so that it is applied to all users. As a result, when a user not specified in the policy tries to make changes to the group, the request is denied.

To create this policy with the visual editor

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane on the left, choose **Policies**.

If this is your first time choosing **Policies**, the **Welcome to Managed Policies** page appears. Choose **Get Started**.

3. Choose **Create policy**.
4. On the **Visual editor** tab, choose **Choose a service** to get started. Then choose **IAM**.
5. Choose **Select actions** and then type **group** in the search box. The visual editor shows all the IAM actions that contain the word **group**. Select all of the check boxes.
6. Choose **Resources** to specify resources for your policy. Based on the actions you chose, you should see **group**, **group-path**, and **user** resource types.
 - **group** – Choose **Add ARN**. For **Resource**, select the check box next to **Any**. For **Group Name With Path**, type the group name **AllUsers**. Then choose **Add**.
 - **group-path** – Select the check box next to **Any**.
 - **user** – Select the check box next to **Any**.

One of the actions that you chose, **ListGroups**, does not support using specific resources. You do not have to choose **All resources** for that action. When you save your policy or view the policy on the **JSON** tab, you can see that IAM automatically creates a new permission block granting this action permission on all resources.

7. To add another permission block, choose **Add additional permissions**.
8. Choose **Choose a service** and then choose **IAM**.
9. Choose **Select actions** and then choose **Switch to deny permissions**. When you do that, the entire block is used to deny permissions.
10. Type **group** in the search box. The visual editor shows you all the IAM actions that contain the word **group**. Select the check boxes next to the following actions:
 - **CreateGroup**
 - **DeleteGroup**
 - **RemoveUserFromGroup**
 - **AttachGroupPolicy**
 - **DeleteGroupPolicy**
 - **DetachGroupPolicy**
 - **PutGroupPolicy**
 - **UpdateGroup**
11. Choose **Resources** to specify the resources for your policy. Based on the actions that you chose, you should see the **group** resource type. Choose **Add ARN**. For **Resource**, select the check box next to **Any**. For **Group Name With Path**, type the group name **AllUsers**. Then choose **Add**.
12. Choose **Specify request conditions (optional)** and then choose **Add condition**. Complete the form with the following values:
 - **Key** – Choose **aws:username**
 - **Qualifier** – Choose **Default**
 - **Operator** – Choose **StringNotEquals**
 - **Value** – Type **srodriguez** and then choose **Add another condition value**. Type **mjackson** and then choose **Add another condition value**. Type **adesai** and then choose **Add**.

This condition ensures that access will be denied to the specified group management actions when the user making the call is not included in the list. Because this explicitly denies permission, it overrides the previous block that allowed those users to call the actions. Users on the list are not denied access, and they are granted permission in the first permission block, so they can fully manage the group.

13. When you are finished, choose **Review policy**.

Note

You can switch between the **Visual editor** and **JSON** tabs any time. However, if you make changes or choose **Review policy** in the **Visual editor** tab, IAM might restructure your policy to optimize it for the visual editor. For more information, see [Policy Restructuring \(p. 548\)](#).

14. On the **Review policy** page, for the **Name**, type **LimitAllUserGroupManagement**. For the **Description**, type **Allows all users Read-only access to a specific group, and allows only specific users access to make changes to the group**. Review the policy summary to make sure that you have granted the intended permissions. Then choose **Create policy** to save your new policy.
15. Attach the policy to your group. For more information, see [Adding and Removing IAM Identity Permissions \(p. 457\)](#).

Alternatively, you can create the same policy using this example JSON policy document. To view this JSON policy, see [IAM: Allows Specific IAM Users to Manage a Group Programmatically and in the Console \(p. 427\)](#). For detailed instructions for creating a policy using a JSON document, see the section called “Creating Policies on the JSON Tab” (p. 446).

Controlling Access to Policies

You can control how your users can apply AWS managed policies. To do this, attach this policy to all your users. Ideally, you can do this using a group.

For example, you might create a policy that allows users to attach only the [IAMUserChangePassword](#) and [PowerUserAccess](#) AWS managed policies to a new IAM user, group, or role.

For customer managed policies, you can control who can create, update, and delete these policies. You can control who can attach and detach policies to and from principal entities (groups, users, and roles). You can also control which policies a user can attach or detach, and to and from which entities.

For example, you can give permissions to an account administrator to create, update, and delete policies. Then you give permissions to a team leader or other limited administrator to attach and detach these policies to and from principal entities that the limited administrator manages.

For more information, refer to these resources:

- To learn more about creating an IAM policy that you can attach to a principal, see [Creating IAM Policies \(p. 443\)](#).
- To learn how to attach an IAM policy to a principal, see [Adding and Removing IAM Identity Permissions \(p. 457\)](#).
- To see an example policy for limiting the use of managed policies, see [IAM: Limits Managed Policies That Can Be Applied to an IAM User, Group, or Role \(p. 433\)](#).

Controlling Permissions for Creating, Updating, and Deleting Customer Managed Policies

You can use [IAM policies \(p. 356\)](#) to control who is allowed to create, update, and delete customer managed policies in your AWS account. The following list contains API operations that pertain directly to creating, updating, and deleting policies or policy versions:

- [CreatePolicy](#)
- [CreatePolicyVersion](#)
- [DeletePolicy](#)
- [DeletePolicyVersion](#)
- [SetDefaultPolicyVersion](#)

The API operations in the preceding list correspond to actions that you can allow or deny—that is, permissions that you can grant—using an IAM policy.

Consider the following example policy. It allows a user to create, update (that is, create a new policy version), delete, and set a default version for all customer managed policies in the AWS account. The example policy also allows the user to list policies and get policies. To learn how to create a policy using this example JSON policy document, see [the section called “Creating Policies on the JSON Tab” \(p. 446\)](#).

Example policy that allows creating, updating, deleting, listing, getting, and setting the default version for all policies

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {"Effect": "Allow",  
         "Action": [  
             "iam:CreatePolicy",  
             "iam:CreatePolicyVersion",  
             "iam>DeletePolicy",  
             "iam>DeletePolicyVersion",  
             "iam:GetPolicy",  
             "iam:GetPolicyVersion",  
             "iam>ListPolicies",  
             "iam>ListPolicyVersions",  
             "iam:SetDefaultPolicyVersion"  
         ],  
         "Resource": "*"  
    }  
}
```

You can create policies that limit the use of these API operations to affect only the managed policies that you specify. For example, you might want to allow a user to set the default version and delete policy versions, but only for specific customer managed policies. You do this by specifying the policy ARN in the `Resource` element of the policy that grants these permissions.

The following example shows a policy that allows a user to delete policy versions and set the default version. But these actions are only allowed for the customer managed policies that include the path `/TEAM-A/`. The customer managed policy ARN is specified in the `Resource` element of the policy. (In this example the ARN includes a path and a wildcard and thus matches all customer managed policies that include the path `/TEAM-A/`). To learn how to create a policy using this example JSON policy document, see [the section called “Creating Policies on the JSON Tab” \(p. 446\)](#).

For more information about using paths in the names of customer managed policies, see [Friendly Names and Paths \(p. 575\)](#).

Example policy that allows deleting policy versions and setting the default version for only specific policies

```
{  
    "Version": "2012-10-17",  
    "Statement": {  
        "Effect": "Allow",  
        "Action": [  
            "iam:DeletePolicyVersion",  
            "iam:SetDefaultPolicyVersion"  
        ],  
        "Resource": "arn:aws:iam::ACCOUNT-ID-WITHOUT-HYPHENS:policy/TEAM-A/*"  
    }  
}
```

Controlling Permissions for Attaching and Detaching Managed Policies

You can also use IAM policies to allow users to work with only specific managed policies. In effect, you can control which permissions a user is allowed to grant to other principal entities.

The following list shows API operations that pertain directly to attaching and detaching managed policies to and from principal entities:

- [AttachGroupPolicy](#)
- [AttachRolePolicy](#)
- [AttachUserPolicy](#)
- [DetachGroupPolicy](#)
- [DetachRolePolicy](#)
- [DetachUserPolicy](#)

You can create policies that limit the use of these API operations to affect only the specific managed policies and/or principal entities that you specify. For example, you might want to allow a user to attach managed policies, but only the managed policies that you specify. Or, you might want to allow a user to attach managed policies, but only to the principal entities that you specify.

The following example policy allows a user to attach managed policies to only the groups and roles that include the path /TEAM-A/. The group and role ARNs are specified in the Resource element of the policy. (In this example the ARNs include a path and a wildcard character and thus match all groups and roles that include the path /TEAM-A/). To learn how to create a policy using this example JSON policy document, see [the section called “Creating Policies on the JSON Tab” \(p. 446\)](#).

Example policy that allows attaching managed policies to only specific groups or roles

```
{  
    "Version": "2012-10-17",  
    "Statement": {  
        "Effect": "Allow",  
        "Action": [  
            "iam:AttachGroupPolicy",  
            "iam:AttachRolePolicy"  
        ],  
        "Resource": [  
            "arn:aws:iam::ACCOUNT-ID-WITHOUT-HYPHENS:group/TEAM-A/*",  
            "arn:aws:iam::ACCOUNT-ID-WITHOUT-HYPHENS:role/TEAM-A/*"  
        ]  
    }  
}
```

You can further limit the actions in the preceding example to affect only specific policies. That is, you can control which permissions a user is allowed to attach to other principal entities—by adding a condition to the policy.

In the following example, the condition ensures that the `AttachGroupPolicy` and `AttachRolePolicy` permissions are allowed only when the policy being attached matches one of the specified policies. The condition uses the `iam:PolicyARN` condition key (p. 611) to determine which policy or policies are allowed to be attached. The following example policy expands on the previous example. It allows a user to attach only the managed policies that include the path `/TEAM-A/` to only the groups and roles that include the path `/TEAM-A/`. To learn how to create a policy using this example JSON policy document, see [the section called “Creating Policies on the JSON Tab” \(p. 446\)](#).

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:AttachGroupPolicy",
            "iam:AttachRolePolicy"
        ],
        "Resource": [
            "arn:aws:iam::ACCOUNT-ID-WITHOUT-HYPHENS:group/TEAM-A/*",
            "arn:aws:iam::ACCOUNT-ID-WITHOUT-HYPHENS:role/TEAM-A/*"
        ],
        "Condition": {"ArnLike":
            {"iam:PolicyARN": "arn:aws:iam::ACCOUNT-ID-WITHOUT-HYPHENS:policy/TEAM-A/*"}
        }
    }
}
```

This policy uses the `ArnLike` condition operator because the ARN includes a wildcard character. For a specific ARN, use the `ArnEquals` condition operator. For more information about `ArnLike` and `ArnEquals`, see [Amazon Resource Name \(ARN\) Condition Operators \(p. 619\)](#) in the *Condition Types* section of the *Policy Element Reference*.

For example, you can limit the use of actions to involve only the managed policies that you specify. You do this by specifying the policy ARN in the `Condition` element of the policy that grants these permissions. For example, to specify the ARN of a customer managed policy:

```
"Condition": {"ArnEquals":
    {"iam:PolicyARN": "arn:aws:iam::123456789012:policy/POLICY-NAME"}
}
```

You can also specify the ARN of an AWS managed policy in a policy's `Condition` element. The ARN of an AWS managed policy uses the special alias `aws` in the policy ARN instead of an account ID, as in this example:

```
"Condition": {"ArnEquals":
    {"iam:PolicyARN": "arn:aws:iam::aws:policy/AmazonEC2FullAccess"}
}
```

Controlling Access to Resources

You can control access to resources using an identity-based policy or a resource-based policy. In an identity-based policy, you attach the policy to an identity and specify what resources that identity can access. In a resource-based policy, you attach a policy to the resource that you want to control. In the policy, you specify which principals can access that resource. For more information about both types of policies, see [Identity-Based Policies and Resource-Based Policies \(p. 379\)](#).

For more information, refer to these resources:

- To learn more about creating an IAM policy that you can attach to a principal, see [Creating IAM Policies \(p. 443\)](#).
- To learn how to attach an IAM policy to a principal, see [Adding and Removing IAM Identity Permissions \(p. 457\)](#).
- Amazon S3 supports using resource-based policies on their buckets. For more information, see [Bucket Policy Examples](#).

Resource Creators Do Not Automatically Have Permissions

If you sign in using the AWS account root user credentials, you have permission to perform any action on resources that belong to the account. However, this isn't true for IAM users. An IAM user might be granted access to create a resource, but the user's permissions, even for that resource, are limited to what's been explicitly granted. This means that just because you create a resource, such as an IAM role, you do not automatically have permission to edit or delete that role. Additionally, your permission can be revoked at any time by the account owner or by another user who has been granted access to manage your permissions.

Controlling Access to Principals in a Specific Account

You can directly grant IAM users in your own account access to your resources. If users from another account need access to your resources, you can create an IAM role. A role is an entity that includes permissions but isn't associated with a specific user. Users from other accounts can then assume the role and access resources according to the permissions you've assigned to the role. For more information, see [Providing Access to an IAM User in Another AWS Account That You Own \(p. 179\)](#).

Note

Some services support resource-based policies as described in [Identity-Based Policies and Resource-Based Policies \(p. 379\)](#) (such as Amazon S3, Amazon SNS, and Amazon SQS). For those services, an alternative to using roles is to attach a policy to the resource (bucket, topic, or queue) that you want to share. The resource-based policy can specify the AWS account that has permissions to access the resource.

Controlling Access to and for IAM Users and Roles Using IAM Resource Tags

Use the information in the following section to control who can access your IAM users and roles and what resources your users and roles can access. For more general information and examples for controlling access to other AWS resources, see [Controlling Access to AWS Resources Using Resource Tags \(p. 392\)](#).

Tags can be attached to the IAM *resource*, passed in the *request*, or attached to the *principal* that is making the request. An IAM user or role can be both a resource and principal. For example, you can write a policy that allows a user to list the groups for a user. This operation is allowed only if the user making the request (the principal) has the same project=blue tag as the user they're trying to view. In this example, the user can view the group membership for any user, including themselves, as long as they are working on the same project.

To control access based on tags, you provide tag information in the [condition element \(p. 611\)](#) of a policy. When you create an IAM policy, you can use IAM tags and the associated tag condition key to control access to any of the following:

- **Resource (p. 390)** – Control access to user or role resources based on their tags. To do this, use the `iam:ResourceTag/key-name` condition key to specify which tag key-value pair must be attached to the resource. A similar service-specific key, such as `ec2:ResourceTag`, is used other AWS resources. For more information, see [Controlling Access to AWS Resources \(p. 392\)](#).

- **Request (p. 390)** – Control what tags can be passed in an IAM request. To do this, use the `aws:RequestTag/key-name` condition key to specify what tags can be added, changed, or removed from an IAM user or role. This key is used the same way for IAM resources and other AWS resources. For more information, see [Controlling Access During AWS Requests \(p. 393\)](#).
- **Principal (p. 391)** – Control what the person making the request (the principal) is allowed to do based on the tags that are attached to that person's IAM user or role. To do this, use the `aws:PrincipalTag/key-name` condition key to specify what tags must be attached to the IAM user or role before the request is allowed.
- **Any part of the authorization process (p. 391)** – Use the `aws:TagKeys` condition key to control whether specific tag keys can be used on a resource, in a request, or by a principal. In this case, the key value does not matter. This key behaves similarly for IAM resources and other AWS resources. However, when you tag a user in IAM, this also controls whether the principal can make the request to any service. For more information, see [Controlling Access Based on Tag Keys \(p. 394\)](#).

You can create an IAM policy using the visual editor, using JSON, or by importing an existing managed policy. For details, see [Creating IAM Policies \(p. 443\)](#).

Controlling Access to IAM Resources

You can use tags in your IAM policies to control access to IAM user and role resources. However, because IAM does not support tags for groups, you cannot use tags to control access to groups.

This example shows how you might create a policy that allows deleting users with the `status=terminated` tag. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:DeleteUser",
            "Resource": "*",
            "Condition": {"StringLike": {"iam:ResourceTag/status": "terminated"}}
        }
    ]
}
```

This example shows how you might create a policy that allows editing tags for all users with the `jobFunction = employee` tag. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iam>ListUserTags",
                "iam:TagUser",
                "iam:UntagUser"
            ],
            "Resource": "*",
            "Condition": {"StringLike": {"iam:ResourceTag/jobFunction": "employee"}}
        }
    ]
}
```

Controlling Access During IAM Requests

You can use tags in your IAM policies to control what tags can be passed in the IAM request. You can specify which tag key-value pairs can be added, changed, or removed from an IAM user or role.

This example shows how you might create a policy that allows tagging users only with a `department = HR` or `department = CS` tag. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:TagUser",
            "Resource": "*",
            "Condition": {"StringLike": {"aws:RequestTag/department": [
                "HR",
                "CS"
            ]}}
        }
    ]
}
```

Controlling Access for IAM Principals

IAM tags enable you to control what the principal is allowed to do based on the tags attached to that person's identity.

This example shows how you might create a policy that allows a principal to start or stop an Amazon EC2 instance. This operation is allowed only when the instance's resource tag and the principal's tag have the same value for the tag key `cost-center`. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:startInstances",
                "ec2:stopInstances"
            ],
            "Resource": "*",
            "Condition": {"StringEquals": {"ec2:ResourceTag/costcenter": "${aws:PrincipalTag/cost-center}"}}
        }
    ]
}
```

Controlling Access Based on Tag Keys

You can use tags in your IAM policies to control whether specific tag keys can be used on a resource, in a request, or by a principal.

This example shows how you might create a policy that allows removing only the tag with the `temporary` key from users. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:UntagUser",
            "Resource": "*",
            "Condition": {"ForAllValues:StringEquals": {"aws:TagKeys": ["temporary"]}}
        }
    ]
}
```

Controlling Access to AWS Resources Using Resource Tags

You can use tags to control access to your AWS resources that support tagging. You can also tag IAM users and roles to control what they can access. To learn how to tag IAM users and roles, see [Tagging IAM Users and Roles \(p. 295\)](#). To view a tutorial for creating and testing a policy that allows IAM roles with principal tags to access resources with matching tags, see [Tutorial: Using Tags for Attribute-Based Access Control in AWS \(p. 41\)](#). Use the information in the following section to control access to other AWS services without tagging IAM users or roles.

Before you use tags to control access to your AWS resources, you must understand how AWS grants access. AWS is composed of collections of *resources*. An Amazon EC2 instance is a resource. An Amazon S3 bucket is a resource. You can use the AWS API, the AWS CLI, or the AWS Management Console to perform an operation, such as creating a bucket in Amazon S3. When you do, you send a *request* for that operation. Your request specifies an action, a resource, a *principal entity* (user or role), a *principal account*, and any necessary request information. All of this information provides *context*.

AWS then checks that you (the principal entity) are authenticated (signed in) and authorized (have permission) to perform the specified action on the specified resource. During authorization, AWS checks all the policies that apply to the context of your request. Most policies are stored in AWS as [JSON documents \(p. 361\)](#) and specify the permissions for principal entities. For more information about policy types and uses, see [Policies and Permissions \(p. 356\)](#).

AWS authorizes the request only if each part of your request is allowed by the policies. To view a diagram and learn more about the IAM infrastructure, see [Understanding How IAM Works \(p. 3\)](#). For details about how IAM determines whether a request is allowed, see [Policy Evaluation Logic \(p. 636\)](#).

Tags can complicate this process because tags can be attached to the *resource* or passed in the *request* to services that support tagging. To control access based on tags, you provide tag information in the [condition element \(p. 611\)](#) of a policy. To learn whether an AWS service supports controlling access using tags, see [AWS Services That Work with IAM \(p. 586\)](#) and look for the services that have **Yes** in the **Authorization based on tags** column. Choose the name of the service to view the authorization and access control documentation for that service.

You can then create an IAM policy that allows or denies access to a resource based on that resource's tag. In that policy, you can use tag condition keys to control access to any of the following:

- **Resource (p. 392)** – Control access to AWS service resources based on the tags on those resources. To do this, use the **ResourceTag/*key-name*** condition key to determine whether to allow access to the resource based on the tags that are attached to the resource.
- **Request (p. 393)** – Control what tags can be passed in a request. To do this, use the **aws:RequestTag/*key-name*** condition key to specify what tag key-value pairs can be passed in a request to tag or untag an AWS resource.
- **Any part of the authorization process (p. 394)** – Use the **aws:TagKeys** condition key to control whether specific tag keys can be used on a resource or in a request.

You can create an IAM policy visually, using JSON, or by importing an existing managed policy. For details, see [Creating IAM Policies \(p. 443\)](#).

Controlling Access to AWS Resources

You can use conditions in your IAM policies to control access to AWS resources based on the tags on that resource. You can do this using the global **aws:ResourceTag/*tag-key*** condition key, or a service-specific key such as **iam:RequestTag/*tag-key***. Some services, such as IAM, support only the service-specific version of this key and not the global version.

Note

Do not use the `ResourceTag` condition key in a policy with the `iam:PassRole` action. You cannot use the tag on an IAM role to control access to who can pass that role. For more information about permissions required to pass a role to a service, see [Granting a User Permissions to Pass a Role to an AWS Service \(p. 258\)](#).

This example shows how you might create a policy that allows starting or stopping Amazon EC2 instances. These operations are allowed only if the instance tag `Owner` has the value of that user's user name. This policy also grants the necessary permissions to complete this action on the console.

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

You can attach this policy to the IAM users in your account. If a user named `richard-roe` attempts to start an Amazon EC2 instance, the instance must be tagged `Owner=richard-roe` or `owner=richard-roe`. Otherwise he will be denied access. The tag key `Owner` matches both `Owner` and `owner` because condition key names are not case-sensitive. For more information, see [IAM JSON Policy Elements: Condition \(p. 611\)](#).

Controlling Access During AWS Requests

You can use conditions in your IAM policies to control what tag key-value pairs can be passed in a request that tags an AWS resource.

This example shows how you might create a policy that allows using the Amazon EC2 `CreateTags` action to attach tags to an instance. You can attach tags only if the tag contains the `environment` key and the `preprod` or `production` values. If you want, you can use the `ForAllValues` modifier with the `aws:TagKeys` condition key to indicate that only the key `environment` is allowed in the request. This stops users from including other keys, such as accidentally using `Environment` instead of `environment`.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "ec2:CreateTags",
        "Resource": "arn:aws:ec2:*:*:instance/*",
        "Condition": {
            "StringEquals": {
                "aws:RequestTag/environment": [
                    "preprod",
                    "production"
                ]
            }
        }
    }
}
```

```
        "production"
    ],
},
"ForAllValues:StringEquals": {"aws:TagKeys": "environment"}
}
}
```

Controlling Access Based on Tag Keys

You can use a condition in your IAM policies to control whether specific tag keys can be used on a resource or in a request.

As a best practice, when you use policies to control access using tags, you should use the [aws:TagKeys condition key \(p. 674\)](#). AWS services that support tags might allow you to create multiple tag key names that differ only by case, such as tagging an Amazon EC2 instance with `foo=bar1` and `Foo=bar2`. Key names are not case sensitive in policy conditions. This means that if you specify `"ec2:ResourceTag:TagKey1": "Value1"` in the condition element of your policy, then the condition matches a resource tag key named either `TagKey1` or `tagkey1`, but not both. To prevent duplicate tags with a key that varies only by case, use the `aws:TagKeys` condition to define the tag keys that your users can apply.

This example shows how you might create a policy that allows creating and tagging a Secrets Manager secret, but only with the tag keys `environment` or `cost-center`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager>CreateSecret",
                "secretsmanager:TagResource"
            ],
            "Resource": "*",
            "Condition": {
                "ForAllValues:StringEquals": {
                    "aws:TagKeys": [
                        "environment",
                        "cost-center"
                    ]
                }
            }
        }
    ]
}
```

Example IAM Identity-Based Policies

A [policy \(p. 356\)](#) is an object in AWS that, when associated with an identity or resource, defines their permissions. AWS evaluates these policies when an IAM principal (user or role) makes a request. Permissions in the policies determine whether the request is allowed or denied. Most policies are stored in AWS as JSON documents that are attached to an IAM identity (user, group of users, or role). Identity-based policies include AWS managed policies, customer managed policies, and inline policies. To learn how to create an IAM policy using these example JSON policy documents, see [the section called "Creating Policies on the JSON Tab" \(p. 446\)](#).

By default all requests are denied, so you must provide access to the services, actions, and resources that you intend for the identity to access. If you also want to allow access to complete the specified actions in the IAM console, you need to provide additional permissions.

The following library of policies can help you define permissions for your IAM identities. After you find the policy that you need, choose **view this policy** to view the JSON for the policy. You can use the JSON policy document as a template for your own policies.

Note

If you would like to submit a policy to be included in this reference guide, use the **Feedback** button at the bottom of this page.

Example Policies: AWS

- Allows access during a specific range of dates. ([View this policy \(p. 397\)](#).)
- Allows enabling and disabling AWS Regions. ([View this policy \(p. 397\)](#).)
- Allows MFA-authenticated users to manage their own credentials on the **My Security Credentials** page. ([View this policy \(p. 398\)](#).)
- Allows specific access when using MFA during a specific range of dates. ([View this policy \(p. 401\)](#).)
- Allows users to manage their own credentials on the **My Security Credentials** page. ([View this policy \(p. 402\)](#).)
- Allows users to manage their own MFA device on the **My Security Credentials** page. ([View this policy \(p. 404\)](#).)
- Allows users to manage their own password on the **My Security Credentials** page. ([View this policy \(p. 406\)](#).)
- Allows users to manage their own password, access keys, and SSH public keys on the **My Security Credentials** page. ([View this policy \(p. 406\)](#).)
- Denies access to AWS based on the requested Region. ([View this policy \(p. 408\)](#).)
- Denies access to AWS based on the source IP address. ([View this policy \(p. 409\)](#).)

Example Policies: AWS Data Pipeline

- Denies access to pipelines that a user did not create ([View this policy \(p. 409\)](#).)

Example Policies: Amazon DynamoDB

- Allows access to a specific Amazon DynamoDB table ([View this policy \(p. 410\)](#).)
- Allows access to specific Amazon DynamoDB columns ([View this policy \(p. 411\)](#).)
- Allows row-level access to Amazon DynamoDB based on an Amazon Cognito ID ([View this policy \(p. 411\)](#).)

Example Policies: Amazon EC2

- Allows an Amazon EC2 instance to attach or detach volumes ([View this policy \(p. 412\)](#).)
- Allows attaching or detaching Amazon EBS volumes to Amazon EC2 instances based on tags ([View this policy \(p. 413\)](#).)
- Allows launching Amazon EC2 instances in a specific subnet, programmatically and in the console ([View this policy \(p. 413\)](#).)
- Allows managing Amazon EC2 security groups associated with a specific VPC, programmatically and in the console ([View this policy \(p. 414\)](#).)
- Allows starting or stopping Amazon EC2 instances a user has tagged, programmatically and in the console ([View this policy \(p. 415\)](#).)

- Allows starting or stopping Amazon EC2 instances based on resource and principal tags, programmatically and in the console ([View this policy \(p. 415\)](#).)
- Allows starting or stopping Amazon EC2 instances when the resource and principal tags match ([View this policy \(p. 416\)](#).)
- Allows full Amazon EC2 access within a specific Region, programmatically and in the console. ([View this policy \(p. 416\)](#).)
- Allows starting or stopping a specific Amazon EC2 instance and modifying a specific security group, programmatically and in the console ([View this policy \(p. 417\)](#).)
- Denies access to specific Amazon EC2 operations without MFA ([View this policy \(p. 417\)](#).)
- Limits terminating Amazon EC2 instances to a specific IP address range ([View this policy \(p. 418\)](#).)

Example Policies: AWS Identity and Access Management (IAM)

- Allows access to the policy simulator API ([View this policy \(p. 419\)](#).)
- Allows access to the policy simulator console ([View this policy \(p. 419\)](#).)
- Allows assuming any roles that have a specific tag, programmatically and in the console ([View this policy \(p. 420\)](#).)
- Allows and denies access to multiple services, programmatically and in the console ([View this policy \(p. 420\)](#).)
- Allows adding a specific tag to an IAM user with a different specific tag, programmatically and in the console ([View this policy \(p. 422\)](#).)
- Allows adding a specific tag to any IAM user or role, programmatically and in the console ([View this policy \(p. 423\)](#).)
- Allows creating a new user only with specific tags ([View this policy \(p. 424\)](#).)
- Allows generating and retrieving IAM credential reports ([View this policy \(p. 424\)](#).)
- Allows managing a group's membership, programmatically and in the console ([View this policy \(p. 425\)](#).)
- Allows managing a specific tag ([View this policy \(p. 426\)](#).)
- Allows passing an IAM role to a specific service ([View this policy \(p. 426\)](#).)
- Allows read-only access to the IAM console without reporting ([View this policy \(p. 427\)](#).)
- Allows read-only access to the IAM console ([View this policy \(p. 427\)](#).)
- Allows specific users to manage a group, programmatically and in the console ([View this policy \(p. 427\)](#).)
- Allows setting the account password requirements, programmatically and in the console ([View this policy \(p. 429\)](#).)
- Allows using the policy simulator API for users with a specific path ([View this policy \(p. 429\)](#).)
- Allows using the policy simulator console for users with a specific path ([View this policy \(p. 429\)](#).)
- Allows IAM users to self-manage an MFA device. ([View this policy \(p. 430\)](#).)
- Allows IAM users to rotate their own credentials, programmatically and in the console. ([View this policy \(p. 431\)](#).)
- Allows viewing service last accessed data for an AWS Organizations policy in the IAM console. ([View this policy \(p. 432\)](#).)
- Limits managed policies that can be applied to an IAM user, group, or role ([View this policy \(p. 433\)](#).)

Example Policies: AWS Lambda

- Allows an AWS Lambda function to access an Amazon DynamoDB table ([View this policy \(p. 433\)](#).)

Example Policies: Amazon RDS

- Allows full Amazon RDS database access within a specific Region. ([View this policy \(p. 434\)](#).)
- Allows restoring Amazon RDS databases, programmatically and in the console ([View this policy \(p. 434\)](#).)
- Allows tag owners full access to Amazon RDS resources that they have tagged ([View this policy \(p. 435\)](#).)

Example Policies: Amazon S3

- Allows an Amazon Cognito user to access objects in their own Amazon S3 bucket ([View this policy \(p. 436\)](#).)
- Allows federated users to access their own home directory in Amazon S3, programmatically and in the console ([View this policy \(p. 437\)](#).)
- Allows full S3 access, but explicitly denies access to the Production bucket if the administrator has not signed in using MFA within the last thirty minutes ([View this policy \(p. 438\)](#).)
- Allows IAM users to access their own home directory in Amazon S3, programmatically and in the console ([View this policy \(p. 439\)](#).)
- Allows a user to manage a single Amazon S3 bucket and denies every other AWS action and resource ([View this policy \(p. 440\)](#).)
- Allows Read and Write access to a specific Amazon S3 bucket ([View this policy \(p. 441\)](#).)
- Allows Read and Write access to a specific Amazon S3 bucket, programmatically and in the console ([View this policy \(p. 441\)](#).)

AWS: Allows Access Within Specific Dates

This example shows how you might create a policy that allows access to the ACTION-NAME action in the service named SERVICE-NAME. Access is restricted to actions that occur between July 1, 2017 and December 31, 2017 (UTC), inclusive. This policy grants the permissions necessary to complete this action from the AWS API or AWS CLI only. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

To learn about using multiple conditions within the Condition block of an IAM policy, see [Multiple Values in a Condition \(p. 613\)](#).

```
{  
    "Version": "2012-10-17",  
    "Statement": {  
        "Effect": "Allow",  
        "Action": "service-prefix:action-name",  
        "Resource": "*",  
        "Condition": {  
            "DateGreaterThan": {"aws:CurrentTime": "2017-07-01T00:00:00Z"},  
            "DateLessThan": {"aws:CurrentTime": "2017-12-31T23:59:59Z"}  
        }  
    }  
}
```

AWS: Allows Enabling and Disabling AWS Regions

This example shows how you might create a policy that allows an administrator to enable and disable the Asia Pacific (Hong Kong) Region (ap-east-1). This policy also grants the necessary permissions to complete this action on the console. This setting appears in the **Account settings** page in the AWS

Management Console. This page includes sensitive account-level information that should be viewed and managed only by account administrators. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

Important

You cannot enable or disable regions that are enabled by default. You can only include regions that are *disabled* by default. For more information, see [Managing AWS Regions in the AWS General Reference](#).

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "EnableDisableHongKong",  
            "Effect": "Allow",  
            "Action": [  
                "account:EnableRegion",  
                "account:DisableRegion"  
            ],  
            "Resource": "*",  
            "Condition": {  
                "StringEquals": {"account:TargetRegion": "ap-east-1"}  
            }  
        },  
        {  
            "Sid": "ViewConsole",  
            "Effect": "Allow",  
            "Action": [  
                "aws-portal:ViewAccount",  
                "account>ListRegions"  
            ],  
            "Resource": "*"  
        }  
    ]  
}
```

AWS: Allows MFA-Authenticated IAM Users to Manage Their Own Credentials on the My Security Credentials Page

This example shows how you might create a policy that allows IAM users that are authenticated using [multi-factor authentication \(MFA\)](#) (p. 119) to manage their own credentials on the [My Security Credentials](#) page. This AWS Management Console page displays account information such as the account ID and canonical user ID. Users can also view and edit their own passwords, access keys, MFA devices, X.509 certificates, and SSH keys and Git credentials. This example policy includes the permissions required to view and edit all of the information on the page. It also requires the user to set up and authenticate using MFA before performing any other operations in AWS. To allow users to manage their own credentials without using MFA, see [AWS: Allows IAM Users to Manage Their Own Credentials on the My Security Credentials Page \(p. 402\)](#).

To learn how users can access the **My Security Credentials** page, see [How IAM Users Change Their Own Password \(Console\) \(p. 109\)](#).

Note

This example policy does not allow users to reset a password while signing in. New users and users with an expired password might try to do so. You can allow this by adding `iam:ChangePassword` and `iam:GetAccountPasswordPolicy` to the statement `DenyAllExceptListedIfNoMFA`. However, IAM does not recommend this. Allowing users to change their password without MFA can be a security risk.

What does this policy do?

- The `AllowViewAccountInfo` statement allows the user to view account-level information. These permissions must be in their own statement because they do not support or do not need to specify a resource ARN. Instead the permissions specify "Resource" : "*". This statement includes the following actions that allow the user to view specific information:
 - `GetAccountSummary` – View the account ID and the account [canonical user ID](#).
 - `GetAccountPasswordPolicy` – View the account password requirements while changing their own IAM user password.
 - `ListVirtualMFADevices` – View details about a virtual MFA device that is enabled for the user.
- The `AllowManageOwnPasswords` statement allows the user to change their own password. This statement also includes the `GetUser` action, which is required to view most of the information on the **My Security Credentials** page.
- The `AllowManageOwnAccessKeys` statement allows the user to create, update, and delete their own access keys.
- The `AllowManageOwnSigningCertificates` statement allows the user to upload, update, and delete their own signing certificates.
- The `AllowManageOwnSSHPublicKeys` statement allows the user to upload, update, and delete their own SSH public keys for CodeCommit.
- The `AllowManageOwnGitCredentials` statement allows the user to create, update, and delete their own Git credentials for CodeCommit.
- The `AllowManageOwnVirtualMFADevice` statement allows the user to create and delete their own virtual MFA device. The resource ARN in this statement allows access to only an MFA device that has the same name as the currently signed-in user. Users can't create or delete any virtual MFA device other than their own.
- The `AllowManageOwnUserMFA` statement allows the user to view or manage the virtual, U2F, or hardware MFA device for their own user. The resource ARN in this statement allows access to only the user's own IAM user. Users can't view or manage the MFA device for other users.
- The `DenyAllExceptListedIfNoMFA` statement denies access to every action in all AWS services, except a few listed actions, but *only if* the user is not signed in with MFA. The statement uses a combination of "Deny" and "NotAction" to explicitly deny access to every action that is not listed. The items listed are not denied or allowed by this statement. However, the actions are allowed by other statements in the policy. For more information about the logic for this statement, see [NotAction with Deny \(p. 607\)](#). If the user is signed in with MFA, then the Condition test fails and this statement does not deny any actions. In this case, other policies or statements for the user determine the user's permissions.

This statement ensures that when the user is not signed in with MFA that they can perform only the listed actions. In addition, they can perform the listed actions only if another statement or policy allows access to those actions. This does not allow a user to create a password at sign-in, because `iam:ChangePassword` action should not be allowed without MFA authorization.

The `...IfExists` version of the `Bool` operator ensures that if the `aws:MultiFactorAuthPresent` key is missing, the condition returns true. This means that a user accessing an API with long-term credentials, such as an access key, is denied access to the non-IAM API operations.

This policy does not allow users to view the **Users** page in the IAM console or use that page to access their own user information. To allow this, add the `iam>ListUsers` action to the `AllowViewAccountInfo` statement and the `DenyAllExceptListedIfNoMFA` statement. It also does not allow users to change their password on their own user page. To allow this, add the `iam>CreateLoginProfile`, `iam>DeleteLoginProfile`, `iam:GetLoginProfile`, and `iam:UpdateLoginProfile` actions to the `AllowManageOwnPasswords` statement. To also allow a user to change their password from their own user page without signing in using MFA, add the `iam>CreateLoginProfile` action to the `DenyAllExceptListedIfNoMFA` statement.

```
{
```

```
"Version": "2012-10-17",
"Statement": [
    {
        "Sid": "AllowViewAccountInfo",
        "Effect": "Allow",
        "Action": [
            "iam:GetAccountPasswordPolicy",
            "iam:GetAccountSummary",
            "iam>ListVirtualMFADevices"
        ],
        "Resource": "*"
    },
    {
        "Sid": "AllowManageOwnPasswords",
        "Effect": "Allow",
        "Action": [
            "iam:ChangePassword",
            "iam:GetUser"
        ],
        "Resource": "arn:aws:iam::*:user/${aws:username}"
    },
    {
        "Sid": "AllowManageOwnAccessKeys",
        "Effect": "Allow",
        "Action": [
            "iam>CreateAccessKey",
            "iam>DeleteAccessKey",
            "iam>ListAccessKeys",
            "iam:UpdateAccessKey"
        ],
        "Resource": "arn:aws:iam::*:user/${aws:username}"
    },
    {
        "Sid": "AllowManageOwnSigningCertificates",
        "Effect": "Allow",
        "Action": [
            "iam>DeleteSigningCertificate",
            "iam>ListSigningCertificates",
            "iam:UpdateSigningCertificate",
            "iam:UploadSigningCertificate"
        ],
        "Resource": "arn:aws:iam::*:user/${aws:username}"
    },
    {
        "Sid": "AllowManageOwnSSHPublicKeys",
        "Effect": "Allow",
        "Action": [
            "iam>DeleteSSHPublicKey",
            "iam:GetSSHPublicKey",
            "iam>ListSSHPublicKeys",
            "iam:UpdateSSHPublicKey",
            "iam:UploadSSHPublicKey"
        ],
        "Resource": "arn:aws:iam::*:user/${aws:username}"
    },
    {
        "Sid": "AllowManageOwnGitCredentials",
        "Effect": "Allow",
        "Action": [
            "iam>CreateServiceSpecificCredential",
            "iam>DeleteServiceSpecificCredential",
            "iam>ListServiceSpecificCredentials",
            "iam:ResetServiceSpecificCredential",
            "iam:UpdateServiceSpecificCredential"
        ],
        "Resource": "arn:aws:iam::*:user/${aws:username}"
    }
]
```

```

},
{
    "Sid": "AllowManageOwnVirtualMFADevice",
    "Effect": "Allow",
    "Action": [
        "iam>CreateVirtualMFADevice",
        "iam>DeleteVirtualMFADevice"
    ],
    "Resource": "arn:aws:iam::*:mfa/${aws:username}"
},
{
    "Sid": "AllowManageOwnUserMFA",
    "Effect": "Allow",
    "Action": [
        "iam>DeactivateMFADevice",
        "iam>EnableMFADevice",
        "iam>ListMFADevices",
        "iam>ResyncMFADevice"
    ],
    "Resource": "arn:aws:iam::*:user/${aws:username}"
},
{
    "Sid": "DenyAllExceptListedIfNoMFA",
    "Effect": "Deny",
    "NotAction": [
        "iam>CreateVirtualMFADevice",
        "iam>EnableMFADevice",
        "iam GetUser",
        "iam>ListMFADevices",
        "iam>ListVirtualMFADevices",
        "iam>ResyncMFADevice",
        "sts:GetSessionToken"
    ],
    "Resource": "*",
    "Condition": {
        "BoolIfExists": {
            "aws:MultiFactorAuthPresent": "false"
        }
    }
}
]
}

```

AWS: Allows Specific Access Using MFA Within Specific Dates

This example shows how you might create a policy that uses multiple conditions, which are evaluated using a logical AND. It allows full access to the service named `SERVICE-NAME-1`, and access to the `ACTION-NAME-A` and `ACTION-NAME-B` actions in the service named `SERVICE-NAME-2`. These actions are allowed only when the user is authenticated using [multifactor authentication \(MFA\)](#). Access is restricted to actions that occur between July 1, 2017 and December 31, 2017 (UTC), inclusive. This policy grants the permissions necessary to complete this action from the AWS API or AWS CLI only. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

To learn about using multiple conditions within the Condition block of an IAM policy, see [Multiple Values in a Condition \(p. 613\)](#).

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "service-prefix-1:*",
            "service-prefix-2:action-name-a",

```

```
        "service-prefix-2:action-name-b"
    ],
    "Resource": "*",
    "Condition": {
        "Bool": {"aws:MultiFactorAuthPresent": true},
        "DateGreaterThanOrEqualTo": {"aws:CurrentTime": "2017-07-01T00:00:00Z"},
        "DateLessThanOrEqualTo": {"aws:CurrentTime": "2017-12-31T23:59:59Z"}
    }
}
```

AWS: Allows IAM Users to Manage Their Own Credentials on the My Security Credentials Page

This example shows how you might create a policy that allows IAM users to manage all of their own credentials on the **My Security Credentials** page. This AWS Management Console page displays account information such as the account ID and canonical user ID. Users can also view and edit their own passwords, access keys, X.509 certificates, SSH keys, and Git credentials. This example policy includes the permissions required to view and edit all information on the page *except* the user's MFA device. To allow users to manage their own credentials with MFA, see [AWS: Allows MFA-Authenticated IAM Users to Manage Their Own Credentials on the My Security Credentials Page \(p. 398\)](#).

To learn how users can access the **My Security Credentials** page, see [How IAM Users Change Their Own Password \(Console\) \(p. 109\)](#).

What does this policy do?

- The `AllowViewAccountInfo` statement allows the user to view account-level information. These permissions must be in their own statement because they do not support or do not need to specify a resource ARN. Instead the permissions specify `"Resource" : "*"`. This statement includes the following actions that allow the user to view specific information:
 - `GetAccountPasswordPolicy` – View the account password requirements while changing their own IAM user password.
 - `GetAccountSummary` – View the account ID and the account [canonical user ID](#).
- The `AllowManageOwnPasswords` statement allows the user to change their own password. This statement also includes the `GetUser` action, which is required to view most of the information on the **My Security Credentials** page.
- The `AllowManageOwnAccessKeys` statement allows the user to create, update, and delete their own access keys.
- The `AllowManageOwnSigningCertificates` statement allows the user to upload, update, and delete their own signing certificates.
- The `AllowManageOwnSSHPublicKeys` statement allows the user to upload, update, and delete their own SSH public keys for CodeCommit.
- The `AllowManageOwnGitCredentials` statement enables the user to create, update, and delete their own Git credentials for CodeCommit.

This policy does not allow users to view or manage their own MFA devices. They also cannot view the **Users** page in the IAM console or use that page to access their own user information. To allow this, add the `iam>ListUsers` action to the `AllowViewAccountInfo` statement. It also does not allow users to change their password on their own user page. To allow this, add the `iam>CreateLoginProfile`, `iam>DeleteLoginProfile`, `iam>GetLoginProfile`, and `iam>UpdateLoginProfile` actions to the `AllowManageOwnPasswords` statement.

```
{
```

```
"Version": "2012-10-17",
"Statement": [
    {
        "Sid": "AllowViewAccountInfo",
        "Effect": "Allow",
        "Action": [
            "iam:GetAccountPasswordPolicy",
            "iam:GetAccountSummary"
        ],
        "Resource": "*"
    },
    {
        "Sid": "AllowManageOwnPasswords",
        "Effect": "Allow",
        "Action": [
            "iam:ChangePassword",
            "iam:GetUser"
        ],
        "Resource": "arn:aws:iam::*:user/${aws:username}"
    },
    {
        "Sid": "AllowManageOwnAccessKeys",
        "Effect": "Allow",
        "Action": [
            "iam>CreateAccessKey",
            "iam>DeleteAccessKey",
            "iam>ListAccessKeys",
            "iam:UpdateAccessKey"
        ],
        "Resource": "arn:aws:iam::*:user/${aws:username}"
    },
    {
        "Sid": "AllowManageOwnSigningCertificates",
        "Effect": "Allow",
        "Action": [
            "iam>DeleteSigningCertificate",
            "iam>ListSigningCertificates",
            "iam:UpdateSigningCertificate",
            "iam:UploadSigningCertificate"
        ],
        "Resource": "arn:aws:iam::*:user/${aws:username}"
    },
    {
        "Sid": "AllowManageOwnSSHPublicKeys",
        "Effect": "Allow",
        "Action": [
            "iam>DeleteSSHPublicKey",
            "iam:GetSSHPublicKey",
            "iam>ListSSHPublicKeys",
            "iam:UpdateSSHPublicKey",
            "iam:UploadSSHPublicKey"
        ],
        "Resource": "arn:aws:iam::*:user/${aws:username}"
    },
    {
        "Sid": "AllowManageOwnGitCredentials",
        "Effect": "Allow",
        "Action": [
            "iam>CreateServiceSpecificCredential",
            "iam>DeleteServiceSpecificCredential",
            "iam>ListServiceSpecificCredentials",
            "iam:ResetServiceSpecificCredential",
            "iam:UpdateServiceSpecificCredential"
        ],
        "Resource": "arn:aws:iam::*:user/${aws:username}"
    }
]
```

```
    ]  
}
```

AWS: Allows MFA-Authenticated IAM Users to Manage Their Own MFA Device on the My Security Credentials Page

This example shows how you might create a policy that allows IAM users that are authenticated through [multi-factor authentication \(MFA\) \(p. 119\)](#) to manage their own MFA device on the [My Security Credentials](#) page. This AWS Management Console page displays account and user information, but the user can only view and edit their own MFA device. To allow users to manage all of their own credentials with MFA, see [AWS: Allows MFA-Authenticated IAM Users to Manage Their Own Credentials on the My Security Credentials Page \(p. 398\)](#).

Note

If an IAM user with this policy is not MFA-authenticated, this policy denies access to all AWS actions except those necessary to authenticate using MFA. To use the AWS CLI and AWS API, IAM users must first retrieve their MFA token using the AWS STS [GetSessionToken](#) operation and then use that token to authenticate the desired operation. Other policies, such as resource-based policies or other identity-based policies can allow actions in other services. This policy will deny that access if the IAM user is not MFA-authenticated.

To learn how users can access the [My Security Credentials](#) page, see [How IAM Users Change Their Own Password \(Console\) \(p. 109\)](#).

What does this policy do?

- The `AllowViewAccountInfo` statement allows the user to view details about a virtual MFA device that is enabled for the user. This permission must be in its own statement because it does not support specifying a resource ARN. Instead you must specify "Resource" : "*".
- The `AllowManageOwnVirtualMFADevice` statement allows the user to create and delete their own virtual MFA device. The resource ARN in this statement allows access to only an MFA device that has the same name as the currently signed-in user. Users can't create or delete any virtual MFA device other than their own.
- The `AllowManageOwnUserMFA` statement allows the user to view or manage their own virtual, U2F, or hardware MFA device. The resource ARN in this statement allows access to only the user's own IAM user. Users can't view or manage the MFA device for other users.
- The `DenyAllExceptListedIfNoMFA` statement denies access to every action in all AWS services, except a few listed actions, but **only if** the user is not signed in with MFA. The statement uses a combination of "Deny" and "NotAction" to explicitly deny access to every action that is not listed. The items listed are not denied or allowed by this statement. However, the actions are allowed by other statements in the policy. For more information about the logic for this statement, see [NotAction with Deny \(p. 607\)](#). If the user is signed in with MFA, then the Condition test fails and this statement does not deny any actions. In this case, other policies or statements for the user determine the user's permissions.

This statement ensures that when the user is not signed in with MFA, they can perform only the listed actions. In addition, they can perform the listed actions only if another statement or policy allows access to those actions.

The `...IfExists` version of the `Bool` operator ensures that if the `aws:MultiFactorAuthPresent` key is missing, the condition returns true. This means that a user accessing an API operation with long-term credentials, such as an access key, is denied access to the non-IAM API operations.

This policy does not allow users to view the [Users](#) page in the IAM console or use that page to access their own user information. To allow this, add the `iam>ListUsers` action to the `AllowViewAccountInfo` statement and the `DenyAllExceptListedIfNoMFA` statement.

Warning

Do not allow add permission to delete an MFA device without MFA authentication. Users with this policy might attempt to assign themselves an MFA device and receive an error that they are not authorized to perform `iam>DeleteVirtualMFADevice`. If this happens, **do not** add that permission to the `DenyAllExceptListedIfNoMFA` statement. Users that are not authenticated using MFA should never be allowed to delete their MFA device. Users might see this error if they previously began assigning a virtual MFA device to their user and cancelled the process. To resolve this issue, you or another administrator must delete the user's existing MFA device using the AWS CLI or AWS API. For more information, see [I Am Not Authorized to Perform: `iam>DeleteVirtualMFADevice` \(p. 547\)](#).

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowViewAccountInfo",
            "Effect": "Allow",
            "Action": "iam>ListVirtualMFADevices",
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnVirtualMFADevice",
            "Effect": "Allow",
            "Action": [
                "iam>CreateVirtualMFADevice",
                "iam>DeleteVirtualMFADevice"
            ],
            "Resource": "arn:aws:iam::*:mfa/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnUserMFA",
            "Effect": "Allow",
            "Action": [
                "iam>DeactivateMFADevice",
                "iam>EnableMFADevice",
                "iam GetUser",
                "iam>ListMFADevices",
                "iam>ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "DenyAllExceptListedIfNoMFA",
            "Effect": "Deny",
            "NotAction": [
                "iam>CreateVirtualMFADevice",
                "iam>EnableMFADevice",
                "iam GetUser",
                "iam>ListMFADevices",
                "iam>ListVirtualMFADevices",
                "iam>ResyncMFADevice",
                "sts:GetSessionToken"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {"aws:MultiFactorAuthPresent": "false"}
            }
        }
    ]
}
```

AWS: Allows IAM Users to Change Their Own Console Password on the My Security Credentials Page

This example shows how you might create a policy that allows IAM users to change their own AWS Management Console password on the **My Security Credentials** page. This AWS Management Console page displays account and user information, but the user can only access their own password. To allow users to manage all of their own credentials with MFA, see [AWS: Allows MFA-Authenticated IAM Users to Manage Their Own Credentials on the My Security Credentials Page \(p. 398\)](#). To allow users to manage their own credentials without using MFA, see [AWS: Allows IAM Users to Manage Their Own Credentials on the My Security Credentials Page \(p. 402\)](#).

To learn how users can access the **My Security Credentials** page, see [How IAM Users Change Their Own Password \(Console\) \(p. 109\)](#).

What does this policy do?

- The `ViewAccountPasswordRequirements` statement allows the user to view the account password requirements while changing their own IAM user password.
- The `ChangeOwnPassword` statement allows the user to change their own password. This statement also includes the `GetUser` action, which is required to view most of the information on the **My Security Credentials** page.

This policy does not allow users to view the **Users** page in the IAM console or use that page to access their own user information. To allow this, add the `iam>ListUsers` action to the `ViewAccountPasswordRequirements` statement.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "ViewAccountPasswordRequirements",  
            "Effect": "Allow",  
            "Action": "iam:GetAccountPasswordPolicy",  
            "Resource": "*"  
        },  
        {  
            "Sid": "ChangeOwnPassword",  
            "Effect": "Allow",  
            "Action": [  
                "iam:GetUser",  
                "iam:ChangePassword"  
            ],  
            "Resource": "arn:aws:iam::*:user/${aws:username}"  
        }  
    ]  
}
```

AWS: Allows IAM Users to Manage Their Own Password, Access Keys, and SSH Public Keys on the My Security Credentials Page

This example shows how you might create a policy that allows IAM users to manage their own password, access keys, and X.509 certificates on the **My Security Credentials** page. This AWS Management Console page displays account information such as the account ID and canonical user ID. Users can also view and edit their own passwords, access keys, MFA devices, X.509 certificates, SSH keys, and Git credentials. This example policy includes the permissions that are required to view and edit only their password, access keys, and X.509 certificate. To allow users to manage all of their own credentials with MFA, see [AWS:](#)

Allows MFA-Authenticated IAM Users to Manage Their Own Credentials on the My Security Credentials Page (p. 398). To allow users to manage their own credentials without using MFA, see AWS: Allows IAM Users to Manage Their Own Credentials on the My Security Credentials Page (p. 402).

To learn how users can access the **My Security Credentials** page, see How IAM Users Change Their Own Password (Console) (p. 109).

What does this policy do?

- The AllowViewAccountInfo statement allows the user to view account-level information. These permissions must be in their own statement because they do not support or do not need to specify a resource ARN. Instead the permissions specify "Resource" : "*". This statement includes the following actions that allow the user to view specific information:
 - GetAccountPasswordPolicy – View the account password requirements while changing their own IAM user password.
 - GetAccountSummary – View the account ID and the account canonical user ID.
- The AllowManageOwnPasswords statement allows the user to change their own password. This statement also includes the GetUser action, which is required to view most of the information on the **My Security Credentials** page.
- The AllowManageOwnAccessKeys statement allows the user to create, update, and delete their own access keys.
- The AllowManageOwnSSHPublicKeys statement allows the user to upload, update, and delete their own SSH public keys for CodeCommit.

This policy does not allow users to view or manage their own MFA devices. They also cannot view the **Users** page in the IAM console or use that page to access their own user information. To allow this, add the iam>ListUsers action to the AllowViewAccountInfo statement. It also does not allow users to change their password on their own user page. To allow this, add the iam>CreateLoginProfile, iam>DeleteLoginProfile, iam>GetLoginProfile, and iam>UpdateLoginProfile actions to the AllowManageOwnPasswords statement.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "AllowViewAccountInfo",  
            "Effect": "Allow",  
            "Action": [  
                "iam:GetAccountPasswordPolicy",  
                "iam:GetAccountSummary"  
            ],  
            "Resource": "*"  
        },  
        {  
            "Sid": "AllowManageOwnPasswords",  
            "Effect": "Allow",  
            "Action": [  
                "iam:ChangePassword",  
                "iam:GetUser"  
            ],  
            "Resource": "arn:aws:iam::*:user/${aws:username}"  
        },  
        {  
            "Sid": "AllowManageOwnAccessKeys",  
            "Effect": "Allow",  
            "Action": [  
                "iam>CreateAccessKey",  
                "iam>DeleteAccessKey",  
                "iam>ListAccessKeys",  
                "iam:UpdateAccessKey"  
            ]  
        }  
    ]  
}
```

```

        "iam:UpdateAccessKey"
    ],
    "Resource": "arn:aws:iam::*:user/${aws:username}"
},
{
    "Sid": "AllowManageOwnSSHPublicKeys",
    "Effect": "Allow",
    "Action": [
        "iam:DeleteSSHPublicKey",
        "iam:GetSSHPublicKey",
        "iam>ListSSHPublicKeys",
        "iam:UpdateSSHPublicKey",
        "iam:UploadSSHPublicKey"
    ],
    "Resource": "arn:aws:iam::*:user/${aws:username}"
}
]
}

```

AWS: Denies Access to AWS Based on the Requested Region

This example shows how you might create a policy that denies access to any operations outside of the `eu-central-1` and `eu-west-1` Regions, except for actions in the listed services. This policy also grants the necessary permissions to complete this action on the console. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

This policy uses the `NotAction` element with the `Deny` effect, which explicitly denies access to all of the actions *not* listed in the statement. Actions in the CloudFront, IAM, Route 53, and AWS Support services should not be denied because these are popular AWS global services with a single endpoint that is physically located in the `us-east-1` Region. Because all requests to these services are made to the `us-east-1` Region, the requests would be denied without the `NotAction` element. Edit this element to include actions for other AWS global services that you use, such as budgets, globalaccelerator, importexport, organizations, or waf. To learn about all of the services that have a single global endpoint, see [AWS Regions and Endpoints](#) in the *AWS General Reference*. For more information about using the `NotAction` element with the `Deny` effect, see [IAM JSON Policy Elements: NotAction \(p. 607\)](#).

Important

This policy does not allow any actions. Use this policy in combination with other policies that allow specific actions.

```

{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DenyAllOutsideEU",
            "Effect": "Deny",
            "NotAction": [
                "cloudfront:*",
                "iam:*",
                "route53:*",
                "support:*
            ],
            "Resource": "*",
            "Condition": {
                "StringNotEquals": {
                    "aws:RequestedRegion": [
                        "eu-central-1",
                        "eu-west-1"
                    ]
                }
            }
        }
    ]
}

```

```
    ]  
}
```

AWS: Denies Access to AWS Based on the Source IP

This example shows how you might create a policy that denies access to all AWS actions in the account when the request comes from outside the specified IP range. The policy is useful when the IP addresses for your company are within the specified ranges. This policy also grants the necessary permissions to complete this action on the console. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

Do not use the `aws:SourceIp` condition key in a service role. It denies access to an AWS service, such as AWS CloudFormation, even when it makes calls on your behalf. For more information about using the `aws:SourceIp` condition key, see [AWS Global Condition Context Keys \(p. 665\)](#).

Important

This policy does not allow any actions. Use this policy in combination with other policies that allow specific actions.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Deny",  
            "Action": "*",  
            "Resource": "*",  
            "Condition": {  
                "NotIpAddress": {  
                    "aws:SourceIp": [  
                        "192.0.2.0/24",  
                        "203.0.113.0/24"  
                    ]  
                }  
            }  
        }  
    ]  
}
```

AWS Data Pipeline: Denies Access to DataPipeline Pipelines That a User Did Not Create

This example shows how you might create a policy that denies access to pipelines that a user did not create. If the value of the `PipelineCreator` field matches the IAM user name, then the specified actions are not denied. This policy grants the permissions necessary to complete this action from the AWS API or AWS CLI only.

Important

This policy does not allow any actions. Use this policy in combination with other policies that allow specific actions.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "ExplicitDenyIfNotTheOwner",  
            "Effect": "Deny",  
            "Action": [  
                "datapipeline:ActivatePipeline",  
                "datapipeline:AddTags",  
                "datapipeline:DeactivatePipeline",  
                "datapipeline:DeletePipeline",  
                "datapipeline:PutStep",  
                "datapipeline:PutStage",  
                "datapipeline:UpdatePipeline",  
                "datapipeline:UpdateStep",  
                "datapipeline:UpdateStage"  
            ]  
        }  
    ]  
}
```

```

        "datapipeline:DescribeObjects",
        "datapipeline:EvaluateExpression",
        "datapipeline:GetPipelineDefinition",
        "datapipeline:PollForTask",
        "datapipeline:PutPipelineDefinition",
        "datapipeline:QueryObjects",
        "datapipeline:RemoveTags",
        "datapipeline:ReportTaskProgress",
        "datapipeline:ReportTaskRunnerHeartbeat",
        "datapipeline:SetStatus",
        "datapipeline:SetTaskStatus",
        "datapipeline:ValidatePipelineDefinition"
    ],
    "Resource": ["*"],
    "Condition": {
        "StringNotEquals": {"datapipeline:PipelineCreator": "${aws:userid}"}
    }
}
]
}

```

Amazon DynamoDB: Allows Access to a Specific Table

This example shows how you might create a policy that allows full access to the `MyTable` DynamoDB table. This policy grants the permissions necessary to complete this action from the AWS API or AWS CLI only. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

Important

This policy allows all actions that can be performed on a DynamoDB table. To review these actions, see [DynamoDB API Permissions: Actions, Resources, and Conditions Reference](#) in the *Amazon DynamoDB Developer Guide*. You could provide the same permissions by listing each individual action. However, if you use the wildcard (*) in the `Action` element, such as `"dynamodb>List*"`, then you don't have to update your policy if DynamoDB adds a new List action.

This policy allows actions only on DynamoDB tables that exist with the specified name. To allow your users Read access to everything in DynamoDB, you can also attach the [AmazonDynamoDBReadOnlyAccess](#) AWS managed policy.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ListAndDescribe",
            "Effect": "Allow",
            "Action": [
                "dynamodb>List*",
                "dynamodb>DescribeReservedCapacity*",
                "dynamodb>DescribeLimits",
                "dynamodb>DescribeTimeToLive"
            ],
            "Resource": "*"
        },
        {
            "Sid": "SpecificTable",
            "Effect": "Allow",
            "Action": [
                "dynamodb>BatchGet*",
                "dynamodb>DescribeStream",
                "dynamodb>DescribeTable",
                "dynamodb>Get*",
                "dynamodb>Update*"
            ],
            "Resource": [
                "arn:aws:dynamodb:us-east-1:123456789012:table/MyTable"
            ]
        }
    ]
}
```

```
        "dynamodb:Query",
        "dynamodb:Scan",
        "dynamodb:BatchWrite*",
        "dynamodb CreateTable",
        "dynamodb>Delete*",
        "dynamodb:Update*",
        "dynamodb:PutItem"
    ],
    "Resource": "arn:aws:dynamodb:*::table/MyTable"
}
]
```

Amazon DynamoDB: Allows Access to Specific Columns

This example shows how you might create a policy that allows access to the specific DynamoDB columns. This policy grants the permissions necessary to complete this action from the AWS API or AWS CLI only. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

The dynamodb:Select requirement prevents the API action from returning any attributes that aren't allowed, such as from an index projection. To learn more about DynamoDB condition keys, see [Specifying Conditions: Using Condition Keys](#) in the *Amazon DynamoDB Developer Guide*. To learn about using multiple conditions or multiple condition keys within the Condition block of an IAM policy, see [Multiple Values in a Condition \(p. 613\)](#).

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "dynamodb:GetItem",
                "dynamodb:BatchGetItem",
                "dynamodb:Query",
                "dynamodb:PutItem",
                "dynamodb:UpdateItem",
                "dynamodb:DeleteItem",
                "dynamodb:BatchWriteItem"
            ],
            "Resource": ["arn:aws:dynamodb:*::table/table-name"],
            "Condition": {
                "ForAllValues:StringEquals": {
                    "dynamodb:Attributes": [
                        "column-name-1",
                        "column-name-2",
                        "column-name-3"
                    ]
                },
                "StringEqualsIfExists": {"dynamodb:Select": "SPECIFIC_ATTRIBUTES"}
            }
        }
    ]
}
```

Amazon DynamoDB: Allows Row-Level Access to DynamoDB Based on an Amazon Cognito ID

This example shows how you might create a policy that allows row-level access to the MyTable DynamoDB table based on an Amazon Cognito ID. This policy grants the permissions necessary to

complete this action from the AWS API or AWS CLI only. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

To use this policy, you must structure your DynamoDB table so the Cognito user ID is the partition key. For more information, see [Creating a Table](#) in the *Amazon DynamoDB Developer Guide*.

To learn more about DynamoDB condition keys, see [Specifying Conditions: Using Condition Keys](#) in the *Amazon DynamoDB Developer Guide*.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "dynamodb>DeleteItem",
                "dynamodb:GetItem",
                "dynamodb:PutItem",
                "dynamodb:Query",
                "dynamodb:UpdateItem"
            ],
            "Resource": ["arn:aws:dynamodb:*:*:table/MyTable"],
            "Condition": {
                "ForAllValues:StringEquals": {
                    "dynamodb:LeadingKeys": ["${cognito-identity.amazonaws.com:sub}"]
                }
            }
        }
    ]
}
```

Amazon EC2: Allows an EC2 Instance to Attach or Detach Volumes

This example shows how you might create a policy that can be attached to a service role. The policy allows the specified EC2 instance to attach or detach volumes. The instance is specified with an ARN in the Condition element. This policy grants the permissions necessary to complete this action from the AWS API or AWS CLI only. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

Amazon EC2 instances can run AWS commands with permissions granted by an [AWS service role for an EC2 instance \(p. 176\)](#) that is attached to the instance profile. You can attach this policy to the role, or add this statement to an existing policy. Only the instance identified by *INSTANCE-ID* can attach or detach volumes to instances in the account, including its own. Other statement elements that might exist in a larger policy are not impacted by the restriction of this one statement. For more information about creating IAM policies to control access to Amazon EC2 resources, see [Controlling Access to Amazon EC2 Resources](#) in the *Amazon EC2 User Guide for Linux Instances*.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AttachVolume",
                "ec2:DetachVolume"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:volume/*",

```

```

        "arn:aws:ec2::::instance/*"
    ],
    "Condition": {
        "ArnEquals": {"ec2:SourceInstanceARN": "arn:aws:ec2::::instance/instance-id"}
    }
}
]
}
}

```

Amazon EC2: Attach or Detach Amazon EBS Volumes to EC2 Instances Based on Tags

This example shows how you might create a policy that allows EBS volume owners to attach or detach their EBS volumes defined using the tag `VolumeUser` to EC2 instances that are tagged as development instances (`Department=Dev`). This policy grants the permissions necessary to complete this action from the AWS API or AWS CLI only. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

```

{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AttachVolume",
                "ec2:DetachVolume"
            ],
            "Resource": "arn:aws:ec2::::instance/*",
            "Condition": {
                "StringEquals": {"ec2:ResourceTag/Department": "Development"}
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AttachVolume",
                "ec2:DetachVolume"
            ],
            "Resource": "arn:aws:ec2::::volume/*",
            "Condition": {
                "StringEquals": {"ec2:ResourceTag/VolumeUser": "${aws:username}"}
            }
        }
    ]
}

```

Amazon EC2: Allows Launching EC2 Instances in a Specific Subnet, Programmatically and in the Console

This example shows how you might create a policy that allows listing information for all EC2 objects and launching EC2 instances in a specific subnet. This policy also grants the necessary permissions to complete this action on the console. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

```

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

```

```

        "Effect": "Allow",
        "Action": [
            "ec2:Describe*",
            "ec2:GetConsole*"
        ],
        "Resource": "*"
    },
    {
        "Effect": "Allow",
        "Action": "ec2:RunInstances",
        "Resource": [
            "arn:aws:ec2:*.*:subnet/subnet-subnet-id",
            "arn:aws:ec2:*.*:network-interface/*",
            "arn:aws:ec2:*.*:instance/*",
            "arn:aws:ec2:*.*:volume/*",
            "arn:aws:ec2:*.*:image/ami-*",
            "arn:aws:ec2:*.*:key-pair/*",
            "arn:aws:ec2:*.*:security-group/*"
        ]
    }
]
}

```

Amazon EC2: Allows Managing EC2 Security Groups Associated With a Specific VPC, Programmatically and in the Console

This example shows how you might create a policy that allows managing Amazon EC2 security groups associated with a specific virtual private cloud (VPC). This policy also grants the necessary permissions to complete this action on the console. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

```

{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AuthorizeSecurityGroupEgress",
                "ec2:AuthorizeSecurityGroupIngress",
                "ec2>DeleteSecurityGroup",
                "ec2:RevokeSecurityGroupEgress",
                "ec2:RevokeSecurityGroupIngress"
            ],
            "Resource": "arn:aws:ec2:*.*:security-group/*",
            "Condition": {
                "ArnEquals": {
                    "ec2:Vpc": "arn:aws:ec2:*.*:vpc/vpc-vpc-id"
                }
            }
        },
        {
            "Action": [
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeSecurityGroupReferences",
                "ec2:DescribeStaleSecurityGroups",
                "ec2:DescribeVpcs"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}

```

Amazon EC2: Allows Starting or Stopping EC2 Instances a User Has Tagged, Programmatically and in the Console

This example shows how you might create a policy that allows an IAM user to start or stop EC2 instances, but only if the instance tag `Owner` has the value of that user's user name. This policy also grants the necessary permissions to complete this action on the console.

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

EC2: Start or Stop Instances Based on Tags

This example shows how you might create a policy that allows starting or stopping instances with the tag key-value pair `Project = DataAnalytics`, but only by principals with the tag key-value pair `Department = Data`. This policy grants the permissions necessary to complete this action from the AWS API or AWS CLI only. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

The condition in the policy returns true if both parts of the condition are true. The instance must have the `Project=DataAnalytics` tag. In addition, the IAM principal (user or role) making the request must have the `Department=Data` tag.

Note

As a best practice, attach policies with the `aws:PrincipalTag` condition key to IAM groups, for the case where some users might have the specified tag and some might not.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "StartStopIfTags",
            "Effect": "Allow",
            "Action": [
                "ec2:StartInstances",
                "ec2:StopInstances",
                "ec2:DescribeTags"
            ],
            "Resource": "arn:aws:ec2:region:account-id:instance/*",
            "Condition": {
                "StringEquals": {
                    "aws:PrincipalTag/Project": "DataAnalytics"
                }
            }
        }
    ]
}
```

```

        "Condition": {
            "StringEquals": {
                "ec2:ResourceTag/Project

```

EC2: Start or Stop Instances Based on Matching Principal and Resource Tags

This example shows how you might create a policy that allows a principal to start or stop an Amazon EC2 instance when the instance's resource tag and the principal's tag have the same value for the tag key CostCenter. This policy grants the permissions necessary to complete this action from the AWS API or AWS CLI only. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

Note

As a best practice, attach policies with the `aws:PrincipalTag` condition key to IAM groups, for the case where some users might have the specified tag and some might not.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:startInstances",
                "ec2:stopInstances"
            ],
            "Resource": "*",
            "Condition": {"StringEquals":
                {"ec2:ResourceTag/CostCenter": "${aws:PrincipalTag/CostCenter}"}
            }
        }
    ]
}
```

Amazon EC2: Allows Full EC2 Access Within a Specific Region, Programmatically and in the Console

This example shows how you might create a policy that allows full EC2 access within a specific region. This policy also grants the necessary permissions to complete this action on the console. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": "ec2:*",
            "Resource": "*",
            "Effect": "Allow",
            "Condition": {
                "StringEquals": {
                    "ec2:Region": "region"
                }
            }
        }
    ]
}
```

}

Amazon EC2: Allows Starting or Stopping an EC2 Instance and Modifying a Security Group, Programmatically and in the Console

This example shows how you might create a policy that allows starting or stopping a specific EC2 instance and modifying a specific security group. This policy also grants the necessary permissions to complete this action on the console. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Action": [  
                "ec2:DescribeInstances",  
                "ec2:DescribeSecurityGroups",  
                "ec2:DescribeSecurityGroupReferences",  
                "ec2:DescribeStaleSecurityGroups"  
            ],  
            "Resource": "*",  
            "Effect": "Allow"  
        },  
        {  
            "Action": [  
                "ec2:AuthorizeSecurityGroupEgress",  
                "ec2:AuthorizeSecurityGroupIngress",  
                "ec2:RevokeSecurityGroupEgress",  
                "ec2:RevokeSecurityGroupIngress",  
                "ec2:StartInstances",  
                "ec2:StopInstances"  
            ],  
            "Resource": [  
                "arn:aws:ec2:*:*:instance/i-instance-id",  
                "arn:aws:ec2:*:*:security-group/sg-security-group-id"  
            ],  
            "Effect": "Allow"  
        }  
    ]  
}
```

Amazon EC2: Requires MFA (GetSessionToken) for Specific EC2 Operations

This example shows how you might create a policy that allows full access to all AWS API operations in Amazon EC2. However, it explicitly denies access to `StopInstances` and `TerminateInstances` API operations if the user is not authenticated using [multi-factor authentication \(MFA\)](#) (p. 119). To do this programmatically, the user must include optional `SerialNumber` and `TokenCode` values while calling the `GetSessionToken` operation. This operation returns temporary credentials that were authenticated using MFA. To learn more about `GetSessionToken`, see [GetSessionToken—Temporary Credentials for Users in Untrusted Environments](#) (p. 316).

What does this policy do?

- The `AllowAllActionsForEC2` statement allows all Amazon EC2 actions.
- The `DenyStopAndTerminateWhenMFAIsNotPresent` statement denies the `StopInstances` and `TerminateInstances` actions when the MFA context is missing. This means that the actions are

denied when the multi-factor authentication context is missing (meaning MFA was not used). A deny overrides the allow.

Note

The condition check for `MultiFactorAuthPresent` in the Deny statement should not be a `{"Bool": {"aws:MultiFactorAuthPresent": false}}` because that key is not present and cannot be evaluated when MFA is not used. So instead, use the `BoolIfExists` check to see whether the key is present before checking the value. For more information, see [...IfExists Condition Operators \(p. 620\)](#).

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAllActionsForEC2",
            "Effect": "Allow",
            "Action": "ec2:*",
            "Resource": "*"
        },
        {
            "Sid": "DenyStopAndTerminateWhenMFAIsNotPresent",
            "Effect": "Deny",
            "Action": [
                "ec2:StopInstances",
                "ec2:TerminateInstances"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {"aws:MultiFactorAuthPresent": false}
            }
        }
    ]
}
```

Amazon EC2: Limits Terminating EC2 Instances to an IP Address Range

This example shows how you might create a policy that limits EC2 instances by allowing the action, but explicitly denying access when the request comes from outside the specified IP range. The policy is useful when the IP addresses for your company are within the specified ranges. This policy grants the permissions necessary to complete this action from the AWS API or AWS CLI only. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

If this policy is used in combination with other policies that allow the `ec2:TerminateInstances` action (such as the [AmazonEC2FullAccess](#) AWS managed policy), then access is denied. This is because an explicit deny statement takes precedence over allow statements. For more information, see [the section called “Determining Whether a Request Is Allowed or Denied Within an Account” \(p. 638\)](#).

Important

The `aws:SourceIp` condition key denies access to an AWS service, such as AWS CloudFormation, that makes calls on your behalf. For more information about using the `aws:SourceIp` condition key, see [AWS Global Condition Context Keys \(p. 665\)](#).

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["ec2:TerminateInstances"],
```

```

        "Resource": ["*"]
    },
    {
        "Effect": "Deny",
        "Action": ["ec2:TerminateInstances"],
        "Condition": {
            "NotIpAddress": {
                "aws:SourceIp": [
                    "192.0.2.0/24",
                    "203.0.113.0/24"
                ]
            }
        },
        "Resource": ["*"]
    }
]
}

```

IAM: Access the Policy Simulator API

This example shows how you might create a policy that allows using the policy simulator API for policies attached to a user, group, or role in the current AWS account. This policy also allows access to simulate less sensitive policies passed to the API as strings. This policy grants the permissions necessary to complete this action from the AWS API or AWS CLI only.

```

{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "iam:GetContextKeysForCustomPolicy",
                "iam:GetContextKeysForPrincipalPolicy",
                "iam:SimulateCustomPolicy",
                "iam:SimulatePrincipalPolicy"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}

```

Note

To allow a user to access the policy simulator console to simulate policies attached to a user, group, or role in the current AWS account, see [IAM: Access the Policy Simulator Console \(p. 419\)](#).

IAM: Access the Policy Simulator Console

This example shows how you might create a policy that allows using the policy simulator console for policies attached to a user, group, or role in the current AWS account. This policy grants the permissions necessary to complete this action from the AWS API or AWS CLI only.

You can access the IAM Policy Simulator console at: <https://policysim.aws.amazon.com/>

```

{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "iam:GetGroup",
                "iam:GetUser"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}

```

```
        "iam:GetGroupPolicy",
        "iam:GetPolicy",
        "iam:GetPolicyVersion",
        "iam:GetRole",
        "iam:GetRolePolicy",
        "iam:GetUser",
        "iam:GetUserPolicy",
        "iam>ListAttachedGroupPolicies",
        "iam>ListAttachedRolePolicies",
        "iam>ListAttachedUserPolicies",
        "iam>ListGroups",
        "iam>ListGroupPolicies",
        "iam>ListGroupsForUser",
        "iam>ListRolePolicies",
        "iam>ListRoles",
        "iam>ListUserPolicies",
        "iam>ListUsers"
    ],
    "Effect": "Allow",
    "Resource": "*"
}
]
}
```

IAM: Assume Roles That Have a Specific Tag

This example shows how you might create a policy that allows an IAM user to assume roles with the tag key-value pair `Project = ExampleCorpABC`. This policy grants the permissions necessary to complete this action from the AWS API or AWS CLI only. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

If a role with this tag exists in the same account as the user, then the user can assume that role. If a role with this tag exists in an account other than the user's, it requires additional permissions. The cross-account role's trust policy must also allow the user or all members of the user's account to assume the role. For information about using roles for cross-account access, see [Providing Access to an IAM User in Another AWS Account That You Own \(p. 179\)](#).

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {"iam:ResourceTag/Project": "ExampleCorpABC"}
            }
        }
    ]
}
```

IAM: Allows and Denies Access to Multiple Services Programmatically and in the Console

This example shows how you might create a policy that allows full access to several services and limited self-managing access in IAM. It also denies access to the Amazon S3 logs bucket or the Amazon EC2 `i-1234567890abcdef0` instance. This policy also grants the necessary permissions to complete this action on the console. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

Warning

This policy allows full access to every action and resource in multiple services. This policy should be applied only to trusted administrators.

You can use this policy as a permissions boundary to define the maximum permissions that an identity-based policy can grant to an IAM user. For more information, see [Delegating Responsibility to Others Using Permissions Boundaries \(p. 374\)](#). When the policy is used as a permissions boundary for a user, the statements define the following boundaries:

- The `AllowServices` statement allows full access to the specified AWS services. This means that the user's actions in these services are limited only by the permissions policies that are attached to the user.
- The `AllowIAMConsoleForCredentials` statement allows access to list all IAM users. This access is necessary to navigate the `Users` page in the AWS Management Console. It also allows viewing the password requirements for the account, which is necessary for the user to change their own password.
- The `AllowManageOwnPasswordAndAccessKeys` statement allows the user to manage only their own console password and programmatic access keys. This is important because if another policy gives a user full IAM access, that user could then change their own or other users' permissions. This statement prevents that from happening.
- The `DenyS3Logs` statement explicitly denies access to the `logs` bucket. This policy enforces company restrictions on the user.
- The `DenyEC2Production` statement explicitly denies access to the `i-1234567890abcdef0` instance.

This policy does not allow access to other services or actions. When the policy is used as a permissions boundary on a user, even if other policies attached to the user allow those actions, AWS denies the request.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "AllowServices",  
            "Effect": "Allow",  
            "Action": [  
                "s3:*",  
                "cloudwatch:*",  
                "ec2:*"  
            ],  
            "Resource": "*"  
        },  
        {  
            "Sid": "AllowIAMConsoleForCredentials",  
            "Effect": "Allow",  
            "Action": [  
                "iam>ListUsers",  
                "iam:GetAccountPasswordPolicy"  
            ],  
            "Resource": "*"  
        },  
        {  
            "Sid": "AllowManageOwnPasswordAndAccessKeys",  
            "Effect": "Allow",  
            "Action": [  
                "iam:*AccessKey*",  
                "iam:ChangePassword",  
                "iam:GetUser",  
                "iam:*LoginProfile*"  
            ],  
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]  
        },  
    ]  
}
```

```
{
    "Sid": "DenyS3Logs",
    "Effect": "Deny",
    "Action": "s3:*",
    "Resource": [
        "arn:aws:s3:::logs",
        "arn:aws:s3:::logs/*"
    ]
},
{
    "Sid": "DenyEC2Production",
    "Effect": "Deny",
    "Action": "ec2:*",
    "Resource": "arn:aws:ec2:/*:instance/i-1234567890abcdef0"
}
]
```

IAM: Add a Specific Tag to a User With a Specific Tag

This example shows how you might create a policy that allows adding the tag key `Department` with the tag values `Marketing`, `Development`, or `QualityAssurance` to an IAM user. That user must already include the tag key-value pair `JobFunction = manager`. You can use this policy to require that a manager belong to only one of three departments. This policy also grants the necessary permissions to complete this action on the console. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

The `ListTagsForAllUsers` statement allows the viewing of tags for all users in your account.

The first condition in the `TagManagerWithSpecificDepartment` statement uses the `StringEquals` condition operator. The condition returns true if both parts of the condition are true. The user to be tagged must already have the `JobFunction=Manager` tag. The request must include the `Department` tag key with one of the listed tag values.

The second condition uses the `ForAllValues:StringEquals` condition operator. The condition returns true if all of the tag keys in the request match the key in the policy. This means that the only tag key in the request must be `Department`. For more information about using `ForAllValues`, see [Creating a Condition with Multiple Keys or Values \(p. 621\)](#).

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ListTagsForAllUsers",
            "Effect": "Allow",
            "Action": [
                "iam>ListUserTags",
                "iam>ListUsers"
            ],
            "Resource": "*"
        },
        {
            "Sid": "TagManagerWithSpecificDepartment",
            "Effect": "Allow",
            "Action": "iam:TagUser",
            "Resource": "*",
            "Condition": {"StringEquals": {
                "iam:ResourceTag/JobFunctionDepartmentMarketing",
                    "Development",
                    "QualityAssurance"
                ]
            }
        }
    ]
}
```

```

        ]
    },
    "ForAllValues:StringEquals": {"aws:TagKeys": "Department"}
}
]
}
}
]
```

IAM: Add a Specific Tag with Specific Values

This example shows how you might create a policy that allows adding only the tag key `CostCenter` and either the tag value `A-123` or the tag value `B-456` to any IAM user or role. You can use this policy to limit tagging to a specific tag key and set of tag values. This policy also grants the necessary permissions to complete this action on the console. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

The `ConsoleDisplay` statement allows the viewing of tags for all users and roles in your account.

The first condition in the `AddTag` statement uses the `StringEquals` condition operator. The condition returns true if the request includes the `CostCenter` tag key with one of the listed tag values.

The second condition uses the `ForAllValues:StringEquals` condition operator. The condition returns true if all of the tag keys in the request match the key in the policy. This means that the only tag key in the request must be `CostCenter`. For more information about using `ForAllValues`, see [Creating a Condition with Multiple Keys or Values \(p. 621\)](#).

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ConsoleDisplay",
            "Effect": "Allow",
            "Action": [
                "iam:GetRole",
                "iam:GetUser",
                "iam>ListRoles",
                "iam>ListRoleTags",
                "iam>ListUsers",
                "iam>ListUserTags"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AddTag",
            "Effect": "Allow",
            "Action": [
                "iam:TagUser",
                "iam:TagRole"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/CostCenterCostCenter"}
            }
        }
    ]
}
```

IAM: Create New Users Only With Specific Tags

This example shows how you might create a policy that allows the creation of IAM users but only with one or both of the `Department` and `JobFunction` tag keys. The `Department` tag key must have either the `Development` or `QualityAssurance` tag value. The `JobFunction` tag key must have the `Employee` tag value. You can use this policy to require that new users have a specific job function and department. This policy grants the permissions necessary to complete this action from the AWS API or AWS CLI only. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

The first condition in the statement uses the `StringEqualsIfExists` condition operator. If a tag with the `Department` or `JobFunction` key is present in the request, then the tag must have the specified value. If neither key is present, then this condition is evaluated as true. The only way that the condition evaluates as false is if one of the specified condition keys is present in the request, but has a different value than those allowed. For more information about using `IfExists`, see [...IfExists Condition Operators \(p. 620\)](#).

The second condition uses the `ForAllValues:StringEquals` condition operator. The condition returns true if there's a match between all every one of the specified tag keys specified in the request, and at least one value in the policy. This means that all of the tags in the request must be in this list. However, the request can include only one of the tags in the list. For example, you can create an IAM user with only the `Department=QualityAssurance` tag. However, you cannot create an IAM user with the `JobFunction=employee` tag and the `Project=core` tag. For more information about using `ForAllValues`, see [Creating a Condition with Multiple Keys or Values \(p. 621\)](#).

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "TagUsersWithOnlyTheseTags",
            "Effect": "Allow",
            "Action": [
                "iam:CreateUser",
                "iam:TagUser"
            ],
            "Resource": "*",
            "Condition": {
                "StringEqualsIfExists": {
                    "aws:RequestTag/DepartmentDevelopment",
                        "QualityAssurance"
                    ],
                    "aws:RequestTag/JobFunctionEmployee"
                },
                "ForAllValues:StringEquals": {
                    "aws:TagKeys": [
                        "Department",
                        "JobFunction"
                    ]
                }
            }
        }
    ]
}
```

IAM: Generate and Retrieve IAM Credential Reports

This example shows how you might create a policy that allows users to generate and download a report that lists all IAM users in their AWS account. The report includes the status of the users' credentials,

including passwords, access keys, MFA devices, and signing certificates. This policy grants the permissions necessary to complete this action from the AWS API or AWS CLI only.

For more information about credential reports, see [Getting Credential Reports for Your AWS Account \(p. 156\)](#).

```
{  
    "Version": "2012-10-17",  
    "Statement": {  
        "Effect": "Allow",  
        "Action": [  
            "iam:GenerateCredentialReport",  
            "iam:GetCredentialReport"  
        ],  
        "Resource": "*"  
    }  
}
```

IAM: Allows Managing a Group's Membership Programmatically and in the Console

This example shows how you might create a policy that allows updating the membership of the group called `MarketingTeam`. This policy also grants the necessary permissions to complete this action on the console. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

What does this policy do?

- The `ViewGroups` statement allows the user to list all the users and groups in the AWS Management Console. It also allows the user to view basic information about the users in the account. These permissions must be in their own statement because they do not support or do not need to specify a resource ARN. Instead the permissions specify `"Resource" : "*"`.
- The `ViewEditThisGroup` statement allows the user to view information about the `MarketingTeam` group, and to add and remove users from that group.

This policy does not allow the user to view or edit the permissions of the users or the `MarketingTeam` group.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "ViewGroups",  
            "Effect": "Allow",  
            "Action": [  
                "iam>ListGroups",  
                "iam>ListUsers",  
                "iam GetUser",  
                "iam>ListGroupsForUser"  
            ],  
            "Resource": "*"  
        },  
        {  
            "Sid": "ViewEditThisGroup",  
            "Effect": "Allow",  
            "Action": [  
                "iam>AddUserToGroup",  
                "iam RemoveUserFromGroup",  
                "iam GetGroup"  
            ],  
            "Resource": "*"  
        }  
    ]  
}
```

```
        "Resource": "arn:aws:iam::*:group/MarketingTeam"  
    ]  
}
```

IAM: Manage a Specific Tag

This example shows how you might create a policy that allows adding and removing the IAM tag with the tag key Department. This policy does not limit the value of the Department tag. This policy grants the permissions necessary to complete this action from the AWS API or AWS CLI only. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": [  
                "iam:TagUser",  
                "iam:TagRole",  
                "iam:UntagUser",  
                "iam:UntagRole"  
            ],  
            "Resource": "*",  
            "Condition": {"ForAllValues:StringEquals": {"aws:TagKeys": "Department"}  
        }  
    ]  
}
```

IAM: Pass an IAM Role to a Specific AWS Service

This example shows how you might create a policy that allows passing any IAM service role to the Amazon CloudWatch service. This policy grants the permissions necessary to complete this action from the AWS API or AWS CLI only. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

A service role is an IAM role that specifies an AWS service as the principal that can assume the role. This allows the service to assume the role and access resources in other services on your behalf. To allow Amazon CloudWatch to assume the role that you pass, you must specify the `cloudwatch.amazonaws.com` service principal as the principal in the trust policy of your role. The service principal is defined by the service. To learn the service principal for a service, see the documentation for that service. For some services, see [AWS Services That Work with IAM \(p. 586\)](#) and look for the services that have Yes in the **Service-Linked Role** column. Choose a Yes with a link to view the service-linked role documentation for that service. Search for `amazonaws.com` to view the service principal.

To learn more about passing a service role to a service, see [Granting a User Permissions to Pass a Role to an AWS Service \(p. 258\)](#).

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": "iam:PassRole",  
            "Resource": "*",  
            "Condition": {  
                "StringEquals": {"iam:PassedToService": "cloudwatch.amazonaws.com"}  
            }  
        }  
    ]  
}
```

IAM: Allows Read-Only Access to the IAM Console Without Reporting

This example shows how you might create a policy that allows IAM users to perform any IAM action that begins with the string Get, List, or Generate. As users work with the console, the console makes requests to IAM to list groups, users, roles, and policies, and to generate reports about those resources.

The asterisk acts as a wildcard. When you use `iam:Get*` in a policy, the resulting permissions include all IAM actions that begin with Get, such as `GetUser` and `GetRole`. Wildcards are useful if new types of entities are added to IAM in the future. In that case, the permissions granted by the policy automatically allow the user to list and get the details about those new entities.

This policy can't be used for reporting purposes.

```
{  
    "Version": "2012-10-17",  
    "Statement": {  
        "Effect": "Allow",  
        "Action": [  
            "iam:Get*",  
            "iam>List*"  
        ],  
        "Resource": "*"  
    }  
}
```

IAM: Allows Read-Only Access to the IAM Console

This example shows how you might create a policy that allows IAM users to perform any IAM action that begins with the string Get, List, or Generate. As users work with the IAM console, the console makes requests to list groups, users, roles, and policies, and to generate reports about those resources.

The asterisk acts as a wildcard. When you use `iam:Get*` in a policy, the resulting permissions include all IAM actions that begin with Get, such as `GetUser` and `GetRole`. Using a wildcard is beneficial, especially if new types of entities are added to IAM in the future. In that case, the permissions granted by the policy automatically allow the user to list and get the details about those new entities.

Use this policy for console access or for reporting purposes.

```
{  
    "Version": "2012-10-17",  
    "Statement": {  
        "Effect": "Allow",  
        "Action": [  
            "iam:Get*",  
            "iam>List*",  
            "iam:Generate*"  
        ],  
        "Resource": "*"  
    }  
}
```

IAM: Allows Specific IAM Users to Manage a Group Programmatically and in the Console

This example shows how you might create a policy that allows specific IAM users to manage the `AllUsers` group. This policy also grants the necessary permissions to complete this action on the

console. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

What does this policy do?

- The `AllowAllUsersToListAllGroups` statement allows listing all groups. This is necessary for console access. This permission must be in its own statement because it does not support a resource ARN. Instead the permissions specify "Resource" : "*".
- The `AllowAllUsersToViewAndManageThisGroup` statement allows all group actions that can be performed on the group resource type. It does not allow the `ListGroupsForUser` action, which can be performed on a user resource type and not a group resource type. For more information about the resource types that you can specify for an IAM action, see [Actions, Resources, and Condition Keys for AWS Identity and Access Management](#).
- The `LimitGroupManagementAccessToSpecificUsers` statement denies users with the specified names access to write and permissions management group actions. When a user specified in the policy attempts to make changes to the group, this statement does not deny the request. That request is allowed by the `AllowAllUsersToViewAndManageThisGroup` statement. If other users attempt to perform these operations, the request is denied. You can view the IAM actions that are defined with the **Write or Permissions management** access levels while creating this policy in the IAM console. To do this, switch from the **JSON** tab to the **Visual editor** tab. For more information about access levels, see [Actions, Resources, and Condition Keys for AWS Identity and Access Management](#).

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAllUsersToListAllGroups",
            "Effect": "Allow",
            "Action": "iam>ListGroups",
            "Resource": "*"
        },
        {
            "Sid": "AllowAllUsersToViewAndManageThisGroup",
            "Effect": "Allow",
            "Action": "iam:*Group*",
            "Resource": "arn:aws:iam::*:group/AllUsers"
        },
        {
            "Sid": "LimitGroupManagementAccessToSpecificUsers",
            "Effect": "Deny",
            "Action": [
                "iam>AddUserToGroup",
                "iam>CreateGroup",
                "iam>RemoveUserFromGroup",
                "iam>DeleteGroup",
                "iam>AttachGroupPolicy",
                "iam>UpdateGroup",
                "iam>DetachGroupPolicy",
                "iam>DeleteGroupPolicy",
                "iam>PutGroupPolicy"
            ],
            "Resource": "arn:aws:iam::*:group/AllUsers",
            "Condition": {
                "StringNotEquals": {
                    "aws:username": [
                        "srodriguez",
                        "mjackson",
                        "adesai"
                    ]
                }
            }
        }
    ]
}
```

```
        ]
    }
```

IAM: Allows Setting the Account Password Requirements Programmatically and in the Console

This example shows how you might create a policy that allows a user to view and update their account's password requirements. The password requirements specify the complexity requirements and mandatory rotation periods for the account members' passwords. This policy also grants the necessary permissions to complete this action on the console.

To learn how to set the account password requirements policy for your account, see [Setting an Account Password Policy for IAM Users \(p. 101\)](#).

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iam:GetAccountPasswordPolicy",
                "iam:UpdateAccountPasswordPolicy"
            ],
            "Resource": "*"
        }
    ]
}
```

IAM: Access the Policy Simulator API Based on User Path

This example shows how you might create a policy that allows using the policy simulator API only for those users that have the path `Department/Development`. This policy grants the permissions necessary to complete this action from the AWS API or AWS CLI only. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "iam:GetContextKeysForPrincipalPolicy",
                "iam:SimulatePrincipalPolicy"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:iam::*:user/Department/Development/*"
        }
    ]
}
```

Note

To create a policy that allows using the policy simulator console for those users that have the path `Department/Development`, see [IAM: Access the Policy Simulator Console Based on User Path \(p. 429\)](#).

IAM: Access the Policy Simulator Console Based on User Path

This example shows how you might create a policy that allows using the policy simulator console only for those users that have the path `Department/Development`. This policy grants the permissions

necessary to complete this action from the AWS API or AWS CLI only. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

You can access the IAM Policy Simulator at: <https://policysim.aws.amazon.com/>

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Action": [  
                "iam:GetPolicy",  
                "iam:GetUserPolicy"  
            ],  
            "Effect": "Allow",  
            "Resource": "*"  
        },  
        {  
            "Action": [  
                "iam:GetUser",  
                "iam>ListAttachedUserPolicies",  
                "iam>ListGroupsForUser",  
                "iam>ListUserPolicies",  
                "iam>ListUsers"  
            ],  
            "Effect": "Allow",  
            "Resource": "arn:aws:iam::*:user/Department/Development/*"  
        }  
    ]  
}
```

IAM: Allows IAM Users to Self-Manage an MFA Device

This example shows how you might create a policy that allows IAM users to self-manage their [multi-factor authentication \(MFA\) \(p. 119\)](#) device. This policy grants the permissions necessary to complete this action from the AWS API or AWS CLI only.

Note

If you add these permissions for a user that is signed in to AWS, they might need to sign out and back in to see these changes.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "AllowListActions",  
            "Effect": "Allow",  
            "Action": [  
                "iam>ListUsers",  
                "iam>ListVirtualMFADevices"  
            ],  
            "Resource": "*"  
        },  
        {  
            "Sid": "AllowIndividualUserToListOnlyTheirOwnMFA",  
            "Effect": "Allow",  
            "Action": [  
                "iam>ListMFADevices"  
            ],  
            "Resource": [  
                "arn:aws:iam::*:mfa/*",  
                "arn:aws:iam::*:user/${aws:username}"  
            ]  
        }  
    ]  
}
```

```

},
{
    "Sid": "AllowIndividualUserToManageTheirOwnMFA",
    "Effect": "Allow",
    "Action": [
        "iam>CreateVirtualMFADevice",
        "iam>DeleteVirtualMFADevice",
        "iam:EnableMFADevice",
        "iam:ResyncMFADevice"
    ],
    "Resource": [
        "arn:aws:iam::*:mfa/${aws:username}",
        "arn:aws:iam::*:user/${aws:username}"
    ]
},
{
    "Sid": "AllowIndividualUserToDeactivateOnlyTheirOwnMFAOnlyWhenUsingMFA",
    "Effect": "Allow",
    "Action": [
        "iam:DeactivateMFADevice"
    ],
    "Resource": [
        "arn:aws:iam::*:mfa/${aws:username}",
        "arn:aws:iam::*:user/${aws:username}"
    ],
    "Condition": {
        "Bool": {
            "aws:MultiFactorAuthPresent": "true"
        }
    }
},
{
    "Sid": "BlockMostAccessUnlessSignedInWithMFA",
    "Effect": "Deny",
    "NotAction": [
        "iam>CreateVirtualMFADevice",
        "iam:EnableMFADevice",
        "iam>ListMFADevices",
        "iam>ListUsers",
        "iam>ListVirtualMFADevices",
        "iam:ResyncMFADevice"
    ],
    "Resource": "*",
    "Condition": {
        "BoolIfExists": {
            "aws:MultiFactorAuthPresent": "false"
        }
    }
}
]
}

```

IAM: Allows IAM Users to Rotate Their Own Credentials Programmatically and in the Console

This example shows how you might create a policy that allows IAM users to rotate their own access keys, signing certificates, service specific credentials, and passwords. This policy also grants the necessary permissions to complete this action on the console.

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

```

```

    "Effect": "Allow",
    "Action": [
        "iam>ListUsers",
        "iam>GetAccountPasswordPolicy"
    ],
    "Resource": "*"
},
{
    "Effect": "Allow",
    "Action": [
        "iam>*AccessKey*",
        "iam>ChangePassword",
        "iam GetUser",
        "iam>*ServiceSpecificCredential*",
        "iam>*SigningCertificate*"
    ],
    "Resource": ["arn:aws:iam::*:user/${aws:username}"]
}
]
}

```

To learn how a user can change their own password in the console, see [the section called “How an IAM User Changes Their Own Password” \(p. 109\)](#).

IAM: View Service Last Accessed Data for an Organizations Policy

This example shows how you might create a policy that allows viewing service last accessed data for a specific Organizations policy. This policy allows retrieving data for the service control policy (SCP) with the *p-policy123* ID. The person who generates and views the report must be authenticated using AWS Organizations master account credentials. This policy allows the requester to retrieve the data for any Organizations entity in their organization. This policy also grants the necessary permissions to complete this action on the console. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

For important information about service last accessed data, including permissions required, troubleshooting, and supported Regions, see [Refining Permissions Using Service Last Accessed Data \(p. 475\)](#).

```

{
    "Version": "2012-10-17",
    "Statement": {
        "Sid": "AllowOrgsReadOnlyAndIamGetReport",
        "Effect": "Allow",
        "Action": [
            "iam>GetOrganizationsAccessReport",
            "organizations>Describe*",
            "organizations>List*"
        ],
        "Resource": "*"
    },
    {
        "Sid": "AllowGenerateReportOnlyForThePolicy",
        "Effect": "Allow",
        "Action": "iam>GenerateOrganizationsAccessReport",
        "Resource": "*",
        "Condition": {
            "StringEquals": {"iam>OrganizationsPolicyId": "p-policy123"}
        }
    }
}

```

IAM: Limits Managed Policies That Can Be Applied to an IAM User, Group, or Role

This example shows how you might create a policy that limits customer managed and AWS managed policies that can be applied to an IAM user, group, or role. This policy grants the permissions necessary to complete this action from the AWS API or AWS CLI only. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": [  
                "iam:AttachUserPolicy",  
                "iam:DetachUserPolicy"  
            ],  
            "Resource": "*",  
            "Condition": {  
                "ArnEquals": {  
                    "iam:PolicyARN": [  
                        "arn:aws:iam::*:policy/policy-name-1",  
                        "arn:aws:iam::*:policy/policy-name-2"  
                    ]  
                }  
            }  
        }  
    }  
}
```

AWS Lambda: Allows a Lambda function to access an Amazon DynamoDB table

This example shows how you might create a policy that allows read and write access to a specific Amazon DynamoDB table. The policy also allows writing log files to CloudWatch Logs. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

To use this policy, attach the policy to a Lambda [service role \(p. 236\)](#). A service role is a role that you create in your account to allow a service to perform actions on your behalf. That service role must include AWS Lambda as the principal in the trust policy. For details about how to use this policy, see [How to Create an AWS IAM Policy to Grant AWS Lambda Access to an Amazon DynamoDB Table](#) in the [AWS Security Blog](#).

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "ReadWriteTable",  
            "Effect": "Allow",  
            "Action": [  
                "dynamodb:BatchGetItem",  
                "dynamodb:GetItem",  
                "dynamodb:Query",  
                "dynamodb:Scan",  
                "dynamodb:BatchWriteItem",  
                "dynamodb:PutItem",  
                "dynamodb:UpdateItem"  
            ],  
            "Resource": "arn:aws:dynamodb:*:*:table/SampleTable"  
        },  
        {  
            "Effect": "Allow",  
            "Action": ["logs:CreateLogStream", "logs:PutLogEvents"],  
            "Resource": "arn:aws:logs:*:*:log-group:SampleLogGroup:*"  
        }  
    ]  
}
```

```

        "Sid": "GetStreamRecords",
        "Effect": "Allow",
        "Action": "dynamodb:GetRecords",
        "Resource": "arn:aws:dynamodb:*:*:table/SampleTable/stream/* "
    },
    {
        "Sid": "WriteLogStreamsAndGroups",
        "Effect": "Allow",
        "Action": [
            "logs:CreateLogStream",
            "logs:PutLogEvents"
        ],
        "Resource": "*"
    },
    {
        "Sid": "CreateLogGroup",
        "Effect": "Allow",
        "Action": "logs:CreateLogGroup",
        "Resource": "*"
    }
]
}

```

Amazon RDS: Allows Full RDS Database Access Within a Specific Region

This example shows how you might create a policy that allows full RDS database access within a specific Region. This policy grants the permissions necessary to complete this action from the AWS API or AWS CLI only. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

```

{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "rds:*",
            "Resource": ["arn:aws:rds:region:*:*"]
        },
        {
            "Effect": "Allow",
            "Action": ["rds:Describe*"],
            "Resource": ["*"]
        }
    ]
}

```

Amazon RDS: Allows Restoring RDS Databases, Programmatically and In the Console

This example shows how you might create a policy that allows restoring RDS databases. This policy also grants the necessary permissions to complete this action on the console.

```

{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:Describe*",

```

```

        "rds>CreateDBParameterGroup",
        "rds>CreateDBSnapshot",
        "rds>DeleteDBSnapshot",
        "rds>Describe*",
        "rds>DownloadDBLogFilePortion",
        "rds>List*",
        "rds>ModifyDBInstance",
        "rds>ModifyDBParameterGroup",
        "rds>ModifyOptionGroup",
        "rds>RebootDBInstance",
        "rds>RestoreDBInstanceFromDBSnapshot",
        "rds>RestoreDBInstanceToPointInTime"
    ],
    "Resource": "*"
}
]
}

```

Amazon RDS: Allows Tag Owners Full Access to RDS Resources That They Have Tagged

This example shows how you might create a policy that allows tag owners full access to RDS resources that they have tagged. This policy grants the permissions necessary to complete this action from the AWS API or AWS CLI only.

```

{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "rds>Describe*",
                "rds>List*"
            ],
            "Effect": "Allow",
            "Resource": "*"
        },
        {
            "Action": [
                "rds>DeleteDBInstance",
                "rds>RebootDBInstance",
                "rds>ModifyDBInstance"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:db-tag/Owner": "${aws:username}"}
            }
        },
        {
            "Action": [
                "rds>ModifyOptionGroup",
                "rds>DeleteOptionGroup"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:og-tag/Owner": "${aws:username}"}
            }
        },
        {
            "Action": [
                "rds>ModifyDBParameterGroup",
                "rds>ResetDBParameterGroup"
            ]
        }
    ]
}

```

```

        ],
        "Effect": "Allow",
        "Resource": "*",
        "Condition": {
            "StringEqualsIgnoreCase": {"rds:pg-tag/Owner": "${aws:username}"}
        }
    },
    {
        "Action": [
            "rds:AuthorizeDBSecurityGroupIngress",
            "rds:RevokeDBSecurityGroupIngress",
            "rds:DeleteDBSecurityGroup"
        ],
        "Effect": "Allow",
        "Resource": "*",
        "Condition": {
            "StringEqualsIgnoreCase": {"rds:secgrp-tag/Owner": "${aws:username}"}
        }
    },
    {
        "Action": [
            "rds:DeleteDBSnapshot",
            "rds:RestoreDBInstanceFromDBSnapshot"
        ],
        "Effect": "Allow",
        "Resource": "*",
        "Condition": {
            "StringEqualsIgnoreCase": {"rds:snapshot-tag/Owner": "${aws:username}"}
        }
    },
    {
        "Action": [
            "rds:ModifyDBSubnetGroup",
            "rds:DeleteDBSubnetGroup"
        ],
        "Effect": "Allow",
        "Resource": "*",
        "Condition": {
            "StringEqualsIgnoreCase": {"rds:subgrp-tag/Owner": "${aws:username}"}
        }
    },
    {
        "Action": [
            "rds:ModifyEventSubscription",
            "rds:AddSourceIdentifierToSubscription",
            "rds:RemoveSourceIdentifierFromSubscription",
            "rds:DeleteEventSubscription"
        ],
        "Effect": "Allow",
        "Resource": "*",
        "Condition": {
            "StringEqualsIgnoreCase": {"rds:es-tag/Owner": "${aws:username}"}
        }
    }
]
}

```

Amazon S3: Allows Amazon Cognito Users to Access Objects in Their Bucket

This example shows how you might create a policy that allows Amazon Cognito users to access objects in a specific S3 bucket. This policy allows access only to objects with a name that includes cognito, the name of the application, and the federated user's ID, represented by the \${cognito-

identity.amazonaws.com:sub} variable. This policy grants the permissions necessary to complete this action from the AWS API or AWS CLI only. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

Note

The 'sub' value used in the object key is not the user's sub value in the User Pool, it is the identity id associated with the user in the Identity Pool.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "ListYourObjects",  
            "Effect": "Allow",  
            "Action": "s3>ListBucket",  
            "Resource": ["arn:aws:s3::::bucket-name"],  
            "Condition": {  
                "StringLike": {  
                    "s3:prefix": ["cognito/application-name/${cognito-  
identity.amazonaws.com:sub}"]  
                }  
            }  
        },  
        {  
            "Sid": "ReadWriteDeleteYourObjects",  
            "Effect": "Allow",  
            "Action": [  
                "s3.GetObject",  
                "s3:PutObject",  
                "s3>DeleteObject"  
            ],  
            "Resource": [  
                "arn:aws:s3::::bucket-name/cognito/application-name/${cognito-  
identity.amazonaws.com:sub}",  
                "arn:aws:s3::::bucket-name/cognito/application-name/${cognito-  
identity.amazonaws.com:sub}/*"  
            ]  
        }  
    ]  
}
```

Amazon Cognito provides authentication, authorization, and user management for your web and mobile apps. Your users can sign in directly with a user name and password, or through a third party such as Facebook, Amazon, or Google.

The two main components of Amazon Cognito are user pools and identity pools. User pools are user directories that provide sign-up and sign-in options for your app users. Identity pools enable you to grant your users access to other AWS services. You can use identity pools and user pools separately or together.

For more information about Amazon Cognito, see the following:

- [Amazon Cognito Identity](#) in the *AWS Mobile SDK for Android Developer Guide*
- [Amazon Cognito Identity](#) in the *AWS Mobile SDK for iOS Developer Guide*

Amazon S3: Allows Federated Users Access to Their S3 Home Directory, Programmatically and In the Console

This example shows how you might create a policy that allows federated users to access their own home directory bucket object in S3. The home directory is a bucket that includes a home folder and folders for

individual federated users. This policy also grants the necessary permissions to complete this action on the console. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

The `#{aws:userid}` variable in this policy resolves to `role-id:specified-name`. The `role-id` part of the federated user ID is a unique identifier assigned to the federated user's role during creation. For more information, see [Unique Identifiers \(p. 579\)](#). The `specified-name` is the `RoleSessionName parameter` passed to the `AssumeRoleWithWebIdentity` request when the federated user assumed their role.

You can view the role ID using the AWS CLI command `aws iam get-role --role-name specified-name`. For example, imagine that you specify the friendly name `John` and the CLI returns the role ID `AROAXXT2NJT7D3SIQN7Z6`. In this case, the federated user ID is `AROAXXT2NJT7D3SIQN7Z6:John`. This policy then allows the federated user `John` to access the Amazon S3 bucket with prefix `AROAXXT2NJT7D3SIQN7Z6:John`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3>ListAllMyBuckets",
                "s3:GetBucketLocation"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "s3>ListBucket",
            "Resource": "arn:aws:s3:::bucket-name",
            "Condition": {
                "StringLike": {
                    "s3:prefix": [
                        "",
                        "home/",
                        "home/${aws:userid}/*"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::bucket-name/home/${aws:userid}",
                "arn:aws:s3:::bucket-name/home/${aws:userid}/*"
            ]
        }
    ]
}
```

Amazon S3: S3 Bucket Access, but Production Bucket Denied Without Recent MFA

This example shows how you might create a policy that allows an Amazon S3 administrator to access any bucket, including updating, adding, and deleting objects. However, it explicitly denies access to the `Production` bucket if the user has not signed in using [multi-factor authentication \(MFA\) \(p. 119\)](#) within the last thirty minutes. This policy grants the permissions necessary to perform this action in the console or programmatically using the AWS CLI or AWS API. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

This policy never allows programmatic access to the Production bucket using long-term user access keys. This is accomplished using the `aws:MultiFactorAuthAge` condition key with the `NumericGreaterThanIfExists` condition operator. This policy condition returns `true` if MFA is not present or if the age of the MFA is greater than 30 minutes. In those situations, access is denied. To access the Production bucket programmatically, the S3 administrator must use temporary credentials that were generated in the last 30 minutes using the [GetSessionToken \(p. 316\)](#) API operation.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ListAllS3Buckets",
            "Effect": "Allow",
            "Action": ["s3>ListAllMyBuckets"],
            "Resource": "arn:aws:s3:::/*"
        },
        {
            "Sid": "AllowBucketLevelActions",
            "Effect": "Allow",
            "Action": [
                "s3>ListBucket",
                "s3:GetBucketLocation"
            ],
            "Resource": "arn:aws:s3:::/*"
        },
        {
            "Sid": "AllowBucketObjectActions",
            "Effect": "Allow",
            "Action": [
                "s3>PutObject",
                "s3>PutObjectAcl",
                "s3>GetObject",
                "s3>GetObjectAcl",
                "s3>DeleteObject"
            ],
            "Resource": "arn:aws:s3:::/*/*"
        },
        {
            "Sid": "RequireMFAForProductionBucket",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::Production/*",
                "arn:aws:s3:::Production"
            ],
            "Condition": {
                "NumericGreaterThanIfExists": {"aws:MultiFactorAuthAge": "1800"}
            }
        }
    ]
}
```

Amazon S3: Allows IAM Users Access to Their S3 Home Directory, Programmatically and In the Console

This example shows how you might create a policy that allows IAM users to access their own home directory bucket object in S3. The home directory is a bucket that includes a home folder and folders for individual users. This policy also grants the necessary permissions to complete this action on the console. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3>ListAllMyBuckets",
                "s3>GetBucketLocation"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "s3>ListBucket",
            "Resource": "arn:aws:s3:::<bucket-name>",
            "Condition": {
                "StringLike": {
                    "s3:prefix": [
                        "",
                        "home/",
                        "home/${aws:username}/*"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::<bucket-name>/home/${aws:username}",
                "arn:aws:s3:::<bucket-name>/home/${aws:username}/*"
            ]
        }
    ]
}
```

Amazon S3: Limits Managing to a Specific S3 Bucket

This example shows how you might create a policy that limits managing an S3 bucket by allowing all S3 actions on the specific bucket, but explicitly denying access to every AWS service except Amazon S3. This policy also denies access to actions that can't be performed on an S3 bucket, such as `s3>ListAllMyBuckets` or `s3>GetObject`. This policy grants the permissions necessary to complete this action from the AWS API or AWS CLI only. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

If this policy is used in combination with other policies (such as the [AmazonS3FullAccess](#) or [AmazonEC2FullAccess](#) AWS managed policies) that allow actions denied by this policy, then access is denied. This is because an explicit deny statement takes precedence over allow statements. For more information, see [the section called “Determining Whether a Request Is Allowed or Denied Within an Account” \(p. 638\)](#).

Warning

[NotAction \(p. 607\)](#) and [NotResource \(p. 610\)](#) are advanced policy elements that must be used with care. This policy denies access to every AWS service except Amazon S3. If you attach this policy to a user, any other policies that grant permissions to other services are ignored and access is denied.

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

```

```
    "Effect": "Allow",
    "Action": "s3:*",
    "Resource": [
        "arn:aws:s3:::bucket-name",
        "arn:aws:s3:::bucket-name/*"
    ]
},
{
    "Effect": "Deny",
    "NotAction": "s3:*",
    "NotResource": [
        "arn:aws:s3:::bucket-name",
        "arn:aws:s3:::bucket-name/*"
    ]
}
]
```

Amazon S3: Allows Read and Write Access to Objects in an S3 Bucket

This example shows how you might create a policy that allows `Read` and `Write` access to objects in a specific S3 bucket. This policy grants the permissions necessary to complete this action from the AWS API or AWS CLI only. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

The `s3:*Object` action uses a wildcard as part of the action name. The `AllObjectActions` statement allows the `GetObject`, `DeleteObject`, `PutObject`, and any other Amazon S3 action that ends with the word "Object".

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ListObjectsInBucket",
            "Effect": "Allow",
            "Action": ["s3>ListBucket"],
            "Resource": ["arn:aws:s3:::bucket-name"]
        },
        {
            "Sid": "AllObjectActions",
            "Effect": "Allow",
            "Action": "s3:*Object",
            "Resource": ["arn:aws:s3:::bucket-name/*"]
        }
    ]
}
```

Note

To allow `Read` and `Write` access to an object in an Amazon S3 bucket and also include additional permissions for console access, see [Amazon S3: Allows Read and Write Access to Objects in an S3 Bucket, Programmatically and in the Console \(p. 441\)](#).

Amazon S3: Allows Read and Write Access to Objects in an S3 Bucket, Programmatically and in the Console

This example shows how you might create a policy that allows `Read` and `Write` access to objects in a specific S3 bucket. This policy also grants the necessary permissions to complete this action on the

console. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

The `s3:*Object` action uses a wildcard as part of the action name. The `AllObjectActions` statement allows the `GetObject`, `DeleteObject`, `PutObject`, and any other Amazon S3 action that ends with the word "Object".

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "ConsoleAccess",  
            "Effect": "Allow",  
            "Action": [  
                "s3:GetAccountPublicAccessBlock",  
                "s3:GetBucketAcl",  
                "s3:GetBucketLocation",  
                "s3:GetBucketPolicyStatus",  
                "s3:GetBucketPublicAccessBlock",  
                "s3>ListAllMyBuckets"  
            ],  
            "Resource": "*"  
        },  
        {  
            "Sid": "ListObjectsInBucket",  
            "Effect": "Allow",  
            "Action": "s3>ListBucket",  
            "Resource": ["arn:aws:s3:::bucket-name"]  
        },  
        {  
            "Sid": "AllObjectActions",  
            "Effect": "Allow",  
            "Action": "s3:*Object",  
            "Resource": ["arn:aws:s3:::bucket-name/*"]  
        }  
    ]  
}
```

Managing IAM Policies

IAM gives you the tools to create and manage all types of IAM policies (managed policies and inline policies). To add permissions to an IAM identity (IAM user, group, or role), you create a policy and then attach the policy to the identity. You can attach multiple policies to an identity, and each policy can contain multiple permissions.

Consult these resources for details:

- For more information about the different types of IAM policies, see [Policies and Permissions \(p. 356\)](#).
- For general information about using policies within IAM, see [Access Management \(p. 355\)](#).
- For information about how permissions are evaluated when multiple policies are in effect for a given IAM identity, see [Policy Evaluation Logic \(p. 636\)](#).
- For information about policy size and naming limitations, see [IAM and STS Limits \(p. 581\)](#).

Topics

- [Creating IAM Policies \(p. 443\)](#)
- [Validating JSON Policies \(p. 448\)](#)

- [Testing IAM Policies with the IAM Policy Simulator \(p. 448\)](#)
- [Adding and Removing IAM Identity Permissions \(p. 457\)](#)
- [Versioning IAM Policies \(p. 465\)](#)
- [Editing IAM Policies \(p. 468\)](#)
- [Deleting IAM Policies \(p. 472\)](#)
- [Refining Permissions Using Service Last Accessed Data \(p. 475\)](#)

Creating IAM Policies

A [policy \(p. 356\)](#) is an entity that, when attached to an identity or resource, defines their permissions. Policies are stored in AWS as JSON documents and are attached to principals as *identity-based policies* in IAM. You can attach an identity-based policy to a principal (or identity), such as an IAM group, user, or role. Identity-based policies include AWS managed policies, customer managed policies, and [inline policies \(p. 364\)](#).

You can create a new IAM policy in the AWS Management Console using one of the following methods:

- **Import** — You can import a managed policy within your account and then edit the policy to customize it to your specific requirements. A managed policy can be an AWS managed policy, or a customer managed policy that you created previously.
- **Visual editor** — You can construct a new policy from scratch in the visual editor. If you use the visual editor, you do not have to understand JSON syntax.
- **JSON** — In the **JSON** tab, you can create a policy using JSON syntax. You can type a new JSON policy document or paste an [example policy \(p. 394\)](#).

You can create an inline policy in the AWS Management Console. An inline policy is one that you create and embed directly to an IAM group, user, or role. To learn more, see [Adding and Removing IAM Identity Permissions \(p. 457\)](#). You cannot create AWS managed policies.

For information about policy size limitations and other quotas, see [IAM and STS Limits \(p. 581\)](#).

Topics

- [Creating IAM Policies \(Console\) \(p. 443\)](#)
- [Creating IAM Policies \(AWS CLI\) \(p. 447\)](#)
- [Creating IAM Policies \(AWS API\) \(p. 447\)](#)

Creating IAM Policies (Console)

No matter which way you choose to create a policy, they all start the same way:

To start creating a new policy (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane on the left, choose **Policies**.

If this is your first time choosing **Policies**, the **Welcome to Managed Policies** page appears. Choose **Get Started**.

3. Choose **Create policy**.
4. Choose one of the following ways to create the policy. Then follow the steps in the corresponding procedure:

- [Importing Existing Managed Policies \(p. 444\)](#)
- [Creating Policies with the Visual Editor \(p. 445\)](#)
- [Creating Policies on the JSON Tab \(p. 446\)](#)

Importing Existing Managed Policies

An easy way to create a new policy is to import an existing managed policy within your account that has at least some of the permissions that you need. You can then customize the policy to match it to your new requirements.

You cannot import an inline policy. To learn about the difference between managed and inline policies, see [Managed Policies and Inline Policies \(p. 364\)](#).

To import an existing managed policy in the visual editor

1. Start the **Create policy** wizard by following the steps in [Creating IAM Policies \(Console\) \(p. 443\)](#). Choose the **Visual editor** tab, and then on the right side of the page, choose **Import managed policy**.
 2. In the **Import managed policies** window, choose the managed policies that most closely match the policy that you want to include in your new policy. You can use the **Filter** menu or type in the search box at the top to limit the results in the list of policies.
 3. Choose **Import**.
- The imported policies are added in new permission blocks at the bottom of your policy.
4. Use the **Visual editor** or choose **JSON** to customize your policy. Then choose **Review policy**.

Note

You can switch between the **Visual editor** and **JSON** tabs any time. However, if you make changes or choose **Review policy** in the **Visual editor** tab, IAM might restructure your policy to optimize it for the visual editor. For more information, see [Policy Restructuring \(p. 548\)](#).

5. On the **Review** page, type a **Name** and a **Description** (optional) for the policy that you are creating. You cannot edit these settings later. Review the policy **Summary** and then choose **Create policy** to save your work.

To import an existing managed policy in the JSON tab

1. Start the **Create policy** wizard by following the steps in [Creating IAM Policies \(Console\) \(p. 443\)](#). Choose the **JSON** tab, and then on the right side of the page, choose **Import managed policy**.
2. In the **Import managed policies** window, choose the managed policies that most closely match the policy that you want to include in your new policy. You can use the **Filter** menu or type in the search box at the top to limit the results in the list of policies.
3. Choose **Import**.

Statements from the imported policies are added to the bottom of your JSON policy.

4. Customize your policy in **JSON**, or choose the **Visual editor**. Then choose **Review policy**.

Note

You can switch between the **Visual editor** and **JSON** tabs any time. However, if you make changes or choose **Review policy** in the **Visual editor** tab, IAM might restructure your policy to optimize it for the visual editor. For more information, see [Policy Restructuring \(p. 548\)](#).

5. On the **Review policy** page, type a **Name** and a **Description** (optional) for the policy that you are creating. You cannot edit these later. Review the policy **Summary** and then choose **Create policy** to save your work.

After you create a policy, you can attach it to your groups, users, or roles. For more information, see [Adding and Removing IAM Identity Permissions \(p. 457\)](#).

Creating Policies with the Visual Editor

The visual editor in the IAM console guides you through creating a policy without having to write JSON syntax. To view an example of using the visual editor to create a policy, see [the section called "Controlling Access to Identities" \(p. 383\)](#).

To use the visual editor to create a policy

1. Start the **Create Policy** wizard by following the steps in [Creating IAM Policies \(Console\) \(p. 443\)](#).
2. On the **Visual editor** tab, choose **Choose a service**. Then choose an AWS service to add to the policy. You can use the search box at the top to limit the results in the list of services. You can choose only one service within a visual editor permission block. To grant access to more than one service, add multiple permission blocks by choosing **Add additional permissions**.
3. Choose **Select actions** and then choose the actions to add to the policy. The visual editor shows the actions available in the service that you selected in the previous step.

You can choose actions in the following ways:

- Use check boxes to select all actions for the service or all actions in one of the predefined **Access level** groups.
- Expand each of the **Access level** groups to choose individual actions.
- Choose **add actions** to type a specific action or use wildcards (*) to specify multiple actions.

By default, the policy that you are creating allows the actions that you choose. To deny the chosen actions instead, choose **Switch to deny permissions**. Because [IAM denies by default \(p. 636\)](#), we recommend as a security best practice that you allow permissions to only those actions and resources that a user needs. This is sometimes called "whitelisting." You should create a JSON statement to deny permissions ("blacklisting") only if you want to override a permission separately allowed by another statement or policy. We recommend that you limit the number of deny permissions to a minimum because they can increase the difficulty of troubleshooting permissions.

4. If the selected service and the actions that you selected in the previous steps do not support choosing [specific resources \(p. 388\)](#), then **All resources** is selected for you. In that case, you cannot edit this section.

If you chose one or more actions that support [resource-level permissions \(p. 388\)](#), then the visual editor lists those resources. You can then choose **Resources** to specify resources for your policy.

You can choose resources in the following ways:

- Choose **Add ARN** to provide the details about your resource. Instead of typing a value, you can also choose **Any** to provide permissions for any value for the specified setting. For example, if you selected the Amazon EC2 **Read** access level group, then the actions in your policy support the **instance** resource type. You must provide the **Region**, **Account**, and **InstanceId** values for your resource. If you provide your account ID but choose **Any** for the Region and instance ID, then the policy grants permissions to any instance in your account.
 - Choose **Add ARN** to specify resources by their [Amazon Resource Name \(ARN\)](#). You can include a wildcard (*) in any field of the ARN (between each pair of colons). For more information, see [IAM JSON Policy Elements: Resource \(p. 609\)](#).
 - Choose **Any** from the far right of the resource section to grant permissions to any resources of a particular type.
 - Choose **All resources** to choose all resources for that service.
5. (Optional) Choose **Specify request conditions (optional)** to add conditions to the policy that you are creating. Conditions limit a JSON policy statement's effect. For example, you can specify that

a user is allowed to perform the actions on the resources only when that user's request happens within a certain time range. You can also use commonly used conditions to limit whether a user must be authenticated using a multi-factor authentication (MFA) device. Or you can require that the request originate from within a certain range of IP addresses. For lists of all of the context keys that you can use in a policy condition, see [Actions, Resources, and Condition Keys for AWS Services \(p. 686\)](#).

You can choose conditions in the following ways:

- Use check boxes to select commonly used conditions.
- Choose **Add condition** to specify other conditions. Choose the condition's **Condition Key**, **Qualifier**, and **Operator**, and then type a **Value**. To add more than one value, choose **Add new value**. You can consider the values as being connected by a logical "OR" operator. When you are finished, choose **Add**.

To add more than one condition, choose **Add condition** again. Repeat as needed. Each condition applies only to this one visual editor permission block. All the conditions must be true for the permission block to be considered a match. In other words, consider the conditions to be connected by a logical "AND" operator.

For more information about the **Condition** element, see [IAM JSON Policy Elements: Condition \(p. 611\)](#) in the [IAM JSON Policy Reference \(p. 597\)](#).

6. To add more permission blocks, choose **Add additional permissions**. For each block, repeat steps 2 through 5.
7. When you are finished, choose **Review policy**.

Note

You can switch between the **Visual editor** and **JSON** tabs any time. However, if you make changes or choose **Review policy** in the **Visual editor** tab, IAM might restructure your policy to optimize it for the visual editor. For more information, see [Policy Restructuring \(p. 548\)](#).

8. On the **Review policy** page, type a **Name** and a **Description** (optional) for the policy that you are creating. Review the policy summary to make sure that you have granted the intended permissions, and then choose **Create policy** to save your new policy.

After you create a policy, you can attach it to your groups, users, or roles. For more information, see [Adding and Removing IAM Identity Permissions \(p. 457\)](#).

Creating Policies on the JSON Tab

You can type or paste policies in JSON by choosing the **JSON** tab. This method is useful for copying an [example policy \(p. 394\)](#) to use in your account. Or, you can type your own JSON policy document in the JSON editor. You can also use the **JSON** tab to toggle between the visual editor and JSON to compare the views.

A JSON [policy \(p. 356\)](#) document consists of one or more statements. Each statement should contain all the actions that share the same effect (Allow or Deny) and support the same resources and conditions. If one action requires you to specify all resources ("*") and another action supports the [Amazon Resource Name \(ARN\)](#) of a specific resource, they must be in two separate JSON statements. For general information about IAM policies, see [Policies and Permissions \(p. 356\)](#). For information about the IAM policy language, see [IAM JSON Policy Reference \(p. 597\)](#).

To use the JSON policy editor to create a policy

1. Start the **Create Policy** wizard by following the steps in [Creating IAM Policies \(Console\) \(p. 443\)](#).
2. Choose the **JSON** tab.

3. Type or paste a JSON policy document. For details about the IAM policy language, see [IAM JSON Policy Reference \(p. 597\)](#).
4. When you are finished, choose **Review policy**. The [Policy Validator \(p. 448\)](#) reports any syntax errors.

Note

You can switch between the **Visual editor** and **JSON** tabs any time. However, if you make changes or choose **Review policy** in the **Visual editor** tab, IAM might restructure your policy to optimize it for the visual editor. For more information, see [Policy Restructuring \(p. 548\)](#).

5. On the **Review policy** page, type a **Name** and a **Description** (optional) for the policy that you are creating. Review the policy **Summary** to see the permissions that are granted by your policy. Then choose **Create policy** to save your work.

After you create a policy, you can attach it to your groups, users, or roles. For more information, see [Adding and Removing IAM Identity Permissions \(p. 457\)](#).

Creating IAM Policies (AWS CLI)

You can create an IAM policy or an inline policy using the AWS Command Line Interface (AWS CLI).

To create a customer managed policy (AWS CLI)

Use the following command:

- [create-policy](#)

To create an inline policy for an IAM identity (group, user or role) (AWS CLI)

Use one of the following commands:

- [put-group-policy](#)
- [put-role-policy](#)
- [put-user-policy](#)

Note

You can embed an inline policy for a [service-linked role \(p. 176\)](#) only in the service that depends on the role. See the [AWS documentation](#) for your service to see whether it supports this feature.

Creating IAM Policies (AWS API)

You can create an IAM policy or an inline policy using the AWS API.

To create a customer managed policy (AWS API)

Call the following operation:

- [CreatePolicy](#)

To create an inline policy for an IAM identity (group, user, or role) (AWS API)

Call one of the following operations:

- [PutGroupPolicy](#)
- [PutRolePolicy](#)

- [PutUserPolicy](#)

Note

You can embed an inline policy for a [service-linked role \(p. 176\)](#) only in the service that depends on the role. See the [AWS documentation](#) for your service to see whether it supports this feature.

Validating JSON Policies

Policy Validator automatically examines your new and existing IAM access control policies to ensure that they comply with the IAM policy grammar. A [policy](#) is a JSON document written using the [IAM policy grammar](#). It defines access permissions for the AWS user, group, or role that you attach the policy to. If Policy Validator determines that a policy is not in compliance with the grammar, it prompts you to fix the policy. Policy Validator is only available if you have non-compliant policies.

You can use the following methods to access Policy Validator:

1. **Creating JSON policies** – When you create a new JSON policy, Policy Validator runs automatically when you choose **Review policy**. If the policy is not valid, you receive a notification and must fix the problem before you can continue.
2. **Editing JSON policies** – When you edit an existing JSON policy, Policy Validator runs automatically when you choose **Review policy**. If the policy is not valid, you receive a notification and must fix the problem before you can continue. Existing policies with errors that were set up before the introduction of Policy Validator will continue to function. However, you cannot edit and save them without fixing the policy syntax errors.

Note

Policy Validator only checks JSON policy syntax and grammar. It does not validate that your ARNs, action names, or condition keys are correct.

Testing IAM Policies with the IAM Policy Simulator

For more information about how and why to use IAM policies, see [Policies and Permissions \(p. 356\)](#).

You can access the IAM Policy Simulator Console at: <https://policysim.aws.amazon.com/>

Getting Started with the IAM Policy Simulator

With the IAM policy simulator, you can test and troubleshoot identity-based policies, IAM permissions boundaries, Organizations service control policies, and resource-based policies. Here are some common things you can do with the policy simulator:

- Test policies that are attached to IAM users, groups, or roles in your AWS account. If more than one policy is attached to the user, group, or role, you can test all the policies, or select individual policies to test. You can test which actions are allowed or denied by the selected policies for specific resources.
- Test and troubleshoot the effect of [permissions boundaries \(p. 370\)](#) on IAM entities. Note: you can only simulate one permissions boundary at a time.
- Test policies that are attached to AWS resources, such as Amazon S3 buckets, Amazon SQS queues, Amazon SNS topics, or Amazon S3 Glacier vaults.
- If your AWS account is a member of an organization in [AWS Organizations](#), then you can test the impact of service control policies (SCPs) on your IAM policies and resource policies.
- Test new policies that are not yet attached to a user, group, or role by typing or copying them into the simulator. These are used only in the simulation and are not saved. Note: you cannot type or copy a resource-based policy into the simulator. To use a resource-based policy in the simulator, you must

include the resource in the simulation. You must also select the check box to include that resource's policy in the simulation.

- Test the policies with selected services, actions, and resources. For example, you can test to ensure that your policy allows an entity to perform the `ListAllMyBuckets`, `CreateBucket`, and `DeleteBucket` actions in the Amazon S3 service on a specific bucket.
- Simulate real-world scenarios by providing context keys, such as an IP address or date, that are included in `Condition` elements in the policies being tested.
- Identify which specific statement in a policy results in allowing or denying access to a particular resource or action.

Topics

- [How the IAM Policy Simulator Works \(p. 449\)](#)
- [Permissions Required for Using the IAM Policy Simulator \(p. 449\)](#)
- [Using the IAM Policy Simulator \(Console\) \(p. 452\)](#)
- [Using the IAM Policy Simulator \(AWS CLI and AWS API\) \(p. 456\)](#)

How the IAM Policy Simulator Works

The simulator evaluates the policies that you choose and determines the effective permissions for each of the actions that you specify. The simulator uses the same policy evaluation engine that is used during real requests to AWS services. But the simulator differs from the live AWS environment in the following ways:

- The simulator does not make an actual AWS service request, so you can safely test requests that might make unwanted changes to your live AWS environment.
- Because the simulator does not simulate running the selected actions, it cannot report any response to the simulated request. The only result returned is whether the requested action would be allowed or denied.
- If you edit a policy inside the simulator, these changes affect only the simulator. The corresponding policy in your AWS account remains unchanged.

Permissions Required for Using the IAM Policy Simulator

You can use the policy simulator console or the policy simulator API to test policies. By default, console users can test policies that are not yet attached to a user, group, or role by typing or copying those policies into the simulator. These policies are used only in the simulation and do not disclose sensitive information. API users must have permissions to test unattached policies. You can allow console or API users to test policies that are attached to IAM users, groups, or roles in your AWS account. To do so, you must provide permission to retrieve those policies. In order to test resource-based policies, users must have permission to retrieve the resource's policy.

For examples of console and API policies that allow a user to simulate policies, see the section called ["Example Policies: AWS Identity and Access Management \(IAM\)" \(p. 396\)](#).

Permissions Required for Using the Policy Simulator Console

You can allow users to test policies that are attached to IAM users, groups, or roles in your AWS account. To do so, you must provide your users with permissions to retrieve those policies. In order to test resource-based policies, users must have permission to retrieve the resource's policy.

To view an example policy that allows using the policy simulator console for policies that are attached to a user, group, or role, see [IAM: Access the Policy Simulator Console \(p. 419\)](#).

To view an example policy that allows using the policy simulator console only for those users with a specific path, see [IAM: Access the Policy Simulator Console Based on User Path \(p. 429\)](#).

To create a policy to allow using the policy simulator console for only one type of entity, use the following procedures.

To allow console users to simulate policies for users

Include the following actions in your policy:

- `iam:GetGroupPolicy`
- `iam:GetPolicy`
- `iam:GetPolicyVersion`
- `iam:GetUser`
- `iam:GetUserPolicy`
- `iam>ListAttachedUserPolicies`
- `iam>ListGroupsForUser`
- `iam>ListGroupPolicies`
- `iam>ListUserPolicies`
- `iam>ListUsers`

To allow console users to simulate policies for groups

Include the following actions in your policy:

- `iam:GetGroup`
- `iam:GetGroupPolicy`
- `iam:GetPolicy`
- `iam:GetPolicyVersion`
- `iam>ListAttachedGroupPolicies`
- `iam>ListGroupPolicies`
- `iam>ListGroups`

To allow console users to simulate policies for roles

Include the following actions in your policy:

- `iam:GetPolicy`
- `iam:GetPolicyVersion`
- `iam:GetRole`
- `iam:GetRolePolicy`
- `iam>ListAttachedRolePolicies`
- `iam>ListRolePolicies`
- `iam>ListRoles`

To test resource-based policies, users must have permission to retrieve the resource's policy.

To allow console users to test resource-based policies in an Amazon S3 bucket

Include the following action in your policy:

- `s3:GetBucketPolicy`

For example, the following policy uses this action to allow console users to simulate a resource-based policy in a specific Amazon S3 bucket.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": "s3:GetBucketPolicy",  
            "Resource": "arn:aws:s3:::bucket-name/*"  
        }  
    ]  
}
```

To allow console users to test policies for AWS Organizations

Include the following actions in your policy:

- `organizations:DescribePolicy`
- `organizations>ListPolicies`
- `organizations>ListPoliciesForTarget`
- `organizations>ListTargetsForPolicy`

Permissions Required for Using the Policy Simulator API

The policy simulator API operations [GetContextKeyForCustomPolicy](#) and [SimulateCustomPolicy](#) allow you to test policies that are not yet attached to a user, group, or role. To test such policies, you pass the policies as strings to the API. These policies are used only in the simulation and do not disclose sensitive information. You can also use the API to test policies that are attached to IAM users, groups, or roles in your AWS account. To do that, you must provide users with permissions to call [GetContextKeyForPrincipalPolicy](#) and [SimulatePrincipalPolicy](#).

To view an example policy that allows using the policy simulator API for attached and unattached policies in the current AWS account, see [IAM: Access the Policy Simulator API \(p. 419\)](#).

To create a policy to allow using the policy simulator API for only one type of policy, use the following procedures.

To allow API users to simulate policies passed directly to the API as strings

Include the following actions in your policy:

- `iam:GetContextKeysForCustomPolicy`
- `iam:SimulateCustomPolicy`

To allow API users to simulate policies attached to IAM users, groups, roles, or resources

Include the following actions in your policy:

- `iam:GetContextKeysForPrincipalPolicy`
- `iam:SimulatePrincipalPolicy`

For example, to give a user named Bob permission to simulate a policy that is assigned to a user named Alice, give Bob access to the following resource: `arn:aws:iam::777788889999:user/alice`.

To view an example policy that allows using the policy simulator API only for those users with a specific path, see [IAM: Access the Policy Simulator API Based on User Path \(p. 429\)](#).

Using the IAM Policy Simulator (Console)

By default, users can test policies that are not yet attached to a user, group, or role by typing or copying those policies into the policy simulator console. These policies are used only in the simulation and do not disclose sensitive information.

To test a policy that is not attached to a user, group, or role (console)

1. Open the IAM policy simulator console at: <https://policysim.aws.amazon.com/>.
2. In the **Mode**: menu at the top of the page, choose **New Policy**.
3. In the **Policy Sandbox**, choose **Create New Policy**.
4. Type or copy a policy into the simulator, and use the simulator as described in the following steps.

After you have permission to use the IAM Policy Simulator Console, you can use the simulator to test an IAM user, group, role, or resource policy.

To test a policy that is attached to a user, group, or role (console)

1. Open the IAM policy simulator console at <https://policysim.aws.amazon.com/>.

Note

To sign in to the policy simulator as an IAM user, use your unique sign-in URL to sign in to the AWS Management Console. Then go to <https://policysim.aws.amazon.com/>. For more information about signing in as an IAM user, see [How IAM Users Sign In to AWS \(p. 90\)](#).

The simulator opens in **Existing Policies** mode and lists the IAM users in your account under **Users, Groups, and Roles**.

2. Choose the option that is appropriate to your task:

To test this:	Do this:
A policy attached to a user	Choose Users in the Users, Groups, and Roles list. Then choose the user.
A policy attached to a group	Choose Groups in the Users, Groups, and Roles list. Then choose the group.
A policy attached to a role	Choose Roles in the Users, Groups, and Roles list. Then choose the role.
A policy attached to a resource	See Step 9 .
A custom policy for a user, group, or role	Choose Create New Policy . In the new Policies pane, type or paste a policy and then choose Apply .

Tip

To test a policy that is attached to group, you can launch the IAM policy simulator directly from the [IAM console](#): In the navigation pane, choose **Groups**. Choose the name of the

group that you want to test a policy on, and then choose the **Permissions** tab. In the **Inline Policies or Managed Policies** section, locate the policy that you want to test. In the **Actions** column for that policy, choose **Simulate Policy**.

To test a customer managed policy that is attached to a user: In the navigation pane, choose **Users**. Choose the name of the user that you want to test a policy on. Then choose the **Permissions** tab and expand the policy that you want to test. On the far right, choose **Simulate policy**. The **IAM Policy Simulator** opens in a new window and displays the selected policy in the **Policies** pane.

3. (Optional) If your account is a member of an organization in [AWS Organizations](#), then any service control policies (SCPs) that affect the simulated user's account appear in the **Policies** pane. The **Policies** pane also shows **IAM policies**, **Resource policies** and **Permissions Boundary policies**. SCPs are JSON policies that specify the maximum permissions for an organization or organizational unit (OU). The SCP limits permissions for entities in member accounts. If an SCP blocks a service or action, then no entity in that account can access that service nor perform that action. This is true even if an administrator explicitly grants permissions to that service or action through an IAM or resource policy. To remove an SCP from the simulation, clear the check box next to the SCP name. To view the SCP contents, choose the name of the SCP.

If your account is not a member of an organization, then there are no SCPs to simulate.

4. (Optional) You can test a policy that is set as a [permissions boundary \(p. 370\)](#) for an IAM entity (user or role), but not for groups. If a permissions boundary policy is currently set for the entity, it appears in the **Policies** pane. You can set only one permissions boundary for an entity. To test a different permissions boundary, you can create a custom permissions boundary. To do this, choose **Create New Policy**. A new **Policies** pane opens. In the menu, choose **Custom IAM Permissions Boundary Policy**. Enter a name for the new policy and type or copy a policy into the space below. Choose **Apply** to save the policy. Next, choose **Back** to return to the original **Policies** pane. Then select the check box next to the permissions boundary you want to use for the simulation.
5. (Optional) You can test only a subset of policies attached to a user, group, or role. To do so, in the **Policies** pane clear the check box next to each policy that you want to exclude.
6. Under **Policy Simulator**, choose **Select service** and then choose the service to test. Then choose **Select actions** and select one or more actions to test. Although the menus show the available selections for only one service at a time, all the services and actions that you have selected appear in **Action Settings and Results**.
7. (Optional) If any of the policies that you choose in [Step 2](#) and [Step 5](#) include conditions with [AWS global condition keys \(p. 665\)](#), then supply values for those keys. You can do this by expanding the **Global Settings** section and typing values for the key names displayed there.

Warning

If you leave the value for a condition key empty, then that key is ignored during the simulation. In some cases, this results in an error, and the simulation fails to run. In other cases, the simulation runs, but the results might not be reliable. In those cases, the simulation does not match the real-world conditions that include a value for the condition key or variable.

8. (Optional) Each selected action appears in the **Action Settings and Results** list with **Not simulated** shown in the **Permission** column until you actually run the simulation. Before you run the simulation, you can configure each action with a resource. To configure individual actions for a specific scenario, choose the arrow to expand the action's row. If the action supports resource-level permissions, you can type the [Amazon Resource Name \(ARN\)](#) of the specific resource whose access you want to test. By default, each resource is set to a wildcard (*). You can also specify a value for any [condition context keys \(p. 686\)](#). As noted previously, keys with empty values are ignored, which can cause simulation failures or unreliable results.
 - a. Choose the arrow next to the action name to expand each row and configure any additional information required to accurately simulate the action in your scenario. If the action requires any resource-level permissions, you can type the [Amazon Resource Name \(ARN\)](#) of the specific resource that you want to simulate access to. By default, each resource is set to a wildcard (*).

- b. If the action supports resource-level permissions but does not require them, then you can choose **Add Resource** to select the resource type that you want to add to the simulation.
- c. If any of the selected policies include a Condition element that references a context key for this action's service, then that key name is displayed under the action. You can specify the value to be used during the simulation of that action for the specified resource.

Actions that require different groups of resource types

Some actions require different resource types under different circumstances. Each group of resource types is associated with a scenario. If one of these applies to your simulation, select it and the simulator requires the resource types appropriate for that scenario. The following list shows each of the supported scenario options and the resources that you must define to run the simulation.

Each of the following Amazon EC2 scenarios requires that you specify instance, image, and security-group resources. If your scenario includes an EBS volume, then you must specify that volume as a resource. If the Amazon EC2 scenario includes a virtual private cloud (VPC), then you must supply the network-interface resource. If it includes an IP subnet, then you must specify the subnet resource. For more information on the Amazon EC2 scenario options, see [Supported Platforms](#) in the *Amazon EC2 User Guide*.

- **EC2-Classic-InstanceStore**

instance, image, security-group

- **EC2-Classic-EBS**

instance, image, security-group, volume

- **EC2-VPC-InstanceStore**

instance, image, security-group, network-interface

- **EC2-VPC-InstanceStore-Subnet**

instance, image, security-group, network-interface, subnet

- **EC2-VPC-EBS**

instance, image, security-group, network-interface, volume

- **EC2-VPC-EBS-Subnet**

instance, image, security-group, network-interface, subnet, volume

9. (Optional) If you want to include a resource-based policy in your simulation, then you must first select the actions that you want to simulate on that resource in [Step 6](#). Expand the rows for the selected actions, and type the ARN of the resource with a policy that you want to simulate. Then select **Include Resource Policy** next to the **ARN** text box. The IAM policy simulator currently supports resource-based policies from only the following services: Amazon S3 (resource-based policies only; ACLs are not currently supported), Amazon SQS, Amazon SNS, and unlocked S3 Glacier vaults (locked vaults are not currently supported).

10. Choose **Run Simulation** in the upper-right corner.

The **Permission** column in each row of **Action Settings and Results** displays the result of the simulation of that action on the specified resource.

11. Choose **Run Simulation** in the upper-right corner.

The **Permission** column in each row of **Action Settings and Results** displays the result of the simulation of that action on the specified resource.

12. To see which statement in a policy explicitly allowed or denied an action, choose the **N matching statement(s)** link in the **Permissions** column to expand the row. Then choose the **Show statement**

link. The **Policies** pane shows the relevant policy with the statement that affected the simulation result highlighted.

Note

If an action is *implicitly* denied—that is, if the action is denied only because it is not explicitly allowed—the **List** and **Show statement** options are not displayed.

Troubleshooting IAM Policy Simulator Console Messages

The following table lists the informational and warning messages you might encounter when using the IAM policy simulator. The table also provides steps you can take to resolve them.

Message	Steps to resolve
This policy has been edited. Changes will not be saved to your account.	<p>No action required.</p> <p>This message is informational. If you edit an existing policy in the IAM policy simulator, your change does not affect your AWS account. The simulator allows you to make changes to policies for testing purposes only.</p>
Cannot get the resource policy. Reason: <i>detailed error message</i>	<p>The simulator is not able to access a requested resource-based policy. Ensure that the specified resource ARN is correct and that the user running the simulation has permission to read the resource's policy.</p>
One or more policies require values in the simulation settings. The simulation might fail without these values.	<p>This message appears if the policy you are testing contains condition keys or variables but you have not provided any values for these keys or variables in Simulation Settings.</p> <p>To dismiss this message, choose Simulation Settings, Then enter a value for each condition key or variable.</p>
You have changed policies. These results are no longer valid.	<p>This message appears if you have changed the selected policy while results are displayed in the Results pane. Results shown in the Results pane are not updated dynamically.</p> <p>To dismiss this message, choose Run Simulation again to display new simulation results based on the changes made in the Policies pane.</p>
The resource you typed for this simulation does not match this service.	<p>This message appears if you have typed an Amazon Resource Name (ARN) in the Simulation Settings pane that does not match the service that you chose for the current simulation. For example, this message appears if you specify an ARN for an Amazon DynamoDB resource but you chose Amazon Redshift as the service to simulate.</p> <p>To dismiss this message, do one of the following:</p> <ul style="list-style-type: none"> Remove the ARN from the box in the Simulation Settings pane.

Message	Steps to resolve
This action belongs to a service that supports special access control mechanisms in addition to resource-based policies, such as Amazon S3 ACLs or S3 Glacier vault lock policies. The policy simulator does not support these mechanisms, so the results can differ from your production environment.	<ul style="list-style-type: none"> Choose the service that matches the ARN that you specified in Simulation Settings. <p>No action required.</p> <p>This message is informational. In the current version, the simulator evaluates policies attached to users and groups, and can evaluate resource-based policies for Amazon S3, Amazon SQS, Amazon SNS, and S3 Glacier. The policy simulator does not support all access control mechanisms supported by other AWS services.</p>
DynamoDB FGAC is currently not supported.	<p>No action required.</p> <p>This informational message refers to <i>fine-grained access control</i>. Fine-grained access control is the ability to use IAM policy conditions to determine who can access individual data items and attributes in DynamoDB tables and indexes. It also refers to the actions that can be performed on these tables and indexes. The current version of the IAM policy simulator does not support this type of policy condition. For more information on DynamoDB fine-grained access control, see Fine-Grained Access Control for DynamoDB.</p>
You have policies that do not comply with the policy syntax. You can use the Policy Validator to review and accept the recommended updates to your policies.	This message appears at the top of the policy list if you have policies that do not comply with the IAM policy grammar. In order to simulate these policies, follow the instructions at Validating JSON Policies (p. 448) to identify and fix these policies.
This policy must be updated to comply with the latest policy syntax rules.	This message is displayed if you have policies that do not comply with the IAM policy grammar. In order to simulate these policies, follow the instructions at Validating JSON Policies (p. 448) to identify and fix these policies.

Using the IAM Policy Simulator (AWS CLI and AWS API)

Policy simulator commands typically require calling API operations to do two things:

- Evaluate the policies and return the list of context keys that they reference. You need to know what context keys are referenced so that you can supply values for them in the next step.
- Simulate the policies, providing a list of actions, resources, and context keys that are used during the simulation.

For security reasons, the API operations have been broken into two groups:

- API operations that simulate only policies that are passed directly to the API as strings. This set includes [GetContextKeysForCustomPolicy](#) and [SimulateCustomPolicy](#).
- API operations that simulate the policies that are attached to a specified IAM user, group, role, or resource. Because these API operations can reveal details of permissions assigned to other

IAM entities, you should consider restricting access to these API operations. This set includes [GetContextKeysForPrincipalPolicy](#) and [SimulatePrincipalPolicy](#). For more information about restricting access to API operations, see [Example Policies: AWS Identity and Access Management \(IAM\) \(p. 396\)](#).

In both cases, the API operations simulate the effect of one or more policies on a list of actions and resources. Each action is paired with each resource and the simulation determines whether the policies allow or deny that action for that resource. You can also provide values for any context keys that your policies reference. You can get the list of context keys that the policies reference by first calling [GetContextKeysForCustomPolicy](#) or [GetContextKeysForPrincipalPolicy](#). If you don't provide a value for a context key, the simulation still runs. But the results might not be reliable because the simulator cannot include that context key in the evaluation.

To get the list of context keys (AWS CLI, AWS API)

Use the following to evaluate a list of policies and return a list of context keys that are used in the policies.

- AWS CLI: `aws iam get-context-keys-for-custom-policy` and `aws iam get-context-keys-for-principal-policy`
- AWS API: [GetContextKeysForCustomPolicy](#) and [GetContextKeysForPrincipalPolicy](#)

To simulate IAM policies (AWS CLI, AWS API)

Use the following to simulate IAM policies to determine a user's effective permissions.

- AWS CLI: `aws iam simulate-custom-policy` and `aws iam simulate-principal-policy`
- AWS API: [SimulateCustomPolicy](#) and [SimulatePrincipalPolicy](#)

Adding and Removing IAM Identity Permissions

You use policies to define the permissions for an identity (user, group, or role). You can add and remove permissions by attaching and detaching IAM policies for an identity using the AWS Management Console, the AWS Command Line Interface (AWS CLI), or the AWS API. You can also use policies to set [permissions boundaries \(p. 370\)](#) for only entities (users or roles) using the same methods. Permissions boundaries are an advanced AWS feature that control the maximum permissions that an entity can have.

Topics

- [Terminology \(p. 457\)](#)
- [View Identity Activity \(p. 458\)](#)
- [Adding IAM Identity Permissions \(Console\) \(p. 458\)](#)
- [Removing IAM Identity Permissions \(Console\) \(p. 460\)](#)
- [Adding IAM Policies \(AWS CLI\) \(p. 461\)](#)
- [Removing IAM Policies \(AWS CLI\) \(p. 462\)](#)
- [Adding IAM Policies \(AWS API\) \(p. 463\)](#)
- [Removing IAM Policies \(AWS API\) \(p. 464\)](#)

Terminology

When you associate permissions policies with identities (users, groups, and roles), terminology and procedures vary depending on whether you are working with a managed or inline policy:

- **Attach** – Used with managed policies. You attach a managed policy to an identity (a user, group, or role). Attaching a policy applies the permissions in the policy to the identity.
- **Detach** – Used with managed policies. You detach a managed policy from an IAM identity (a user, group, or role). Detaching a policy removes its permissions from the identity.
- **Embed** – Used with inline policies. You embed an inline policy in an identity (a user, group, or role). Embedding a policy applies the permissions in the policy to the identity. Because an inline policy is stored in the identity, it is embedded rather than attached, though the results are similar.

Note

You can embed an inline policy for a [service-linked role \(p. 176\)](#) only in the service that depends on the role. See the [AWS documentation](#) for your service to see whether it supports this feature.

- **Delete** – Used with inline policies. You delete an inline policy from an IAM identity (a user, group, or role). Deleting a policy removes its permissions from the identity.

Note

You can delete an inline policy for a [service-linked role \(p. 176\)](#) only in the service that depends on the role. See the [AWS documentation](#) for your service to see whether it supports this feature.

You can use the console, AWS CLI, or AWS API to perform any of these actions.

More Information

- For more information about the difference between managed and inline policies, see [Managed Policies and Inline Policies \(p. 364\)](#).
- For more information about permissions boundaries, see [Permissions Boundaries for IAM Entities \(p. 370\)](#).
- For general information about IAM policies, see [Policies and Permissions \(p. 356\)](#).
- For information about policy size limitations, see [IAM and STS Limits \(p. 581\)](#).

View Identity Activity

Before you change the permissions for an identity (user, group, or role), you should review their recent service-level activity. This is important because you don't want to remove access from a principal (person or application) who is using it. For more information about viewing service last accessed data, see [Refining Permissions Using Service Last Accessed Data \(p. 475\)](#).

Adding IAM Identity Permissions (Console)

You can use the AWS Management Console to add permissions to an identity (user, group, or role). To do this, attach managed policies that control permissions, or specify a policy that serves as a [permissions boundary \(p. 370\)](#). You can also embed an inline policy.

To use a managed policy as a permissions policy for an identity (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Policies**.
3. In the list of policies, select the check box next to the name of the policy to attach. You can use the **Filter** menu and the search box to filter the list of policies.
4. Choose **Policy actions**, and then choose **Attach**.

5. Select one or more identities to attach the policy to. You can use the **Filter** menu and the search box to filter the list of principal entities. After selecting the identities, choose **Attach policy**.

To use a managed policy to set a permissions boundary (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Policies**.
3. In the list of policies, choose the name of the policy to set. You can use the **Filter** menu and the search box to filter the list of policies.
4. On the policy summary page, choose the **Policy usage tab**, and then, if necessary, open the **Permissions boundaries** section and choose **Set boundary**.
5. Select one or more users or roles on which to use the policy for a permissions boundary. You can use the **Filter** menu and the search box to filter the list of principal entities. After selecting the principals, choose **Set boundaries**.

To embed an inline policy for a user or role (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Users or Roles**.
3. In the list, choose the name of the user or role to embed a policy in.
4. Choose the **Permissions** tab.
5. Scroll to the bottom of the page and choose **Add inline policy**.

Note

You cannot embed an inline policy in a [service-linked role \(p. 176\)](#) in IAM. Because the linked service defines whether you can modify the permissions of the role, you might be able to add additional policies from the service console, API, or AWS CLI. To view the service-linked role documentation for a service, see [AWS Services That Work with IAM \(p. 586\)](#) and choose **Yes** in the **Service-Linked Role** column for your service.

6. Choose from the following methods to view the steps required to create your policy:
 - [Importing Existing Managed Policies \(p. 444\)](#) – You can import a managed policy within your account and then edit the policy to customize it to your specific requirements. A managed policy can be an AWS managed policy or a customer managed policy that you created previously.
 - [Creating Policies with the Visual Editor \(p. 445\)](#) – You can construct a new policy from scratch in the visual editor. If you use the visual editor, you do not have to understand JSON syntax.
 - [Creating Policies on the JSON Tab \(p. 446\)](#) – In the **JSON** tab, you can use JSON syntax to create a policy. You can type a new JSON policy document or paste an [example policy \(p. 394\)](#).
7. After you create an inline policy, it is automatically embedded in your user or role.

To embed an inline policy for a group (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Groups**.
3. In the list, choose the name of the group to embed a policy in.
4. Choose the **Permissions** tab and expand the **Inline Policies** section if necessary.

5. Choose **Create Group Policy**. If there are no existing policies in **Groups**, instead choose [click here](#) to create your first inline policy.
6. Choose **Policy Generator** or **Custom Policy**, and then choose **Select**.
7. Do one of the following:
 - If you chose **Custom Policy**, specify a name for the policy and create your policy document. [Policy Validator \(p. 448\)](#) reports any syntax errors.
 - If you are using the policy generator to create your policy, choose the appropriate **Effect**, **AWS Service**, and **Actions** options. Type the Amazon Resource Name (ARN) (if applicable), and add any conditions that you want to include. Then choose **Add Statement**. You can add as many statements as you want to the policy. When you are finished adding statements, choose **Next Step**.
8. When you are satisfied with the policy, choose **Apply Policy**.

To change the permissions boundary for one or more entities (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Policies**.
3. In the list of policies, choose the name of the policy to set. You can use the **Filter** menu and the search box to filter the list of policies.
4. On the policy summary page, choose the **Policy usage tab**, and then, if necessary, open the **Permissions boundaries** section. Select the check box next to the users or roles whose boundaries you want to change and then choose **Change boundary**.
5. Select a new policy to use for a permissions boundary. You can use the **Filter** menu and the search box to filter the list of policies. After selecting the policy, choose **Change boundary**.

Removing IAM Identity Permissions (Console)

You can use the AWS Management Console to remove permissions from an identity (user, group, or role). To do this, detach managed policies that control permissions, or remove a policy that serves as a [permissions boundary \(p. 370\)](#). You can also delete an inline policy.

To detach a managed policy used as a permissions policy (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Policies**.
3. In the list of policies, select the check box next to the name of the policy to detach. You can use the **Filter** menu and the search box to filter the list of policies.
4. Choose **Policy actions**, and then choose **Detach**.
5. Select the identities to detach the policy from. You can use the **Filter** menu and the search box to filter the list of identities. After selecting the identities, choose **Detach policy**.

To remove a permissions boundary (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Policies**.

3. In the list of policies, choose the name of the policy to set. You can use the **Filter** menu and the search box to filter the list of policies.
4. On the policy summary page, choose the **Policy usage tab**, and then, if necessary, open the **Permissions boundaries** section and choose **Remove boundary**.
5. Confirm that you want to remove the boundary and choose **Remove**.

To delete an inline policy (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Groups, Users, or Roles**.
3. In the list, choose the name of the group, user, or role that has the policy you want to remove.
4. Choose the **Permissions** tab. If you chose **Groups**, expand the **Inline Policies** section if necessary.
5. If in **Groups**, choose **Remove Policy**. If in **Users or Roles**, choose **X**.

Adding IAM Policies (AWS CLI)

You can use the AWS CLI to add permissions to an identity (user, group, or role). To do this, attach managed policies that control permissions, or specify a policy that serves as a [permissions boundary \(p. 370\)](#). You can also embed an inline policy.

To use a managed policy as a permissions policy for an entity (AWS CLI)

1. (Optional) To view information about a managed policy, run the following commands:
 - To list managed policies: [aws iam list-policies](#)
 - To retrieve detailed information about a managed policy: [get-policy](#)
2. To attach a managed policy to an identity (user, group, or role), use one of the following commands:
 - [aws iam attach-user-policy](#)
 - [aws iam attach-group-policy](#)
 - [aws iam attach-role-policy](#)

To use a managed policy to set a permissions boundary (AWS CLI)

1. (Optional) To view information about a managed policy, run the following commands:
 - To list managed policies: [aws iam list-policies](#)
 - To retrieve detailed information about a managed policy: [aws iam get-policy](#)
2. To use a managed policy to set the permissions boundary for an entity (user or role), use one of the following commands:
 - [aws iam put-user-permissions-boundary](#)
 - [aws iam put-role-permissions-boundary](#)

To embed an inline policy (AWS CLI)

To embed an inline policy to an identity (user, group, or role that is not a [service-linked role \(p. 176\)](#)), use one of the following commands:

- [aws iam put-user-policy](#)

- [aws iam put-group-policy](#)
- [aws iam put-role-policy](#)

Removing IAM Policies (AWS CLI)

You can use the AWS CLI to detach managed policies that control permissions, or remove a policy that serves as a [permissions boundary \(p. 370\)](#). You can also delete an inline policy.

To detach a managed policy used as a permissions policy (AWS CLI)

1. (Optional) To view information about a policy, run the following commands:
 - To list managed policies: [aws iam list-policies](#)
 - To retrieve detailed information about a managed policy: [aws iam get-policy](#)
2. (Optional) To find out about the relationships between the policies and identities, run the following commands:
 - To list the identities (users, groups, and roles) to which a managed policy is attached:
 - [aws iam list-entities-for-policy](#)
 - To list the managed policies attached to an identity (a user, group, or role), use one of the following commands:
 - [aws iam list-attached-user-policies](#)
 - [aws iam list-attached-group-policies](#)
 - [aws iam list-attached-role-policies](#)
3. To detach a managed policy from an identity (user, group, or role), use one of the following commands:
 - [aws iam detach-user-policy](#)
 - [aws iam detach-group-policy](#)
 - [aws iam detach-role-policy](#)

To remove a permissions boundary (AWS CLI)

1. (Optional) To view which managed policy is currently used to set the permissions boundary for a user or role, run the following commands:
 - [aws iam get-user](#)
 - [aws iam get-role](#)
2. (Optional) To view the users or roles on which a managed policy is used for a permissions boundary, run the following command:
 - [aws iam list-entities-for-policy](#)
3. (Optional) To view information about a managed policy, run the following commands:
 - To list managed policies: [aws iam list-policies](#)
 - To retrieve detailed information about a managed policy: [aws iam get-policy](#)
4. To remove a permissions boundary from a user or role, use one of the following commands:
 - [aws iam delete-user-permissions-boundary](#)
 - [aws iam delete-role-permissions-boundary](#)

To delete an inline policy (AWS CLI)

1. (Optional) To list all inline policies that are attached to an identity (user, group, role), use one of the following commands:
 - [aws iam list-user-policies](#)
 - [aws iam list-group-policies](#)
 - [aws iam list-role-policies](#)
2. (Optional) To retrieve an inline policy document that is embedded in an identity (user, group, or role), use one of the following commands:
 - [aws iam get-user-policy](#)
 - [aws iam get-group-policy](#)
 - [aws iam get-role-policy](#)
3. To delete an inline policy from an identity (user, group, or role that is not a [service-linked role](#) (p. 176)), use one of the following commands:
 - [aws iam delete-user-policy](#)
 - [aws iam delete-group-policy](#)
 - [aws iam delete-role-policy](#)

Adding IAM Policies (AWS API)

You can use the AWS API to attach managed policies that control permissions or specify a policy that serves as a [permissions boundary](#) (p. 370). You can also embed an inline policy.

To use a managed policy as a permissions policy for an entity (AWS API)

1. (Optional) To view information about a policy, call the following operations:
 - To list managed policies: [ListPolicies](#)
 - To retrieve detailed information about a managed policy: [GetPolicy](#)
2. To attach a managed policy to an identity (user, group, or role), call one of the following operations:
 - [AttachUserPolicy](#)
 - [AttachGroupPolicy](#)
 - [AttachRolePolicy](#)

To use a managed policy to set a permissions boundary (AWS API)

1. (Optional) To view information about a managed policy, call the following operations:
 - To list managed policies: [ListPolicies](#)
 - To retrieve detailed information about a managed policy: [GetPolicy](#)
2. To use a managed policy to set the permissions boundary for an entity (user or role), call one of the following operations:
 - [PutUserPermissionsBoundary](#)
 - [PutRolePermissionsBoundary](#)

To embed an inline policy (AWS API)

To embed an inline policy in an identity (user, group, or role that is not a *service-linked role* (p. 176)), call one of the following operations:

- [PutUserPolicy](#)
- [PutGroupPolicy](#)
- [PutRolePolicy](#)

Removing IAM Policies (AWS API)

You can use the AWS API to detach managed policies that control permissions or remove a policy that serves as a [permissions boundary](#) (p. 370). You can also delete an inline policy.

To detach a managed policy used as a permissions policy (AWS API)

1. (Optional) To view information about a policy, call the following operations:
 - To list managed policies: [ListPolicies](#)
 - To retrieve detailed information about a managed policy: [GetPolicy](#)
2. (Optional) To find out about the relationships between the policies and identities, call the following operations:
 - To list the identities (users, groups, and roles) to which a managed policy is attached:
 - [ListEntitiesForPolicy](#)
 - To list the managed policies attached to an identity (a user, group, or role), call one of the following operations:
 - [ListAttachedUserPolicies](#)
 - [ListAttachedGroupPolicies](#)
 - [ListAttachedRolePolicies](#)
3. To detach a managed policy from an identity (user, group, or role), call one of the following operations:
 - [DetachUserPolicy](#)
 - [DetachGroupPolicy](#)
 - [DetachRolePolicy](#)

To remove a permissions boundary (AWS API)

1. (Optional) To view which managed policy is currently used to set the permissions boundary for a user or role, call the following operations:
 - [GetUser](#)
 - [GetRole](#)
2. (Optional) To view the users or roles on which a managed policy is used for a permissions boundary, call the following operation:
 - [ListEntitiesForPolicy](#)
3. (Optional) To view information about a managed policy, call the following operations:
 - To list managed policies: [ListPolicies](#)
 - To retrieve detailed information about a managed policy: [GetPolicy](#)
4. To remove a permissions boundary from a user or role, call one of the following operations:
 - [DeleteUserPermissionsBoundary](#)

- [DeleteRolePermissionsBoundary](#)

To delete an inline policy (AWS API)

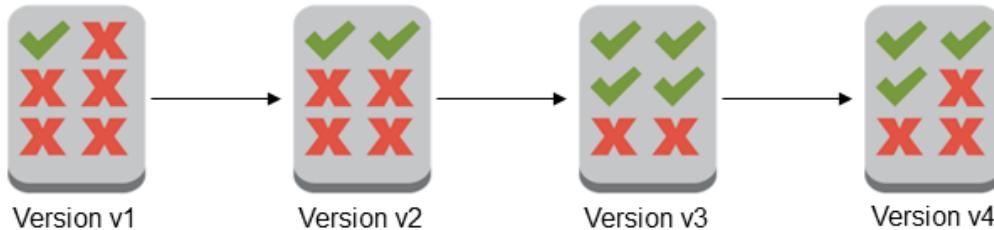
1. (Optional) To list all inline policies that are attached to an identity (user, group, role), call one of the following operations:
 - [ListUserPolicies](#)
 - [ListGroupPolicies](#)
 - [ListRolePolicies](#)
2. (Optional) To retrieve an inline policy document that is embedded in an identity (user, group, or role), call one of the following operations:
 - [GetUserPolicy](#)
 - [GetGroupPolicy](#)
 - [GetRolePolicy](#)
3. To delete an inline policy from an identity (user, group, or role that is not a [service-linked role \(p. 176\)](#)), call one of the following operations:
 - [DeleteUserPolicy](#)
 - [DeleteGroupPolicy](#)
 - [DeleteRolePolicy](#)

Versioning IAM Policies

When you make changes to an IAM customer managed policy, and when AWS makes changes to an AWS managed policy, the changed policy doesn't overwrite the existing policy. Instead, IAM creates a new *version* of the managed policy. IAM stores up to five versions of your customer managed policies. IAM does not support versioning for inline policies.

The following diagram illustrates versioning for a customer managed policy.

Multiple versions of a single managed policy



A policy version is different from a `Version` policy element. The `Version` policy element is used within a policy and defines the version of the policy language. To learn more about the `Version` policy element see [IAM JSON Policy Elements: Version \(p. 598\)](#).

You can use versions to track changes to a managed policy. For example, you might make a change to a managed policy and then discover that the change had unintended effects. In this case, you can roll back to a previous version of the managed policy by setting the previous version as the *default* version.

The following sections explain how you can use versioning for managed policies.

Topics

- [Permissions for Setting the Default Version of a Policy \(p. 466\)](#)
- [Setting the Default Version of Customer Managed Policies \(p. 466\)](#)
- [Using Versions to Roll Back Changes \(p. 468\)](#)
- [Version Limits \(p. 468\)](#)

Permissions for Setting the Default Version of a Policy

The permissions that are required to set the default version of a policy correspond to the AWS API operations for the task. You can use the `CreatePolicyVersion` or `SetDefaultPolicyVersion` API operations to set the default version of a policy. To allow someone to set the default policy version of an existing policy, you can allow access to either the `iam:CreatePolicyVersion` action or the `iam:SetDefaultPolicyVersion` action. The `iam:CreatePolicyVersion` action allows them to create a new version of the policy and to set that version as the default. The `iam:SetDefaultPolicyVersion` action allows them to set any existing version of the policy as the default.

Important

Denying the `iam:SetDefaultPolicyVersion` action in a user's policy does not stop the user from creating a new policy version and setting it as the default.

You can use the following policy to deny a user access to change an existing customer managed policy:

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Deny",  
            "Action": [  
                "iam:CreatePolicyVersion",  
                "iam:SetDefaultPolicyVersion"  
            ],  
            "Resource": "arn:aws:iam::*:policy/POLICY-NAME"  
        }  
    ]  
}
```

Setting the Default Version of Customer Managed Policies

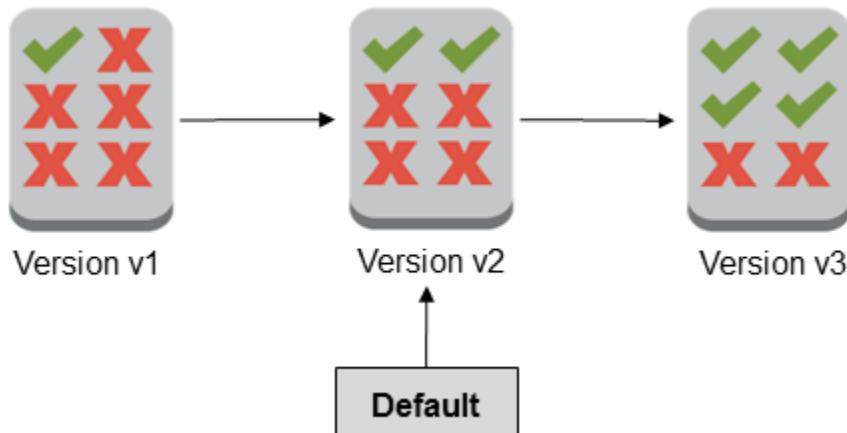
One of the versions of a managed policy is set as the *default* version. The policy's default version is the operative version—that is, it's the version that is in effect for all of the principal entities (users, groups, and roles) that the managed policy is attached to.

When you create a customer managed policy, the policy begins with a single version identified as v1. For managed policies with only a single version, that version is automatically set as the default. For customer managed policies with more than one version, you choose which version to set as the default. For AWS managed policies, the default version is set by AWS. The following diagrams illustrate this concept.

Managed policy with one version



Managed policy with multiple versions



You can set the default version of a customer managed policy to apply that version to every IAM identity (user, group, and role) where the policy is attached. You cannot set the default version for an AWS managed policy or an inline policy.

To set the default version of a customer managed policy (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Policies**.
3. In the list of policies, choose the policy name of the policy to set the default version of. You can use the **Filter** menu and the search box to filter the list of policies.
4. Choose the **Policy versions** tab. Select the check box next to the version that you want to set as the default version, and then choose **Set as default**.

To learn how to set the default version of a customer managed policy from the AWS Command Line Interface or the AWS API, see [Editing Customer Managed Policies \(AWS CLI\) \(p. 471\)](#).

Using Versions to Roll Back Changes

You can set the default version of a customer managed policy to roll back your changes. For example, consider the following scenario:

You create a customer managed policy that allows users to administer a particular Amazon S3 bucket using the AWS Management Console. Upon creation, your customer managed policy has only one version, identified as v1, so that version is automatically set as the default. The policy works as intended.

Later, you update the policy to add permission to administer a second Amazon S3 bucket. IAM creates a new version of the policy, identified as v2, that contains your changes. You set version v2 as the default, and a short time later your users report that they lack permission to use the Amazon S3 console. In this case, you can roll back to version v1 of the policy, which you know works as intended. To do this, you set version v1 as the default version. Your users are now able to use the Amazon S3 console to administer the original bucket.

Later, after you determine the error in version v2 of the policy, you update the policy again to add permission to administer the second Amazon S3 bucket. IAM creates another new version of the policy, identified as v3. You set version v3 as the default, and this version works as intended. At this point, you delete version v2 of the policy.

Version Limits

A managed policy can have up to five versions. If you need to make changes to a managed policy beyond five versions from the AWS Command Line Interface, or the AWS API, you must first delete one or more existing versions. If you use the AWS Management Console, you do not have to delete a version before editing your policy. When you save a sixth version, a dialog box appears that prompts you to delete one or more nondefault versions of your policy. You can view the JSON policy document for each version to help you decide. For details about this dialog box, see [the section called "Editing IAM Policies" \(p. 468\)](#).

You can delete any version of the managed policy that you want, except for the default version. When you delete a version, the version identifiers for the remaining versions do not change. As a result, version identifiers might not be sequential. For example, if you delete versions v2 and v4 of a managed policy and add two new versions, the remaining version identifiers might be v1, v3, v5, v6, and v7.

Editing IAM Policies

A [policy \(p. 356\)](#) is an entity that, when attached to an identity or resource, defines their permissions. Policies are stored in AWS as JSON documents and are attached to principals as *identity-based policies* in IAM. You can attach an identity-based policy to a principal (or identity), such as an IAM group, user, or role. Identity-based policies include AWS managed policies, customer managed policies, and [inline policies \(p. 364\)](#). You can edit customer managed policies and inline policies in IAM. AWS managed policies cannot be edited. For information about policy size limitations and other quotas, see [IAM and STS Limits \(p. 581\)](#).

Topics

- [View Policy Access \(p. 468\)](#)
- [Editing Customer Managed Policies \(Console\) \(p. 469\)](#)
- [Editing Inline Policies \(Console\) \(p. 470\)](#)
- [Editing Customer Managed Policies \(AWS CLI\) \(p. 471\)](#)
- [Editing Customer Managed Policies \(AWS API\) \(p. 471\)](#)

View Policy Access

Before you change the permissions for a policy, you should review its recent service-level activity. This is important because you don't want to remove access from a principal (person or application) who is using

it. For more information about viewing service last accessed data, see [Refining Permissions Using Service Last Accessed Data \(p. 475\)](#).

Editing Customer Managed Policies (Console)

You can edit customer managed policies to change the permissions that are defined in the policy. A customer managed policy can have up to five versions. This is important because if you make changes to a managed policy beyond five versions, the AWS Management Console prompts you to decide which version to delete. You can also change the default version or delete a version of a policy before you edit it to avoid being prompted. To learn more about versions, see [Versioning IAM Policies \(p. 465\)](#).

To edit a customer managed policy (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Policies**.
3. In the list of policies, choose the policy name of the policy to edit. You can use the **Filter** menu and the search box to filter the list of policies.
4. Choose the **Permissions** tab, and then choose **Edit policy**.
5. Do one of the following:
 - Choose the **Visual editor** tab to change your policy without understanding JSON syntax. You can make changes to the service, actions, resources, or optional conditions for each permission block in your policy. You can also import a policy to add additional permissions to the bottom of your policy. When you are finished making changes, choose **Review policy** to continue.
 - Choose the **JSON** tab to modify your policy by typing or pasting text in the JSON text box. You can also import a policy to add additional permissions to the bottom of your policy. When you are finished making changes, choose **Review policy** to continue. [Policy Validator \(p. 448\)](#) reports any syntax errors.

Note

You can switch between the **Visual editor** and **JSON** tabs any time. However, if you make changes or choose **Review policy** in the **Visual editor** tab, IAM might restructure your policy to optimize it for the visual editor. For more information, see [Policy Restructuring \(p. 548\)](#).

6. On the **Review** page, review the policy **Summary** and then choose **Save changes** to save your work.
7. If the managed policy already has the maximum of five versions, choosing **Save** displays a dialog box. To save your new version, you must remove at least one earlier version. You cannot delete the default version. Choose from the following options:
 - **Remove oldest non-default policy version (version v# - created # days ago)** – Use this option to see which version will be deleted and when it was created. You can view the JSON policy document for all nondefault versions by choosing the second option, **Select versions to remove**.
 - **Select versions to remove** – Use this option to view the JSON policy document and choose one or more versions to delete.

After choosing the versions to remove, choose **Delete version and save** to save your new policy version.

To set the default version of a customer managed policy (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Policies**.

3. In the list of policies, choose the policy name of the policy to set the default version of. You can use the **Filter** menu and the search box to filter the list of policies.
4. Choose the **Policy versions** tab. Select the check box next to the version that you want to set as the default version, and then choose **Set as default**.

To delete a version of a customer managed policy (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Policies**.
3. Choose the name of the customer managed policy that has a version you want to delete. You can use the **Filter** menu and the search box to filter the list of policies.
4. Choose the **Policy versions** tab. Select the check box next to the version that you want to delete. Then choose **Delete**.
5. Confirm that you want to delete the version, and then choose **Delete**.

Editing Inline Policies (Console)

You can edit an inline policy from the AWS Management Console.

To edit an inline policy for a user or role (console)

1. In the navigation pane, choose **Users or Roles**.
2. Choose the name of the user or role with the policy that you want to modify. Then choose the **Permissions** tab and expand the policy.
3. To edit an inline policy, choose **Edit Policy**.
4. Do one of the following:
 - Choose the **Visual editor** tab to change your policy without understanding JSON syntax. You can make changes to the service, actions, resources, or optional conditions for each permission block in your policy. You can also import a policy to add additional permissions to the bottom of your policy. When you are finished making changes, choose **Review policy** to continue.
 - Choose the **JSON** tab to modify your policy by typing or pasting text in the JSON text box. You can also import a policy to add additional permissions to the bottom of your policy. When you are finished making changes, choose **Review policy** to continue. [Policy Validator \(p. 448\)](#) reports any syntax errors. To save your changes without affecting the currently attached entities, clear the check box for **Save as default version**.

Note

You can switch between the **Visual editor** and **JSON** tabs any time. However, if you make changes or choose **Review policy** in the **Visual editor** tab, IAM might restructure your policy to optimize it for the visual editor. For more information, see [Policy Restructuring \(p. 548\)](#).

5. On the **Review** page, review the policy **Summary** and then choose **Save changes** to save your work.

To edit an inline policy for a group (console)

1. In the navigation pane, choose **Groups**.
2. Choose the name of the group with the policy that you want to modify. Then choose the **Permissions** tab.
3. To edit an inline policy, choose **Edit Policy**.

4. After you have modified your JSON policy, choose **Save** to save your changes.

Editing Customer Managed Policies (AWS CLI)

You can edit a customer managed policy from the AWS Command Line Interface (AWS CLI).

Note

A managed policy can have up to five versions. If you need to make changes to a customer managed policy beyond five versions, you must first delete one or more existing versions.

To edit a customer managed policy (AWS CLI)

1. (Optional) To view information about a policy, run the following commands:
 - To list managed policies: [list-policies](#)
 - To retrieve detailed information about a managed policy: [get-policy](#)
2. (Optional) To find out about the relationships between the policies and identities, run the following commands:
 - To list the identities (users, groups, and roles) to which a managed policy is attached:
 - [list-entities-for-policy](#)
 - To list the managed policies attached to an identity (a user, group, or role):
 - [list-attached-user-policies](#)
 - [list-attached-group-policies](#)
 - [list-attached-role-policies](#)
3. To edit a customer managed policy, run the following command:
 - [create-policy-version](#)

To set the default version of a customer managed policy (AWS CLI)

1. (Optional) To list managed policies, run the following command:
 - [list-policies](#)
2. To set the default version of a customer managed policy, run the following command:
 - [set-default-policy-version](#)

To delete a version of a customer managed policy (AWS CLI)

1. (Optional) To list managed policies, run the following command:
 - [list-policies](#)
2. To delete a customer managed policy, run the following command:
 - [delete-policy-version](#)

Editing Customer Managed Policies (AWS API)

You can edit a customer managed policy using the AWS API.

Note

A managed policy can have up to five versions. If you need to make changes to a customer managed policy beyond five versions, you must first delete one or more existing versions.

To edit a customer managed policy (AWS API)

1. (Optional) To view information about a policy, call the following operations:
 - To list managed policies: [ListPolicies](#)
 - To retrieve detailed information about a managed policy: [GetPolicy](#)
2. (Optional) To find out about the relationships between the policies and identities, call the following operations:
 - To list the identities (users, groups, and roles) to which a managed policy is attached:
 - [ListEntitiesForPolicy](#)
 - To list the managed policies attached to an identity (a user, group, or role):
 - [ListAttachedUserPolicies](#)
 - [ListAttachedGroupPolicies](#)
 - [ListAttachedRolePolicies](#)
3. To edit a customer managed policy, call the following operation:
 - [CreatePolicyVersion](#)

To set the default version of a customer managed policy (AWS API)

1. (Optional) To list managed policies, call the following operation:
 - [ListPolicies](#)
2. To set the default version of a customer managed policy, call the following operation:
 - [SetDefaultPolicyVersion](#)

To delete a version of a customer managed policy (AWS API)

1. (Optional) To list managed policies, call the following operation:
 - [ListPolicies](#)
2. To delete a customer managed policy, call the following operation:
 - [DeletePolicyVersion](#)

Deleting IAM Policies

You can delete IAM policies using the AWS Management Console, the AWS Command Line Interface (AWS CLI), or the IAM API.

For more information about the difference between managed and inline policies, see [Managed Policies and Inline Policies \(p. 364\)](#).

For general information about IAM policies, see [Policies and Permissions \(p. 356\)](#).

For information about policy size limitations and other quotas, see [IAM and STS Limits \(p. 581\)](#).

Topics

- [View Policy Access \(p. 473\)](#)
- [Deleting IAM Policies \(Console\) \(p. 473\)](#)
- [Deleting IAM Policies \(AWS CLI\) \(p. 473\)](#)

- [Deleting IAM Policies \(AWS API\) \(p. 474\)](#)

View Policy Access

Before you delete a policy, you should review its recent service-level activity. This is important because you don't want to remove access from a principal (person or application) who is using it. For more information about viewing service last accessed data, see [Refining Permissions Using Service Last Accessed Data \(p. 475\)](#).

Deleting IAM Policies (Console)

You can delete a customer managed policy to remove it from your AWS account. You cannot delete AWS managed policies.

To delete a customer managed policy (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Policies**.
3. Select the check box next to the customer managed policy to delete. You can use the **Filter** menu and the search box to filter the list of policies.
4. Choose **Policy actions**, and then choose **Delete**.
5. Confirm that you want to delete the policy, and then choose **Delete**.

To delete an inline policy for a group, user, or role (console)

1. In the navigation pane, choose **Groups, Users, or Roles**.
2. Choose the name of the group, user, or role with the policy that you want to delete. Then choose the **Permissions** tab. If you chose **Users** or **Roles**, expand the policy.
3. To delete an inline policy in **Groups**, choose **Remove Policy**. To delete an inline policy in **Users** or **Roles**, choose **X**.

Deleting IAM Policies (AWS CLI)

You can delete a customer managed policy from the AWS Command Line Interface.

To delete a customer managed policy (AWS CLI)

1. (Optional) To view information about a policy, run the following commands:
 - To list managed policies: `list-policies`
 - To retrieve detailed information about a managed policy: `get-policy`
2. (Optional) To find out about the relationships between the policies and identities, run the following commands:
 - To list the identities (users, groups, and roles) to which a managed policy is attached, run the following command:
 - `list-entities-for-policy`
 - To list the managed policies attached to an identity (a user, group, or role), run one of the following commands:
 - `list-attached-user-policies`

- [list-attached-group-policies](#)
 - [list-attached-role-policies](#)
3. To delete a customer managed policy, run the following command:
- [delete-policy](#)

To delete an inline policy (AWS CLI)

1. (Optional) To list all inline policies that are attached to an identity (user, group, role), use one of the following commands:
 - [aws iam list-user-policies](#)
 - [aws iam list-group-policies](#)
 - [aws iam list-role-policies](#)
2. (Optional) To retrieve an inline policy document that is embedded in an identity (user, group, or role), use one of the following commands:
 - [aws iam get-user-policy](#)
 - [aws iam get-group-policy](#)
 - [aws iam get-role-policy](#)
3. To delete an inline policy from an identity (user, group, or role that is not a [service-linked role \(p. 176\)](#)), use one of the following commands:
 - [aws iam delete-user-policy](#)
 - [aws iam delete-group-policy](#)
 - [aws iam delete-role-policy](#)

Deleting IAM Policies (AWS API)

You can delete a customer managed policy using the AWS API.

To delete a customer managed policy (AWS API)

1. (Optional) To view information about a policy, call the following operations:
 - To list managed policies: [ListPolicies](#)
 - To retrieve detailed information about a managed policy: [GetPolicy](#)
2. (Optional) To find out about the relationships between the policies and identities, call the following operations:
 - To list the identities (users, groups, and roles) to which a managed policy is attached, call the following operation:
 - [ListEntitiesForPolicy](#)
 - To list the managed policies attached to an identity (a user, group, or role), call one of the following operations:
 - [ListAttachedUserPolicies](#)
 - [ListAttachedGroupPolicies](#)
 - [ListAttachedRolePolicies](#)
3. To delete a customer managed policy, call the following operation:
 - [DeletePolicy](#)

To delete an inline policy (AWS API)

1. (Optional) To list all inline policies that are attached to an identity (user, group, role), call one of the following operations:
 - [ListUserPolicies](#)
 - [ListGroupPolicies](#)
 - [ListRolePolicies](#)
2. (Optional) To retrieve an inline policy document that is embedded in an identity (user, group, or role), call one of the following operations:
 - [GetUserPolicy](#)
 - [GetGroupPolicy](#)
 - [GetRolePolicy](#)
3. To delete an inline policy from an identity (user, group, or role that is not a *service-linked role* (p. 176)), call one of the following operations:
 - [DeleteUserPolicy](#)
 - [DeleteGroupPolicy](#)
 - [DeleteRolePolicy](#)

Refining Permissions Using Service Last Accessed Data

You can view service last accessed data for entities or policies in IAM or AWS Organizations. This data is available for an IAM policy or entity (user or role) in your account. The data for IAM includes information about the allowed services that IAM entities last attempted to access and when. To learn more about the report and how to view service last accessed data for IAM, see [Viewing Service Last Accessed Data for IAM \(p. 479\)](#).

If you sign in using master account credentials, you can also view service last accessed data for an AWS Organizations entity or policy in your organization. AWS Organizations entities include the organization root, organizational units (OUs), or accounts. The Organizations service last accessed data includes information about the allowed services that principals in an Organizations account last attempted to access and when. To learn more about the report and how to view service last accessed data for AWS Organizations, see [Viewing Service Last Accessed Data for Organizations \(p. 482\)](#).

You can then use the service last accessed data to refine your policies and allow access to only the services that your entities use. This helps you to better adhere to the best practice of [least privilege \(p. 61\)](#).

For example scenarios for using service last accessed data to make decisions about the permissions that you grant to your IAM entities or Organizations entities, see [Example Scenarios for Using Service Last Accessed Data \(p. 486\)](#).

Topics

- [Things to Know \(p. 476\)](#)
- [Permissions Required \(p. 476\)](#)
- [Troubleshooting Activity for IAM and Organizations Entities \(p. 478\)](#)
- [Regions Where Data Is Tracked \(p. 479\)](#)
- [Viewing Service Last Accessed Data for IAM \(p. 479\)](#)
- [Viewing Service Last Accessed Data for Organizations \(p. 482\)](#)

- [Example Scenarios for Using Service Last Accessed Data \(p. 486\)](#)

Things to Know

Before you use service last accessed data from a report to change the permissions for an IAM entity or Organizations entity, review the following details about the data.

- **Reporting period** – Recent activity usually appears in the IAM console within four hours. IAM reports activity for the last 365 days, or less if your Region began supporting this feature within the last year. For more information, see [Regions Where Data Is Tracked \(p. 479\)](#).
- **Authenticated IAM entities** – The data for IAM includes only authenticated IAM entities (users or roles) in your account. Data for Organizations includes only authenticated principals (IAM users, IAM roles, or the AWS account root user) in the specified Organizations entity. The data does not include unauthenticated attempts.
- **IAM policy types** – The data for IAM includes services that are allowed by an IAM entity's policies. These are policies attached to a role or attached to a user directly or through a group. Access allowed by other policy types is not included in your report. The excluded policy types include resource-based policies, access control lists, AWS Organizations SCPs, IAM permissions boundaries, and session policies. To learn how the different policy types are evaluated to allow or deny access, see [Policy Evaluation Logic \(p. 636\)](#).
- **Organizations policy types** – The data for AWS Organizations includes only services that are allowed by an Organizations entity's inherited service control policies (SCPs). SCPs are policies attached to a root, OU, or account. Access allowed by other policy types is not included in your report. The excluded policy types include identity-based policies, resource-based policies, access control lists, IAM permissions boundaries, and session policies. To learn how the different policy types are evaluated to allow or deny access, see [Policy Evaluation Logic \(p. 636\)](#).
- **Specifying a policy ID** – When you use the AWS CLI or AWS API to generate a report for service last accessed data in Organizations, you can optionally specify a policy ID. The resulting report includes data for the services that are allowed by only that policy. The data includes the most recent account activity in the specified Organizations entity or the entity's children. For more information, see [aws iam generate-organizations-access-report](#) or [GenerateOrganizationsAccessReport](#).
- **Organizations master account** – You must sign in using your organization's master account credentials to view service last accessed data. You can choose to view data for the master account using the IAM console, the AWS CLI, or the AWS API. The resulting report lists all AWS services, because the master account is not limited by SCPs. If you specify a policy ID in the CLI or API, the policy is ignored. For each service, the report includes data for only the master account. However, reports for other Organizations entities do not return data for activity in the master account.
- **Organizations settings** – An administrator must [enable SCPs in your organization root](#) before you can generate data for Organizations.

Permissions Required

To view the service last accessed data in the AWS Management Console, you must have a policy that grants the necessary permissions.

Permissions for IAM Data

To use the IAM console to view the service last accessed data for an IAM user, role, or policy, you must have a policy that includes the following actions:

- `iam:GenerateServiceLastAccessedDetails`
- `iam:Get*`
- `iam>List*`

These permissions allow a user to see the following:

- Which users, groups, or roles are attached to a [managed policy](#)
- Which services a user or role can access
- The last time they accessed the service

To use the AWS CLI or AWS API to view service last accessed data for IAM, you must have permissions that match the operation you want to use:

- `iam:GenerateServiceLastAccessedDetails`
- `iam:GetServiceLastAccessedDetails`
- `iam:GetServiceLastAccessedDetailsWithEntities`
- `iam>ListPoliciesGrantingServiceAccess`

This example shows how you might create a policy that allows viewing IAM service last accessed data. Additionally, it allows read-only access to all of IAM. This policy also grants the necessary permissions to complete this action on the console.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {"Effect": "Allow",  
         "Action": [  
             "iam:GenerateServiceLastAccessedDetails",  
             "iam:Get*",  
             "iam>List*"  
         ],  
         "Resource": "*"  
     }  
}
```

Permissions for AWS Organizations Data

To use the IAM console to view a report for the root, OU, or account entities in Organizations, you must have a policy that includes the following actions:

- `iam:GenerateOrganizationsAccessReport`
- `iam:GetOrganizationsAccessReport`
- `organizations:DescribeAccount`
- `organizations:DescribeOrganization`
- `organizations:DescribeOrganizationalUnit`
- `organizations:DescribePolicy`
- `organizations>ListChildren`
- `organizations>ListParents`
- `organizations>ListPoliciesForTarget`
- `organizations>ListRoots`
- `organizations>ListTargetsForPolicy`

To use the AWS CLI or AWS API to view service last accessed data for Organizations, you must have a policy that includes the following actions:

- `iam:GenerateOrganizationsAccessReport`
- `iam:GetOrganizationsAccessReport`

- organizations:DescribePolicy
- organizations>ListChildren
- organizations>ListParents
- organizations>ListPoliciesForTarget
- organizations>ListRoots
- organizations>ListTargetsForPolicy

This example shows how you might create a policy that allows viewing service last accessed data for Organizations. Additionally, it allows read-only access to all of Organizations. This policy also grants the necessary permissions to complete this action on the console.

```
{  
    "Version": "2012-10-17",  
    "Statement": {  
        "Effect": "Allow",  
        "Action": [  
            "iam:GenerateOrganizationsAccessReport",  
            "iam:GetOrganizationsAccessReport",  
            "organizations:Describe*",  
            "organizations>List*"  
        ],  
        "Resource": "*"  
    }  
}
```

You can also use the [iam:OrganizationsPolicyId \(p. 677\)](#) condition key to allow generating a report only for a specific Organizations policy. For an example policy, see [IAM: View Service Last Accessed Data for an Organizations Policy \(p. 432\)](#).

Troubleshooting Activity for IAM and Organizations Entities

In some cases, your AWS Management Console service last accessed data table might be empty. Or perhaps your AWS CLI or AWS API request returns an empty set of data or a null field. In these cases, review the following issues:

- For an IAM user, make sure that the user has at least one inline or managed policy attached, either directly or through group memberships.
- For an IAM group, verify that the group has at least one inline or managed policy attached.
- For an IAM group, the report returns only the service last access data for members that used the group's policies to access a service. To learn whether a member used other policies, review the service last accessed data for that user.
- For an IAM role, verify that the role has at least one inline or managed policy attached.
- For an IAM entity (user or role), review other policy types that might affect the permissions of that entity. These include resource-based policies, access control lists, AWS Organizations policies, IAM permissions boundaries, or session policies. For more information, see [Policy Types \(p. 356\)](#) or [Evaluating Policies Within a Single Account \(p. 636\)](#).
- For an IAM policy, make sure that the specified managed policy is attached to at least one user, group with members, or role.
- For an Organizations entity (root, OU, or account), make sure that you are signed using Organizations master account credentials.
- Verify that [SCPs are enabled in your organization root](#).

When you make changes, wait at least four hours for activity to appear in your IAM console report. If you use the AWS CLI or AWS API, you must generate a new report to view the updated data.

Regions Where Data Is Tracked

AWS collects service last accessed data in most Regions. Data is stored for a maximum of 365 days. When AWS adds additional Regions, those Regions are added to the following table, including the date that AWS started tracking data in each Region:

Region Name	Region	Tracking Start Date
US East (Ohio)	us-east-2	October 27, 2017
US East (N. Virginia)	us-east-1	October 1, 2015
US West (N. California)	us-west-1	October 1, 2015
US West (Oregon)	us-west-2	October 1, 2015
Asia Pacific (Tokyo)	ap-northeast-1	October 1, 2015
Asia Pacific (Seoul)	ap-northeast-2	January 6, 2016
Asia Pacific (Singapore)	ap-southeast-1	October 1, 2015
Asia Pacific (Sydney)	ap-southeast-2	October 1, 2015
Asia Pacific (Mumbai)	ap-south-1	June 27, 2016
Asia Pacific (Hong Kong)	ap-east-1	April 24, 2019
Middle East (Bahrain)	me-south-1	July 29, 2019
Canada (Central)	ca-central-1	October 28, 2017
Europe (Frankfurt)	eu-central-1	October 1, 2015
Europe (Stockholm)	eu-north-1	December 12, 2018
Europe (Ireland)	eu-west-1	October 1, 2015
Europe (London)	eu-west-2	October 28, 2017
Europe (Paris)	eu-west-3	December 18, 2017
South America (São Paulo)	sa-east-1	December 11, 2015

If a Region is not listed in the previous table, then that Region does not yet provide service last accessed data.

Viewing Service Last Accessed Data for IAM

You can view service last accessed data for IAM using the AWS Management Console, AWS CLI, or AWS API. For more information about service last accessed data, see [Refining Permissions Using Service Last Accessed Data \(p. 475\)](#).

You can view data for each type of resource in IAM. In each case, the data includes allowed services for the given reporting period:

- **User** – View the last time that the user tried to access each allowed service.
- **Group** – View information about the last time that a group member attempted to access each allowed service. This report also includes the total number of members that attempted access.

- **Role** – View the last time that someone used the role in an attempt to access each allowed service.
- **Policy** – View information about the last time that a user or role attempted to access each allowed service. This report also includes the total number of entities that attempted access.

Note

Before you view the access data for a resource in IAM, make sure you understand the reporting period, reported entities, and the evaluated policy types for your data. For more details, see the section called "Things to Know" (p. 476).

Viewing Data for IAM (Console)

You can view service last accessed data for IAM on the **Access Advisor** tab in the IAM console.

To view data for IAM (console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose either **Groups, Users, Roles, or Policies**.
3. Choose any user, group, role, or policy name to open its **Summary** page and choose the **Access Advisor** tab. View the following information, based on the resource that you chose:
 - **Group** – View the list of services that group members (users) can access, when a member last accessed the service, what group policies they used, and which group member made the request. Choose the name of the policy to learn whether it is a managed policy or an inline group policy. Choose the name of the group member to see all of the members of the group and when they last accessed the service.
 - **User** – View the list of services that the user can access, when they last accessed the service, and what policies they used. Choose the name of the policy to learn whether the policy is managed, an inline user policy, or an inline policy for the group to which the user belongs.
 - **Role** – View the list of services that the role can access, when the role last accessed the service, and what policies were used. Choose the name of the policy to learn whether it is a managed policy or an inline role policy.
 - **Policy** – View the list of services with allowed actions in the policy, when the policy was last used to access the service, and which entity (user or role) used the policy. Choose the name of the entity to learn which entities have this policy attached and when they last accessed the service.

Viewing Data for IAM (AWS CLI)

You can use the AWS CLI to retrieve data about the last time that an IAM resource was used to attempt to access AWS services. An IAM resource can be a user, group, role, or policy.

To view data for IAM (AWS CLI)

1. Generate a report. The request must include the ARN of the IAM resource (user, group, role, or policy) for which you want a report. It returns a job-id that you can then use in the `get-service-last-accessed-details` and `get-service-last-accessed-details-with-entities` operations to monitor the job-status until the job is complete.
 - [aws iam generate-service-last-accessed-details](#)
2. Retrieve details about the report using the job-id parameter from the previous step.
 - [aws iam get-service-last-accessed-details](#)

This operation returns the following information, based on the type of resource that you requested in the `generate-service-last-accessed-details` operation:

- **User** – Returns a list of services that the specified user can access. For each service, the operation returns the date and time of the user's last attempt and the ARN of the user.
 - **Group** – Returns a list of services that members of the specified group can access using the policies attached to the group. For each service, the operation returns the date and time of the last attempt made by any group member (user). It also returns the ARN of that user and the total number of group members that have attempted to access the service. Use the [GetServiceLastAccessedDetailsWithEntities](#) operation to retrieve a list of all of the members.
 - **Role** – Returns a list of services that the specified role can access. For each service, the operation returns the date and time of the role's last attempt and the ARN of the role.
 - **Policy** – Returns a list of services for which the specified policy allows access. For each service, the operation returns the date and time that an entity (user or role) last attempted to access the service using the policy. It also returns the ARN of that entity and the total number of entities that attempted access.
3. Learn more about the entities that used group or policy permissions in an attempt to access a specific service. This operation returns a list of entities with each entity's ARN, ID, name, path, type (user or role), and when they last attempted to access the service. You can also use this operation for users and roles, but it only returns information about that entity.
 - [aws iam get-service-last-accessed-details-with-entities](#)
 4. Learn more about the identity-based policies that an identity (user, group, or role) used in an attempt to access a specific service. When you specify an identity and service, this operation returns a list of permissions policies that the identity can use to access the specified service. This operation gives the current state of policies and does not depend on the generated report. It also does not return other policy types, such as resource-based policies, access control lists, AWS Organizations policies, IAM permissions boundaries, or session policies. For more information, see [Policy Types \(p. 356\)](#) or [Evaluating Policies Within a Single Account \(p. 636\)](#).
 - [aws iam list-policies-granting-service-access](#)

Viewing Data for IAM (AWS API)

You can use the AWS API to retrieve data about the last time that an IAM resource was used to attempt to access AWS services. An IAM resource can be a user, group, role, or policy.

To view data for IAM (AWS API)

1. Generate a report. The request must include the ARN of the IAM resource (user, group, role, or policy) for which you want a report. It returns a `JobId` that you can then use in the `GetServiceLastAccessedDetails` and `GetServiceLastAccessedDetailsWithEntities` operations to monitor the `JobStatus` until the job is complete.
 - [GenerateServiceLastAccessedDetails](#)
2. Retrieve details about the report using the `JobId` parameter from the previous step.
 - [GetServiceLastAccessedDetails](#)

This operation returns the following information, based on the type of resource that you requested in the `GenerateServiceLastAccessedDetails` operation:

- **User** – Returns a list of services that the specified user can access. For each service, the operation returns the date and time of the user's last attempt and the ARN of the user.
- **Group** – Returns a list of services that members of the specified group can access using the policies attached to the group. For each service, the operation returns the date and time of the last attempt made by any group member (user). It also returns the ARN of that user

and the total number of group members that have attempted to access the service. Use the [GetServiceLastAccessedDetailsWithEntities](#) operation to retrieve a list of all of the members.

- **Role** – Returns a list of services that the specified role can access. For each service, the operation returns the date and time of the role's last attempt and the ARN of the role.
 - **Policy** – Returns a list of services for which the specified policy allows access. For each service, the operation returns the date and time that an entity (user or role) last attempted to access the service using the policy. It also returns the ARN of that entity and the total number of entities that attempted access.
3. Learn more about the entities that used group or policy permissions in an attempt to access a specific service. This operation returns a list of entities with each entity's ARN, ID, name, path, type (user or role), and when they last attempted to access the service. You can also use this operation for users and roles, but it only returns information about that entity.
 - [GetServiceLastAccessedDetailsWithEntities](#)
 4. Learn more about the identity-based policies that an identity (user, group, or role) used in an attempt to access a specific service. When you specify an identity and service, this operation returns a list of permissions policies that the identity can use to access the specified service. This operation gives the current state of policies and does not depend on the generated report. It also does not return other policy types, such as resource-based policies, access control lists, AWS Organizations policies, IAM permissions boundaries, or session policies. For more information, see [Policy Types \(p. 356\)](#) or [Evaluating Policies Within a Single Account \(p. 636\)](#).
 - [ListPoliciesGrantingServiceAccess](#)

Viewing Service Last Accessed Data for Organizations

You can view service last accessed data for AWS Organizations using the IAM console, AWS CLI, or AWS API. For important information about the data, permissions required, troubleshooting, and supported Regions, see [Refining Permissions Using Service Last Accessed Data \(p. 475\)](#).

When you sign in to the IAM console using AWS Organizations master account credentials, you can view data for any entity in your organization. Organizations entities include the organization root, organizational units (OUs), and accounts. You can also use the IAM console to view data for any service control policies (SCPs) in your organization. IAM shows a list of services that are allowed by any SCPs that apply to the entity. For each service, you can view the most recent account activity data for the chosen Organizations entity or the entity's children.

When you use the AWS CLI or AWS API with master account credentials, you can generate a data report for any entities or policies in your organization. A programmatic report for an entity includes a list of services that are allowed by any SCPs that apply to the entity. For each service, the report includes the most recent activity for accounts in the specified Organizations entity or the entity's subtree.

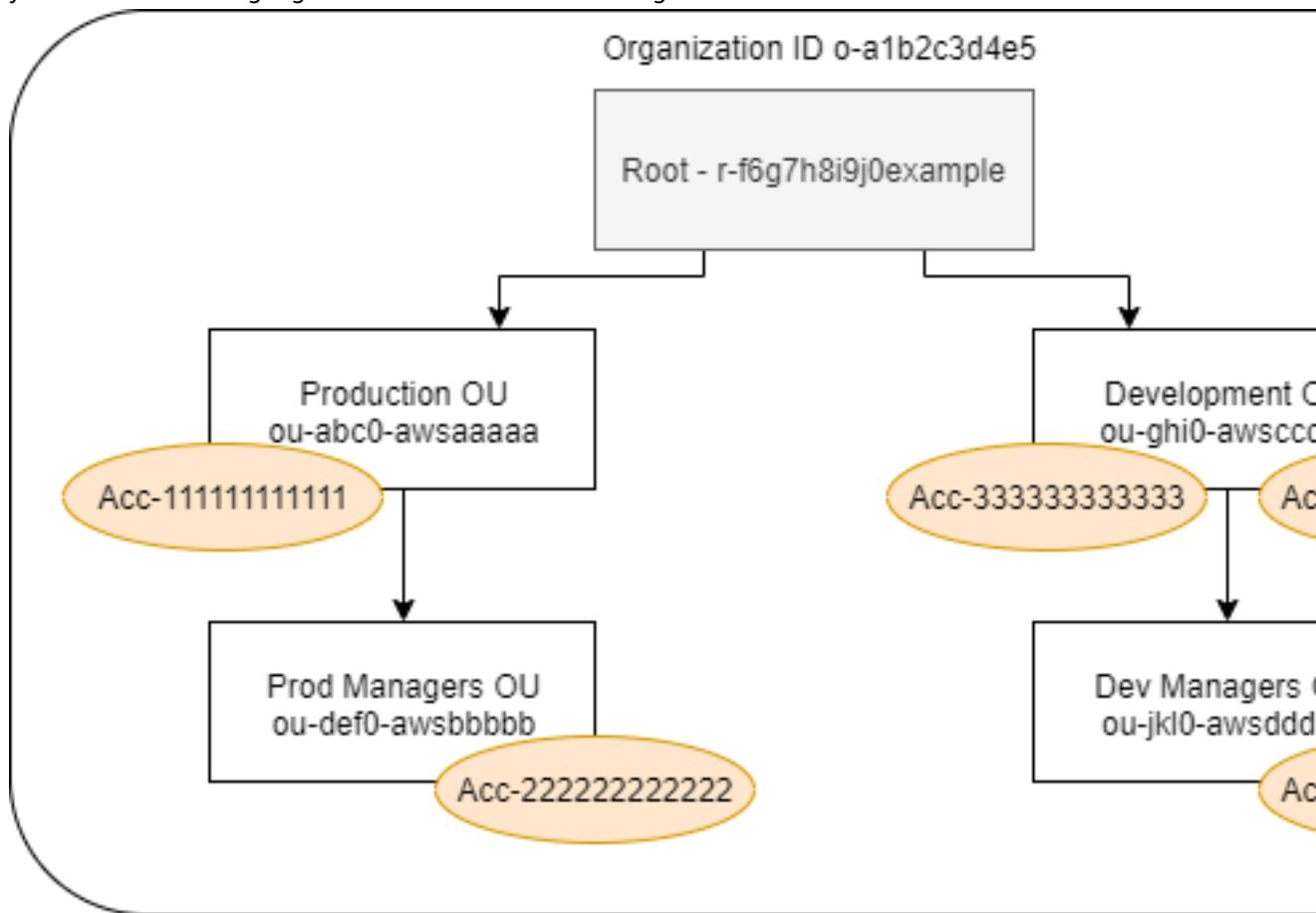
When you generate a programmatic data report for a policy, you must specify an Organizations entity. This report includes a list of services that are allowed by the specified SCP. For each service, it includes the most recent account activity in the entity or entity's children that are granted permission by that policy. For more information, see [aws iam generate-organizations-access-report](#) or [GenerateOrganizationsAccessReport](#).

Before you view the report, make sure that you understand the master account requirements and data, reporting period, reported entities, and the evaluated policy types. For more details, see [the section called "Things to Know" \(p. 476\)](#).

Understand the AWS Organizations Entity Path

When you use the AWS CLI or AWS API to generate an AWS Organizations access report, you must specify an entity path. A path is a text representation of the structure of an Organizations entity.

You can build an entity path using the known structure of your organization. For example, assume that you have the following organizational structure in AWS Organizations.



The path for the **Dev Managers OU** is built using the IDs of the organization, root, and all OUs in the path down to and including the OU.

```
o-a1b2c3d4e5/r-f6g7h8i9j0example/ou-ghi0-awscccccc/ou-jkl0-awsdddddd
```

The path for the account in the **Production OU** is built using the IDs of the organization, root, the OU, and the account number.

```
o-a1b2c3d4e5/r-f6g7h8i9j0example/ou-abc0-awsaaaaaa/111111111111
```

Note

Organization IDs are globally unique but OU IDs and root IDs are unique only within an organization. This means that no two organizations share the same organization ID. However, another organization might have an OU or root with the same ID as yours. We recommend that you always include the organization ID when you specify an OU or root.

Viewing Data for Organizations (Console)

You can use the IAM console to view service last accessed data for your root, OU, account, or policy.

To view data for the root (console)

1. Sign in to the AWS Management Console using Organizations master account credentials, and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane below the **Access reports** section, choose **Organization activity**.
3. On the **Organization activity** page, choose **Root**.
4. On the **Details and activity** tab, view the **Service access report** section. The data includes a list of services that are allowed by the policies that are attached directly to the root. The data shows you from which account the service was last accessed and when. For more details about which principal accessed the service, sign in as an administrator in that account and [view the IAM service last accessed data \(p. 479\)](#).
5. Choose the **Attached SCPs** tab to view the list of the service control policies (SCPs) that are attached to the root. IAM shows you the number of target entities to which each policy is attached. You can use this information to decide which SCP to review.
6. Choose the name of an SCP to view all of the services that the policy allows. For each service, view from which account the service was last accessed, and when.
7. Choose **Edit in AWS Organizations** to view additional details and edit the SCP in the Organizations console. For more information, see [Updating an SCP](#) in the *AWS Organizations User Guide*.

To view data for an OU or account (console)

1. Sign in to the AWS Management Console using Organizations master account credentials, and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane below the **Access reports** section, choose **Organization activity**.
3. On the **Organization activity** page, expand the structure of your organization. Then choose the name of the OU or any account that you want to view except the master account.
4. On the **Details and activity** tab, view the **Service access report** section. The data includes a list of services that are allowed by the SCPs attached to the OU or account *and* all of its parents. The data shows you from which account the service was last accessed and when. For more details about which principal accessed the service, sign in as an administrator in that account and [view the IAM service last accessed data \(p. 479\)](#).
5. Choose the **Attached SCPs** tab to view the list of the service control policies (SCPs) that are attached directly to the OU or account. IAM shows you the number of target entities to which each policy is attached. You can use this information to decide which SCP to review.
6. Choose the name of an SCP to view all of the services that the policy allows. For each service, view from which account the service was last accessed, and when.
7. Choose **Edit in AWS Organizations** to view additional details and edit the SCP in the Organizations console. For more information, see [Updating an SCP](#) in the *AWS Organizations User Guide*.

To view data for the master account (console)

1. Sign in to the AWS Management Console using Organizations master account credentials, and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane below the **Access reports** section, choose **Organization activity**.
3. On the **Organization activity** page, expand the structure of your organization and choose the name of your master account.
4. On the **Details and activity** tab, view the **Service access report** section. The data includes a list of all AWS services. The master account is not limited by SCPs. The data shows you whether the account last accessed the service and when. For more details about which principal accessed the service, sign in as an administrator in that account and [view the IAM service last accessed data \(p. 479\)](#).
5. Choose the **Attached SCPs** tab to confirm that there are no attached SCPs because the account is the master account.

To view data for a policy (console)

1. Sign in to the AWS Management Console using Organizations master account credentials, and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane below the **Access reports** section, choose **Service control policies (SCPs)**.
3. On the **Service control policies (SCPs)** page, view a list of the policies in your organization. You can view the number of target entities to which each policy is attached.
4. Choose the name of an SCP to view all of the services that the policy allows. For each service, view from which account the service was last accessed, and when.
5. Choose **Edit in AWS Organizations** to view additional details and edit the SCP in the Organizations console. For more information, see [Updating an SCP](#) in the *AWS Organizations User Guide*.

Viewing Data for Organizations (AWS CLI)

You can use the AWS CLI to retrieve service last accessed data for your Organizations root, OU, account, or policy.

To view Organizations service last accessed data (AWS CLI)

1. Use your Organizations master account credentials with the required IAM and Organizations permissions, and confirm that SCPs are enabled for your root. For more information, see [Things to Know \(p. 476\)](#).
2. Generate a report. The request must include the path of the Organizations entity (root, OU, or account) for which you want a report. You can optionally include an `organization-policy-id` parameter to view a report for a specific policy. The command returns a `job-id` that you can then use in the `get-organizations-access-report` command to monitor the `job-status` until the job is complete.
 - [`aws iam generate-organizations-access-report`](#)
3. Retrieve details about the report using the `job-id` parameter from the previous step.
 - [`aws iam get-organizations-access-report`](#)

This command returns a list of services that entity members can access. For each service, the command returns the date and time of an account member's last attempt and the entity path of the account. It also returns the total number of services that are available to access and the number of services that were not accessed. If you specified the optional `organization-policy-id` parameter, then the services that are available to access are those that are allowed by the specified policy.

Viewing Data for Organizations (AWS API)

You can use the AWS API to retrieve service last accessed data for your Organizations root, OU, account, or policy.

To view Organizations service last accessed data (AWS API)

1. Use your Organizations master account credentials with the required IAM and Organizations permissions, and confirm that SCPs are enabled for your root. For more information, see [Things to Know \(p. 476\)](#).
2. Generate a report. The request must include the path of the Organizations entity (root, OU, or account) for which you want a report. You can optionally include an `OrganizationPolicyId` parameter to view a report for a specific policy. The operation returns a `JobId` that you can then

use in the `GetOrganizationsAccessReport` operation to monitor the `JobStatus` until the job is complete.

- [GenerateOrganizationsAccessReport](#)
3. Retrieve details about the report using the `JobId` parameter from the previous step.
- [GetOrganizationsAccessReport](#)

This operation returns a list of services that entity members can access. For each service, the operation returns the date and time of an account member's last attempt and the entity path of the account. It also returns the total number of services that are available to access, and the number of services that were not accessed. If you specified the optional `OrganizationsPolicyId` parameter, then the services that are available to access are those that are allowed by the specified policy.

Example Scenarios for Using Service Last Accessed Data

You can use service last accessed data to make decisions about the permissions that you grant to your IAM entities or AWS Organizations entities. For more information, see [Refining Permissions Using Service Last Accessed Data \(p. 475\)](#).

Note

Before you view the access data for an entity or policy in IAM or AWS Organizations, make sure that you understand the reporting period, reported entities, and the evaluated policy types for your data. For more details, see [the section called "Things to Know" \(p. 476\)](#).

It's up to you as an administrator to balance the accessibility and least privilege that's appropriate for your company.

Using Data to Reduce Permissions for an IAM Group

You can use service last accessed data to reduce IAM group permissions to include only those services that your users need. This method is an important step in [granting least privilege \(p. 61\)](#) at a service level.

For example, Paulo Santos is the administrator in charge of defining AWS user permissions for Example Corp. This company just started using AWS, and the software development team has not yet defined what AWS services they will use. Paulo wants to give the team permission to access only the services they need, but since that is not yet defined, he temporarily gives them power-user permissions. Then he uses service last accessed data to reduce the group's permissions.

Paulo creates a managed policy named `ExampleDevelopment` using the following JSON text. He then attaches it to a group named `Development` and adds all of the developers to the group.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "FullAccessToAllServicesExceptPeopleManagement",  
            "Effect": "Allow",  
            "NotAction": [  
                "iam:*",  
                "organizations:/*"  
            ],  
            "Resource": "*"  
        },  
        {  
            "Sid": "RequiredIamAndOrgsActions",  
            "Effect": "Deny",  
            "Action": "iam:List*",  
            "Resource": "*"  
        }  
    ]  
}
```

```
        "Effect": "Allow",
        "Action": [
            "iam:CreateServiceLinkedRole",
            "iam>DeleteServiceLinkedRole",
            "iam>ListRoles",
            "organizations:DescribeOrganization"
        ],
        "Resource": "*"
    }
]
```

Paulo decides to wait for 90 days before he [views the service last accessed data \(p. 480\)](#) for the Development group using the AWS Management Console. He views the list of services that the group members accessed. He learns that the users accessed five services within the last week: AWS CloudTrail, Amazon CloudWatch Logs, Amazon EC2, AWS KMS, and Amazon S3. They accessed a few other services when they were first evaluating AWS, but not since then.

Paulo decides to reduce the policy permissions to include only those five services and the required IAM and Organizations actions. He edits `ExampleDevelopment` policy using the following JSON text.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "FullAccessToListedServices",
            "Effect": "Allow",
            "Action": [
                "s3:*",
                "kms:*",
                "cloudtrail:*",
                "logs:*",
                "ec2:*
```

```
            ],
            "Resource": "*"
        },
        {
            "Sid": "RequiredIamAndOrgsActions",
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceLinkedRole",
                "iam>DeleteServiceLinkedRole",
                "iam>ListRoles",
                "organizations:DescribeOrganization"
            ],
            "Resource": "*"
        }
    ]
}
```

To further reduce permissions, Paulo can view the account's events in AWS CloudTrail [Event history](#). There he can view detailed event information that he can use to reduce the policy's permissions to include only the actions and resources that the developers need. For more information, see [Viewing CloudTrail Events in the CloudTrail Console](#) in the [AWS CloudTrail User Guide](#).

Using Data to Reduce Permissions for an IAM User

You can use service last accessed data to reduce the permissions for an individual IAM user.

For example, Martha Rivera is an IT administrator responsible for ensuring that people in her company do not have excess AWS permissions. As part of a periodic security check, she reviews the permissions of

all IAM users. One of these users is an application developer named Nikhil Jayashankar, who previously filled the role of a security engineer. Because of the change in job requirements, Nikhil is a member of both the app-dev group and the security-team group. The app-dev group for his new job grants permissions to multiple services including Amazon EC2, Amazon EBS, Auto Scaling, Route 53, and Elastic Transcoder. The security-team group for his old job grants permissions to IAM and CloudTrail.

As an administrator, Martha signs into the IAM console and chooses **Users**, chooses the name `nikhilj`, and then chooses the **Access Advisor** tab.

Martha reviews the **Last Accessed** column and notices that Nikhil has not recently accessed IAM, CloudTrail, Route 53, Amazon Elastic Transcoder, and a number of other AWS services. Within her company, Martha confirms that Nikhil has no business need to access IAM and CloudTrail anymore because he is no longer a member of the internal security team.

Martha is now ready to act on the service last accessed data. However, unlike the group in the previous example, an IAM user like `nikhilj` might be subject to multiple policies and be a member of multiple groups. Martha must proceed with caution to avoid inadvertently disrupting access for `nikhilj` or other group members. In addition to learning what access Nikhil should have, she must determine *how* he is receiving these permissions.

Martha chooses the **Permissions** tab, where she views which policies are attached directly to `nikhilj` and those attached from a group. She expands each policy and views the policy summary to learn which policy allows access to the services that Nikhil is not using:

- **IAM** – The `IAMFullAccess` AWS managed policy is attached directly to `nikhilj` and attached to the `security-team` group.
- **CloudTrail** – The `AWSCloudTrailReadOnlyAccess` AWS managed policy is attached to the `security-team` group.
- **Route 53** – The `App-Dev-Route53` customer managed policy is attached to the `app-dev` group.
- **Elastic Transcoder** – The `App-Dev-ElasticTranscoder` customer managed policy is attached to the `app-dev` group.

Martha decides to remove the `IAMFullAccess` AWS managed policy that is attached directly to `nikhilj`. She also removes Nikhil's membership to the `security-team` group. These two actions remove the unnecessary access to IAM and CloudTrail.

Nikhil's permissions to access to Route 53 and Elastic Transcoder are granted by the `app-dev` group. Although Nikhil isn't using those services, other members of the group might be. Martha reviews the service last accessed data for the `app-dev` group and learns that several members recently accessed Route 53, but no group members have accessed Elastic Transcoder in the last year. She removes the `App-Dev-ElasticTranscoder` customer managed policy from the group.

Martha then reviews the service last accessed data for the `App-Dev-ElasticTranscoder` customer managed policy. She learns that the policy is not attached to any other IAM identities. She investigates within her company to make sure that the policy will not be needed in the future, and then she deletes it.

Using Data Before Deleting IAM Resources

You can use service last accessed data before you delete an IAM resource to make sure that a certain amount of time has passed since someone last used the resource. This applies to users, groups, roles, and policies. To learn more about these actions, see the following topics:

- **Users** – [Deleting a user \(p. 93\)](#)
- **Groups** – [Deleting a group \(p. 173\)](#)
- **Roles** – [Deleting a role \(p. 289\)](#)
- **Policies** – [Deleting a managed policy \(this also detaches the policy from identities\) \(p. 472\)](#)

Using Data Before Editing IAM Policies

You can review service last accessed data for an IAM identity (user, group, or role), or for an IAM policy before editing a policy that affects that resource. This is important because you don't want to remove access for someone that is using it.

For example, Arnav Desai is a developer and AWS administrator for Example Corp. When his team started using AWS, they gave all developers power-user access that allowed them full access to all services except IAM and Organizations. As a first step towards [granting least privilege \(p. 61\)](#), Arnav wants to use the AWS CLI to review the managed policies in his account.

To do this, Arnav first lists the customer managed permissions policies in his account that are attached to an identity, using the following command:

```
aws iam list-policies --scope Local --only-attached --policy-usage-filter PermissionsPolicy
```

From the response, he captures the ARN for each policy. Arnav then generates a report for service last accessed data for each policy using the following command.

```
aws iam generate-service-last-accessed-details --arn arn:aws:iam::123456789012:policy/ExamplePolicy1
```

From that response, he captures the ID of the generated report from the `JobId` field. Arnav then polls the following command until the `JobStatus` field returns a value of `COMPLETED` or `FAILED`. If the job failed, he captures the error.

```
aws iam get-service-last-accessed-details --job-id 98a765b4-3cde-2101-2345-example678f9
```

When the job has a status of `COMPLETED`, Arnav parses the contents of the JSON-formatted `ServicesLastAccessed` array.

```
"ServicesLastAccessed": [
    {
        "TotalAuthenticatedEntities": 1,
        "LastAuthenticated": "2018-11-01T21:24:33.222Z",
        "ServiceNamespace": "dynamodb",
        "LastAuthenticatedEntity": "arn:aws:iam::123456789012:user/IAMExampleUser",
        "ServiceName": "Amazon DynamoDB"
    },
    {
        "TotalAuthenticatedEntities": 0,
        "ServiceNamespace": "ec2",
        "ServiceName": "Amazon EC2"
    },
    {
        "TotalAuthenticatedEntities": 3,
        "LastAuthenticated": "2018-08-25T15:29:51.156Z",
        "ServiceNamespace": "s3",
        "LastAuthenticatedEntity": "arn:aws:iam::123456789012:role/IAMExampleRole",
        "ServiceName": "Amazon S3"
    }
]
```

From this information, Arnav learns that the `ExamplePolicy1` policy allows access to three services, Amazon DynamoDB, Amazon S3, and Amazon EC2. The IAM user named `IAMExampleUser` last

attempted to access DynamoDB on November 1, and someone used the IAMExampleRole role to attempt to access Amazon S3 on August 25. There are also two more entities that attempted to access Amazon S3 in the last year. However, nobody has attempted to access Amazon EC2 in the last year.

This means that Arnav can safely remove the Amazon EC2 actions from the policy. Arnav wants to review the current JSON document for the policy. First, he must determine the version number of the policy using the following command.

```
aws iam list-policy-versions --policy-arn arn:aws:iam::123456789012:policy/ExamplePolicy1
```

From the response, Arnav collects the current default version number from the `Versions` array. He then uses that version number (`v2`) to request the JSON policy document using the following command.

```
aws iam get-policy-version --policy-arn arn:aws:iam::123456789012:policy/ExamplePolicy1 --version-id v2
```

Arnav stores the JSON policy document returned in the `Document` field of the `PolicyVersion` array. Within the policy document, Arnav searches for actions with in the `ec2` namespace. If there are no actions from other namespaces remaining in the policy, then he detaches the policy from the affected identities (users, groups, and roles). He then deletes the policy. In this case, the policy does include the Amazon DynamoDB and Amazon S3 services. So Arnav removes the Amazon EC2 actions from the document and saves his changes. He then uses the following command to update the policy using the new version of the document and to set that version as the default policy version.

```
aws iam create-policy-version --policy-arn arn:aws:iam::123456789012:policy/ExamplePolicy1 --policy-document file://UpdatedPolicy.json --set-as-default
```

The `ExamplePolicy1` policy is now updated to remove access to the unnecessary Amazon EC2 service.

Other IAM Scenarios

Information about when an IAM resource (user, group, role, or policy) last attempted to access a service can help you when you complete any of the following tasks:

- **Policies** – Editing an existing customer-managed or inline policy to remove permissions (p. 468)
- **Policies** – Converting an inline policy to a managed policy and then deleting it (p. 63)
- **Policies** – Adding an explicit deny to an existing policy (p. 642)
- **Policies** – Detaching a managed policy from an identity (user, group, or role) (p. 460)
- **Entities** – Set a permissions boundary to control the maximum permissions that an entity (user or role) can have (p. 457)
- **Groups** – Removing users from a group (p. 171)

Using Data to Refine Permissions for an Organizational Unit

You can use service last accessed data to refine the permissions for an organizational unit (OU) in AWS Organizations.

For example, John Stiles is an AWS Organizations administrator. He is responsible for ensuring that people in company AWS accounts do not have excess permissions. As part of a periodic security audit, he reviews the permissions of his organization. His `Development` OU contains accounts that are often used to test new AWS services. John decides to periodically review the report for services that have not been accessed in more than 180 days. He then removes permissions for the OU members to access those services.

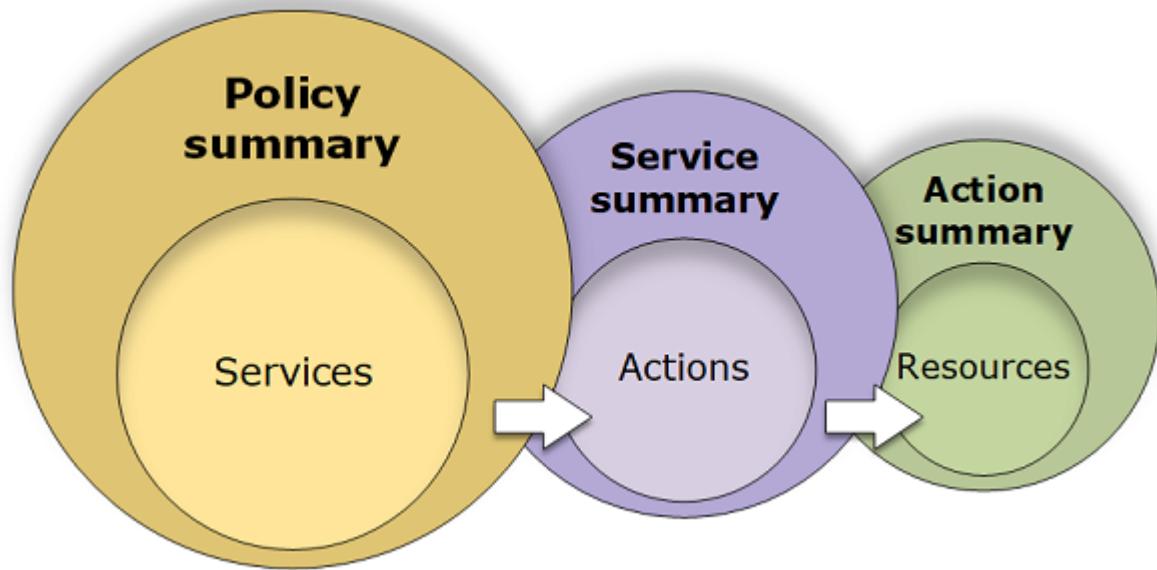
John signs into the IAM console using his master account credentials. In the IAM console, he locates the Organizations data for the Development OU. He reviews the **Service access report** table and sees two AWS services that have not been accessed in more than his preferred period of 180 days. He remembers adding permissions for the development teams to access Amazon Lex and AWS Database Migration Service. John contacts the development teams and confirms that they no longer have a business need to test these services.

John is now ready to act on the service last accessed data. He chooses **Edit in AWS Organizations** and is reminded that the SCP is attached to multiple entities. He chooses **Continue**. In AWS Organizations, he reviews the targets to learn to which Organizations entities that the SCP is attached. All of entities are within the Development OU.

John decides to deny access to the Amazon Lex and AWS Database Migration Service actions in the NewServiceTest SCP. This action removes the unnecessary access to the services.

Understanding Permissions Granted by a Policy

The IAM console includes *policy summary* tables that describe the access level, resources, and conditions that are allowed or denied for each service in a policy. Policies are summarized in three tables: the [policy summary \(p. 492\)](#), the [service summary \(p. 502\)](#), and the [action summary \(p. 507\)](#). The *policy summary* table includes a list of services. Choose a service there to see the *service summary*. This summary table includes a list of the actions and associated permissions for the chosen service. You can choose an action from that table to view the *action summary*. This table includes a list of resources and conditions for the chosen action.



You can view policy summaries on the **Users** page or **Roles** page for all policies (managed and inline) that are attached to that user. View summaries on the **Policies** page for all managed policies. Managed policies include AWS managed policies, AWS managed job function policies, and customer managed policies. You can view summaries for these policies on the **Policies** page regardless of whether they are attached to a user or other IAM identity.

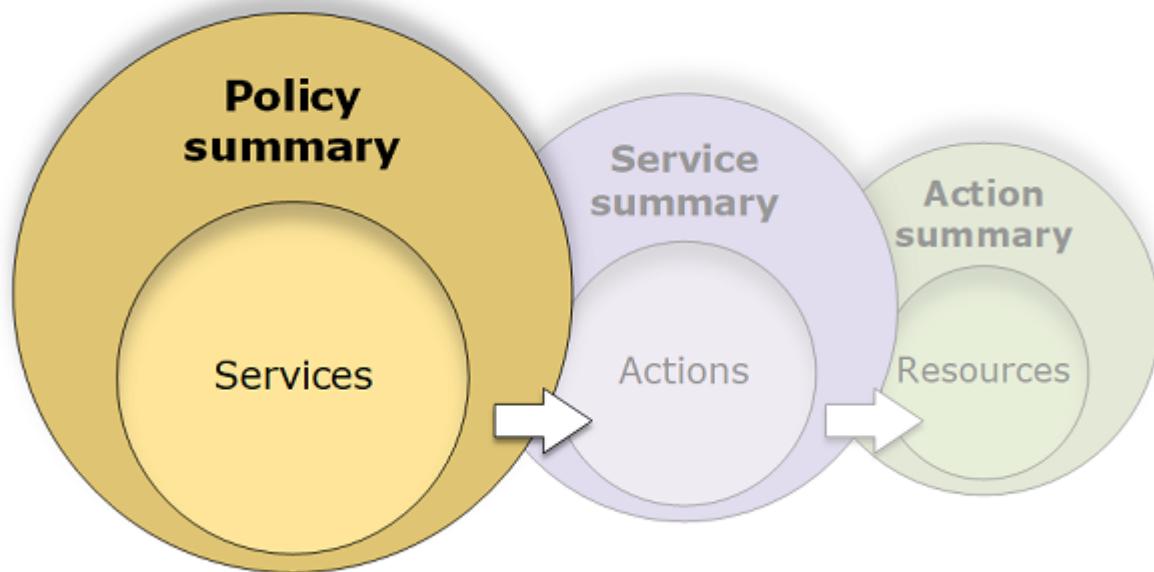
You can use the information in the policy summaries to understand the permissions that are allowed or denied by your policy. Policy summaries can help you [troubleshoot \(p. 547\)](#) and fix policies that are not providing the permissions that you expect.

Topics

- [Policy Summary \(List of Services\) \(p. 492\)](#)
- [Service Summary \(List of Actions\) \(p. 502\)](#)
- [Action Summary \(List of Resources\) \(p. 507\)](#)
- [Examples of Policy Summaries \(p. 510\)](#)

Policy Summary (List of Services)

Policies are summarized in three tables: the policy summary, the [service summary \(p. 502\)](#), and the [action summary \(p. 507\)](#). The *policy summary* table includes a list of services and summaries of the permissions that are defined by the chosen policy.



The policy summary table is grouped into one or more **Uncategorized services**, **Explicit deny**, and **Allow** sections. If the policy includes a service that IAM does not recognize, then the service is included in the **Uncategorized services** section of the table. If IAM recognizes the service, then it is included under the **Explicit deny** or **Allow** sections of the table, depending on the effect of the policy (Deny or Allow).

Viewing Policy Summaries

You can view the summaries for any policies that are attached to a user on the **Users** page. You can view the summaries for any policies that are attached to a role on the **Roles** page. You can view the policy summary for managed policies on the **Policies** page. If your policy does not include a policy summary, see [Missing Policy Summary \(p. 551\)](#) to learn why.

To view the policy summary from the Policies page

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Policies**.
3. In the list of policies, choose the name of the policy that you want to view.
4. On the **Summary** page for the policy, view the **Permissions** tab to see the policy summary.

To view the summary for a policy attached to a user

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. Choose **Users** from the navigation pane.
3. In the list of users, choose the name of the user whose policy you want to view.
4. On the **Summary** page for the user, view the **Permissions** tab to see the list of policies that are attached to the user directly or from a group.
5. In the table of policies for the user, expand the row of the policy that you want to view.

To view the summary for a policy attached to a role

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Roles**.
3. In the list of roles, choose the name of the role whose policy you want to view.
4. On the **Summary** page for the role, view the **Permissions** tab to see the list of policies that are attached to the role.
5. In the table of policies for the role, expand the row of the policy that you want to view.

Editing Policies to Fix Warnings

While viewing a policy summary, you might find a typo or notice that the policy does not provide the permissions that you expected. You cannot edit a policy summary directly. However, you can edit a managed policy using the visual policy editor, which catches many of the same errors and warnings that the policy summary reports. You can then view the changes in the policy summary to confirm that you fixed all of the issues. To learn how to edit an inline policy, see [the section called "Editing IAM Policies" \(p. 468\)](#). You cannot edit AWS managed policies.

To edit a policy for your policy summary using the Visual editor tab

1. Open the policy summary as explained in the previous procedures.
2. Choose **Edit policy**.

If you are on the **Users** page and choose to edit a customer managed policy that is attached to that user, you are redirected to the **Policies** page. You can edit customer managed policies only on the **Policies** page.

3. Choose the **Visual editor** tab to view the editable visual representation of your policy. IAM might restructure your policy to optimize it for the visual editor and to make it easier for you to find and fix any problems. The warnings and error messages on the page can guide you to fix any issues with your policy. For more information about how IAM restructures policies, see [Policy Restructuring \(p. 548\)](#).
4. Edit your policy and choose **Review policy** to see your changes reflected in the policy summary. If you still see a problem, choose **Previous** to return to the editing screen.
5. Choose **Save** to save your changes.

To edit a policy for your policy summary using the JSON tab

1. Open the policy summary as explained in the previous procedures.
2. Choose **{ } JSON** and **Policy summary** to compare the policy summary to the JSON policy document. You can use this information to determine which lines in the policy document you want to change.

3. Choose **Edit policy** and then choose the **JSON** tab to edit the JSON policy document.

Note

You can switch between the **Visual editor** and **JSON** tabs any time. However, if you make changes or choose **Review policy** in the **Visual editor** tab, IAM might restructure your policy to optimize it for the visual editor. For more information, see [Policy Restructuring \(p. 548\)](#).

If you are on the **Users** page and choose to edit a customer managed policy that is attached to that user, you are redirected to the **Policies** page. You can edit customer managed policies only on the **Policies** page.

4. Edit your policy and choose **Review policy** to see your changes reflected in the policy summary. If you still see a problem, choose **Previous** to return to the editing screen.
5. Choose **Save** to save your changes.

Understanding the Elements of a Policy Summary

In the following example of a user details page, the **PolSumUser** user has eight attached policies. The **SummaryAllElements** policy is a managed policy (customer managed policy) that is attached directly to the user. This policy is expanded to show the policy summary. To view the JSON policy document for this policy, see [the section called “**SummaryAllElements** JSON Policy Document” \(p. 499\)](#).

The screenshot shows the AWS IAM Policy Summary page for a user named PolSumUser. At the top, it displays the User ARN (arn:aws:iam::072398337363:user/PolSumUser), Path (/), and Creation time (2017-02-16 12:58 PDT). Below this, there are tabs for Permissions (highlighted with a red circle 1), Groups (1), Security credentials, and Access Advisor.

The Permissions tab shows an 'Add permissions' button and a message indicating 8 attached policies. A summary table lists the policy name (SummaryAllElements) and type (Managed policy). A warning message (red circle 3) states: "This policy defines some actions, resources, or conditions that do not provide permissions. To grant access, policies must have an action that has an applicable resource or condition. For details, choose Show remaining. Learn more".

Below the summary, there are buttons for Policy summary (red circle 4), JSON ({}), Edit policy, Simulate policy (red circle 5), and a Filter search bar (red circle 6). A large red box (red circle 7) highlights the main service list table, which includes columns for Service, Access level, Resource, and Request condition.

The table shows the following grants:

Service	Access level	Resource	Request condition
Unrecognized services			
codedploy	⚠️		
Explicit deny (1 of 103 services)			
S3	⚠️ Full: Read, Write, Permissions management Limited: List	Multiple	None
Allow (3 of 103 services) Show remaining 100			
Billing	Full: Read Limited: Write	All resources	Multiple
EC2	⚠️ None	All resources	None
S3	⚠️ Limited: Write, Permissions management	BucketName = developer_bucket, ObjectPath = All	s3:x-amz-acl = public-read

At the bottom of the table, there are links for EC2_Troubleshoot and Managed policy, along with a close button (X).

In the preceding image, the policy summary is visible from within the user details page:

1. The **Permissions** tab for a user includes the policies that are attached to the **PolSumUser** user.
2. The **SummaryAllElements** policy is one of several policies that are attached to the user. The policy is expanded in order to view the policy summary.
3. If the policy does not grant permissions to all the actions, resources, and conditions defined in the policy, then a warning or error banner appears at the top of the page. The policy summary then includes details about the problem. To learn how policy summaries help you to understand and troubleshoot the permissions that your policy grants, see the section called ["My Policy Does Not Grant the Expected Permissions" \(p. 554\)](#).
4. Use the **Policy summary** and **{ } JSON** buttons to toggle between the policy summary and the JSON policy document.
5. **Simulate policy** opens the policy simulator for testing the policy.
6. Use the search box to reduce the list of services and easily find a specific service.
7. The expanded view shows additional details of the **SummaryAllElements** policy.

The following policy summary table image shows the expanded **SummaryAllElements** policy on the **PolSumUser** user details page.

A Service	G Access level	H Resource	I Request condition
B Unrecognized services			
codeddeploy ⚠			
C Explicit deny (1 of 103 services)			
S3 ⚠	Full: Read, Write, Permissions management Limited: List	Multiple	None
D Allow (3 of 103 services) Show remaining 100			
Billing	Full: Read Limited: Write	All resources	Multiple
E EC2 ⚠	None	All resources	None
F S3 ⚠	Limited: Write, Permissions management	BucketName = developer_bucket, ObjectPath = All	s3:x-amz-acl = public-read

In the preceding image, the policy summary is visible from within the user details page:

- A. **Service** – This column lists the services that are defined within the policy and provides details for each service. Each service name in the policy summary table is a link to the *service summary* table, which is explained in [Service Summary \(List of Actions\) \(p. 502\)](#). In this example, permissions are defined for the Amazon S3, Billing, and Amazon EC2 services. The policy also defines permissions for a (misspelled) codedploy service, which IAM does not recognize.
- B. **Unrecognized services** – This policy includes an unrecognized service (in this case **codedploy** ⚠). You can use this warning to check whether a service name might include a typo. If the service name is correct, then the service might not support policy summaries, might be in preview, or might be a custom service. To request policy summary support for a generally available (GA) service, see [Service Does Not Support IAM Policy Summaries \(p. 553\)](#). In this example, the policy includes an unrecognized codedploy service that is missing an e. Because of this typo, the policy does not provide the expected AWS CodeDeploy permissions. You can [edit the policy \(p. 493\)](#) to include the accurate codedploy service name; the service then appears in the policy summary.
- C. For those services that IAM recognizes, it arranges services according to whether the policy allows or explicitly denies the use of the service. In this example, the policy includes **Allow** and **Deny** statements for the Amazon S3 service. Therefore the policy summary includes S3 within both the **Explicit deny** and **Allow** sections.
- D. **Show remaining 100** – Choose this link to expand the table to include the services that are not defined by the policy. These services are *implicitly denied* (or denied by default) within this policy. However, a statement in another policy might still allow or explicitly deny using the service. The policy summary summarizes the permissions of a single policy. To learn about how the AWS service decides whether a given request should be allowed or denied, see [Policy Evaluation Logic \(p. 636\)](#).
- E. **EC2 ⚠** – This service includes an unrecognized action. IAM recognizes service names, actions, and resource types for services that support policy summaries. When a service is recognized but contains an action that is not recognized, IAM includes a warning next to that service. In this example, IAM can't recognize at least one Amazon EC2 action. To learn more about unrecognized actions and to view the unrecognized action in an S3 service summary, see [Service Summary \(List of Actions\) \(p. 502\)](#).
- F. **S3 ⚠** – This service includes an unrecognized resource. IAM recognizes service names, actions, and resource types for services that support policy summaries. When a service is recognized but contains

Note

IAM reviews service names, actions, and resource types for services that support policy summaries. However, your policy summary might include a resource value or condition that does not exist. Always test your policies with the [policy simulator \(p. 448\)](#).

- F. **S3 ⚠** – This service includes an unrecognized resource. IAM recognizes service names, actions, and resource types for services that support policy summaries. When a service is recognized but contains

a resource type that is not recognized, IAM includes a warning next to that service. In this example, IAM can't recognize at least one Amazon S3 action. To learn more about unrecognized resources and to view the unrecognized resource type in an S3 service summary, see [Service Summary \(List of Actions\) \(p. 502\)](#).

G. Access level – This column tells whether the actions in each access level (List, Read, Write, and Permissions management) have Full or Limited permissions defined in the policy. For additional details and examples of the access level summary, see [Understanding Access Level Summaries Within Policy Summaries \(p. 501\)](#).

- **Full access** – This entry indicates that the service has access to all actions within all four of the access levels available for the service. In this example, because this row is in the **Explicit deny** section of the table, all Amazon S3 actions are denied for the resources included in the policy.
- If the entry does not include **Full access**, then the service has access to some but not all of the actions for the service. The access is then defined by following descriptions for each of the four access level classifications (List, Read, Write, and Permissions management):

Full: The policy provides access to all actions within each access level classification listed. In this example, the policy provides access to all of the Billing Read actions.

Limited: The policy provides access to one or more but not all actions within each access level classification listed. In this example, the policy provides access to some of the Billing Write actions.

H. Resource – This column shows the resources that the policy specifies for each service.

- **Multiple** – The policy includes more than one but not all of the resources within the service. In this example, access is explicitly denied to more than one Amazon S3 resource.
- **All resources** -- The policy is defined for all resources within the service. In this example, the policy allows the listed actions to be performed on all Billing resources.
- **Resource text** – The policy includes one resource within the service. In this example, the listed actions are allowed on only the developer_bucket Amazon S3 bucket resource. Depending on the information that the service provides to IAM, you might see an ARN such as arn:aws:s3:::developer_bucket/*, or you might see the defined resource type, such as BucketName = developer_bucket.

Note

This column can include a resource from a different service. If the policy statement that includes the resource does not include both actions and resources from the same service, then your policy includes mismatched resources. IAM does not warn you about mismatched resources when you create a policy, or when you view a policy in the policy summary. If this column includes a mismatched resource, then you should review your policy for errors. To better understand your policies, always test them with the [policy simulator \(p. 448\)](#).

I. Request condition – This column indicates whether the services or actions associated with the resource are subject to conditions.

- **None** – The policy includes no conditions for the service. In this example no conditions are applied to the denied actions in the Amazon S3 service.
- **Condition text** – The policy includes one condition for the service. In this example, the listed **Billing** actions are allowed only if the IP address of the source matches 203.0.113.0/24.
- **Multiple** – The policy includes more than one condition for the service. In this example, access to the listed Amazon S3 actions is allowed based on more than one condition. To view each of the multiple conditions for the policy, choose **{ } JSON** to view the policy document.

When a policy or an element within the policy does not grant permissions, IAM provides additional warnings and information in the policy summary. The following policy summary table shows the expanded **Show remaining 100** services on the **PolSumUser** user details page with the possible warnings.

Service	Access level	a Resource	b Request condition
Unrecognized services			
codeddeploy ⚠			
Explicit deny (1 of 103 services)			
S3 ⚠	Full: Read, Write, Permissions management Limited: List	Multiple ⚠ One or more actions do not have an applicable resource.	None
Allow (3 of 103 services) Hide remaining 100			
d ... e	None		
Billing	Full: Read Limited: Write	All resources	Multiple
CodeBuild	f ⚠ None - No actions are defined.	arn:aws:codebuild:us-east-1:123456789012:project/my-demo-project	None
CodeCommit	None	g ⚠ No resources are defined.	None
CodeDeploy	⚠ None - No actions are defined.	arn:aws:codedeploy:us-west-2:123456789012:deploymentgroup:*	None
EC2 ⚠	None	All resources	None
S3 ⚠	Limited: Write, Permissions management	BucketName = developer_bucket, ObjectPath = All ⚠ One or more resources do not have an applicable action.	s3:x-amz-acl = public-read ⚠ One or more conditions do not have an applicable action.

In the preceding image, you can see all services that include defined actions, resources, or conditions with no permissions:

- a. **Resource warnings** – For services that do not provide permissions for all of the included actions or resources, you see one of the following warnings in the **Resource** column of the table:

- ⚠ **No resources are defined.** – This means that the service has defined actions but no supported resources are included in the policy.
- ⚠ **One or more actions do not have an applicable resource.** – This means that the service has defined actions, but that some of those actions don't have a supported resource.
- ⚠ **One or more resources do not have an applicable action.** – This means that the service has defined resources, but that some of those resources don't have a supporting action.

If a service includes both actions that do not have an applicable resource and resources that do not have an applicable resource, then only the **One or more resources do not have an applicable action** warning is shown. This is because when you view the service summary for the service, resources that do not apply to any action are not shown. For the `ListAllMyBuckets` action, this policy includes the last warning because the action does not support resource-level permissions, and does not support the `s3:x-amz-acl` condition key. If you fix either the resource problem or the condition problem, the remaining issue appears in a detailed warning.

- b. **Request condition warnings** – For services that do not provide permissions for all of the included conditions, you see one of the following warnings in the **Request condition** column of the table:

- ⚠ **One or more actions do not have an applicable condition.** – This means that the service has defined actions, but that some of those actions don't have a supported condition.
- ⚠ **One or more conditions do not have an applicable action.** – This means that the service has defined conditions, but that some of those conditions don't have a supporting action.

- c. **Multiple |  One or more actions do not have an applicable resource.** – The Deny statement for Amazon S3 includes more than one resource. It also includes more than one action, and some actions support the resources and some do not. To view this policy, see [the section called "SummaryAllElements JSON Policy Document" \(p. 499\)](#). In this case, the policy includes all Amazon S3 actions, and only the actions that can be performed on a bucket or bucket object are denied.
- d. The ellipses (...) indicate that all the services are included in the page, but we are showing only the rows with information relevant to this policy. When you view this page in the AWS Management Console, you see all the AWS services.
- e. The background color in the table rows indicates services that do not grant any permissions. You cannot get any additional information about these services in the policy summary. For services in white rows, you can choose the name of the service to view the service summary (list of actions) page. There you can learn more about the permissions granted for that service.
- f. ** None - No actions are defined.** – This means that the service is defined as a resource or condition, but that no actions are included for the service, and therefore the service provides no permissions. In this case, the policy includes a CodeBuild resource but no CodeBuild actions.
- g. ** No resources are defined** – The service has defined actions, but no supported resources are included in the policy, and therefore the service provides no permissions. In this case, the policy includes CodeCommit actions but no CodeCommit resources.
- h. **BucketName = developer_bucket, ObjectPath = All |  One or more resources do not have an applicable action.** – The service has a defined bucket object resource, and at least one more resource that does not have a supporting action.
- i. **s3:x-amz-acl = public-read |  One or more conditions do not have an applicable action.** – The service has a defined s3:x-amz-acl condition key, and at least one more condition key that does not have a supporting action.

SummaryAllElements JSON Policy Document

The **SummaryAllElements** policy is not intended for you to use to define permissions in your account. Rather, it is included to demonstrate the errors and warnings that you might encounter while viewing a policy summary.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "aws-portal:ViewBilling",
                "aws-portal:ViewPaymentMethods",
                "aws-portal:ModifyPaymentMethods",
                "aws-portal:ViewAccount",
                "aws-portal:ModifyAccount",
                "aws-portal:ViewUsage"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "IpAddress": {
                    "aws:SourceIp": "203.0.113.0/24"
                }
            }
        },
    ]
}
```

```
{  
    "Effect": "Deny",  
    "Action": [  
        "s3:*"  
    ],  
    "Resource": [  
        "arn:aws:s3::::customer",  
        "arn:aws:s3::::customer/*"  
    ]  
},  
{  
    "Effect": "Allow",  
    "Action": [  
        "ec2:GetConsoleScreenshots"  
    ],  
    "Resource": [  
        "*"  
    ]  
},  
{  
    "Effect": "Allow",  
    "Action": [  
        "codedeploy:*",  
        "codecommit:*"  
    ],  
    "Resource": [  
        "arn:aws:codedeploy:us-west-2:123456789012:deploymentgroup:*,  
        "arn:aws:codebuild:us-east-1:123456789012:project/my-demo-project"  
    ]  
},  
{  
    "Effect": "Allow",  
    "Action": [  
        "s3>ListAllMyBuckets",  
        "s3GetObject",  
        "s3DeleteObject",  
        "s3PutObject",  
        "s3PutObjectAcl"  
    ],  
    "Resource": [  
        "arn:aws:s3::::developer_bucket",  
        "arn:aws:s3::::developer_bucket/*",  
        "arn:aws:autoscaling:us-east-2:123456789012:autoscalgrp"  
    ],  
    "Condition": {  
        "StringEquals": {  
            "s3:x-amz-acl": [  
                "public-read"  
            ],  
            "s3:prefix": [  
                "custom",  
                "other"  
            ]  
        }  
    }  
}  
]  
}
```

Understanding Access Level Summaries Within Policy Summaries

AWS Access Level Summary

Policy summaries include an access level summary that describes the action permissions defined for each service that is mentioned in the policy. To learn about policy summaries, see [Understanding Permissions Granted by a Policy \(p. 491\)](#). Access level summaries indicate whether the actions in each access level (List, Read, Write, and Permissions management) have Full or Limited permissions defined in the policy. To view the access level classification that is assigned to each action in a service, see [Actions, Resources, and Condition Keys for AWS Services \(p. 686\)](#).

The following example describes the access provided by a policy for the given services. For examples of full JSON policy documents and their related summaries, see [Examples of Policy Summaries \(p. 510\)](#).

Service	Access Level	This policy provides the following
IAM	Full access	Access to all actions within the IAM service.
CloudWatch	Full: List	Access to all CloudWatch actions in the List access level, but no access to actions with the Read, Write, or Permissions management access level classification.
Data Pipeline	Limited: List, Read	Access to at least one but not all AWS Data Pipeline actions in the List and Read access level, but not the Write or Permissions management actions.
EC2	Full: List, Read Limited: Write	Access to all Amazon EC2 List and Read actions and access to at least one but not all Amazon EC2 Write actions, but no access to actions with the Permissions management access level classification.
S3	Limited: Read, Write, Permissions management	Access to at least one but not all Amazon S3 Read, Write and Permissions management actions.
CodeDeploy	(empty)	Unknown access, because IAM does not recognize this service.
API Gateway	None	No access is defined in the policy.
CodeBuild	 No actions are defined.	No access because no actions are defined for the service. To learn how to understand and troubleshoot this issue, see the section called "My Policy Does Not Grant the Expected Permissions" (p. 554) .

As [previously mentioned \(p. 497\)](#), **Full access** indicates that the policy provides access to all the actions within the service. Policies that provide access to some but not all actions within a service are further grouped according to the access level classification. This is indicated by one of the following access-level groupings:

- **Full:** The policy provides access to all actions within the specified access level classification.

- **Limited:** The policy provides access to one or more but not all actions within the specified access level classification.
- **None:** The policy provides no access.
- (empty): IAM does not recognize this service. If the service name includes a typo, then the policy provides no access to the service. If the service name is correct, then the service might not support policy summaries or might be in preview. In this case, the policy might provide access, but that access cannot be shown in the policy summary. To request policy summary support for a generally available (GA) service, see [Service Does Not Support IAM Policy Summaries \(p. 553\)](#).

Access level summaries that include limited (partial) access to actions are grouped using the AWS access level classifications `List`, `Read`, `Write`, `Permissions Management`, or `Tagging`.

AWS Access Levels

AWS defines the following access level classifications for the actions in a service:

- **List:** Permission to list resources within the service to determine whether an object exists. Actions with this level of access can list objects but cannot see the contents of a resource. For example, the Amazon S3 action `ListBucket` has the **List** access level.
- **Read:** Permission to read but not edit the contents and attributes of resources in the service. For example, the Amazon S3 actions `GetObject` and `GetBucketLocation` have the **Read** access level.
- **Write:** Permission to create, delete, or modify resources in the service. For example, the Amazon S3 actions `CreateBucket`, `DeleteBucket` and `PutObject` have the **Write** access level. `Write` actions might also allow modifying a resource tag. However, an action that allows only changes to tags has the `Tagging` access level.
- **Permissions management:** Permission to grant or modify resource permissions in the service. For example, most IAM and AWS Organizations actions, as well as actions like the Amazon S3 actions `PutBucketPolicy` and `DeleteBucketPolicy` have the **Permissions management** access level.

Tip

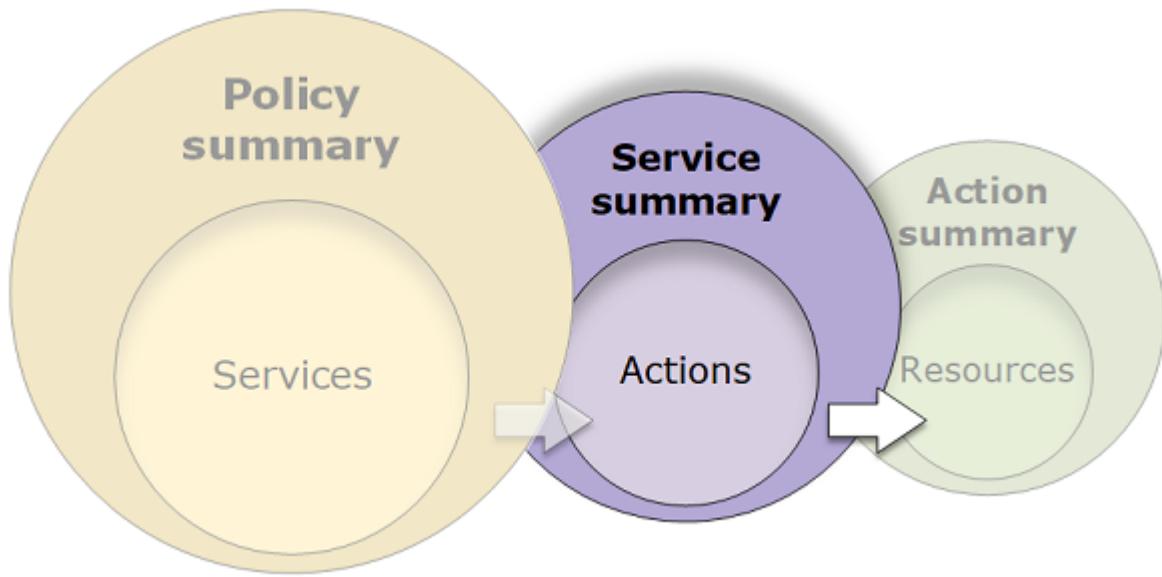
To improve the security of your AWS account, restrict or regularly monitor policies that include the **Permissions management** access level classification.

- **Tagging:** Permission to perform actions that only change the state of resource tags. For example, the IAM actions `TagRole` and `UntagRole` have the **Tagging** access level because they allow only tagging or untagging a role. However, the `CreateRole` action allows tagging a role resource when you create that role. Because the action does not only add a tag, it has the `Write` access level.

To view the access level classification for all of the actions in a service, see [Actions, Resources, and Condition Keys for AWS Services \(p. 686\)](#).

Service Summary (List of Actions)

Policies are summarized in three tables: the [policy summary \(p. 492\)](#), the service summary, and the [action summary \(p. 507\)](#). The *service summary* table includes a list of the actions and summaries of the permissions that are defined by the policy for the chosen service.



You can view a service summary for each service listed in the policy summary that grants permissions. The table is grouped into **Uncategorized actions**, **Uncategorized resource types**, and access level sections. If the policy includes an action that IAM does not recognize, then the action is included in the **Uncategorized actions** section of the table. If IAM recognizes the action, then it is included under one of the access level (**List**, **Read**, **Write** and **Permissions management**) sections of the table. To view the access level classification that is assigned to each action in a service, see [Actions, Resources, and Condition Keys for AWS Services \(p. 686\)](#).

Viewing Service Summaries

You can view the service summary for managed policies on the **Policies** page, or view service summaries for inline and managed policies attached to a user or role through the **Users** page and **Roles** page. However, if you choose a service name on the **Users** page or **Roles** page from a managed policy, you are redirected to the **Policies** page. Service summaries for managed policies must be viewed on the **Policies** page.

To view the service summary for a managed policy

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Policies**.
3. In the list of policies, choose the name of the policy that you want to view.
4. On the **Summary** page for the policy, view the **Permissions** tab to see the policy summary.
5. In the policy summary list of services, choose the name of the service that you want to view.

To view the service summary for a policy attached to a user

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Users**.
3. In the list of users, choose the name of the user whose policy you want to view.
4. On the **Summary** page for the user, view the **Permissions** tab to see the list of policies that are attached to the user directly or from a group.

5. In the table of policies for the user, expand the row of the policy that you want to view.
6. In the policy summary list of services, choose the name of the service that you want to view.

Note

If the policy that you select is an inline policy that is attached directly to the user, then the service summary table appears. If the policy is an inline policy attached from a group, then you are taken to the JSON policy document for that group. If the policy is a managed policy, then you are taken to the service summary for that policy on the **Policies** page.

To view the service summary for a policy attached to a role

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. Choose **Roles** from the navigation pane.
3. In the list of roles, choose the name of the role whose policy you want to view.
4. On the **Summary** page for the role, view the **Permissions** tab to see the list of policies that are attached to the role.
5. In the table of policies for the role, expand the row of the policy that you want to view.
6. In the policy summary list of services, choose the name of the service that you want to view.

Understanding the Elements of a Service Summary

The example below is the service summary for Amazon S3 actions that are allowed from the **SummaryAllElements** policy summary (see [the section called "SummaryAllElements JSON Policy Document" \(p. 499\)](#)). The actions for this service are grouped by **Uncategorized actions**, **Uncategorized resource types**, and access level. For example, two **Write** actions are defined out of the total 29 **Write** actions available for the service.

Action	Resource	Request condition
ListAllMyBuckets (No access)		None
GetObject (No access)	BucketName = developer_bucket, ObjectPath = All	! s3:x-amz-acl = public-read is not a supported condition key for this action.
PutObject	BucketName = developer_bucket, ObjectPath = All	s3:x-amz-acl = public-read
Permissions management (1 of 6 actions)		
PutObjectAcl	BucketName = developer_bucket, ObjectPath = All	s3:x-amz-acl = public-read

The service summary page for a managed policy includes the following information:

- If the policy does not grant permissions to all the actions, resources, and conditions defined for the service in the policy, then a warning banner appears at the top of the page. The service summary then includes details about the problem. To learn how policy summaries help you to understand and troubleshoot the permissions that your policy grants, see [the section called "My Policy Does Not Grant the Expected Permissions" \(p. 554\)](#).
- Next to the **Back** link appears the name of the service (in this case **S3**). The service summary for this service includes the list of allowed actions that are defined in the policy. If instead, the text **(Explicitly denied)** appears next to the name of a service, then the actions listed in the service summary table are explicitly denied.
- Choose **{ } JSON** to see additional details about the policy. You can do this to view all conditions that are applied to the actions. (If you are viewing the service summary for an inline policy that is attached directly to a user, you must close the service summary dialog box and return to the policy summary to access the JSON policy document.)
- To view the summary for a specific action, type keywords into the search box to reduce the list of available actions.
- Action (2 of 69 actions)** – This column lists the actions that are defined within the policy and provides the resources and conditions for each action. If the policy grants permissions to the action, then the action name links to the [action summary \(p. 507\)](#) table. The count indicates the number of recognized actions that provide permissions. The total is the number of known actions for the service. In this example, 2 actions provide permissions out of 69 total known S3 actions.
- Show/Hide remaining 67** – Choose this link to expand or hide the table to include actions that are known but do not provide permissions for this service. Expanding the link also displays warnings for any elements that do not provide permissions.

7. Unrecognized resource types – This policy includes at least one unrecognized resource type within the policy for this service. You can use this warning to check whether a resource type might include a typo. If the resource type is correct, then the service might not fully support policy summaries, might be in preview, or might be a custom service. To request policy summary support for a specific resource type in a generally available (GA) service, see [Service Does Not Support IAM Policy Summaries \(p. 553\)](#). In this example, the autoscaling service name is missing an a.

8. Unrecognized actions – This policy includes at least one unrecognized action within the policy for this service. You can use this warning to check whether an action might include a typo. If the action name is correct, then the service might not fully support policy summaries, might be in preview, or might be a custom service. To request policy summary support for a specific action in a generally available (GA) service, see [Service Does Not Support IAM Policy Summaries \(p. 553\)](#). In this example,

the DeleteObject  action is missing an e.

Note

IAM reviews service names, actions, and resource types for services that support policy summaries. However, your policy summary might include a resource value or condition that does not exist. Always test your policies with the [policy simulator \(p. 448\)](#).

9. For those actions that IAM recognizes, the table groups these actions into at least one or up to four sections, depending on the level of access that the policy allows or denies. The sections are **List**, **Read**, **Write**, and **Permissions management**. You can also see the number of actions that are defined out of the total number of actions available within each access level. To view the access level classification that is assigned to each action in a service, see [Actions, Resources, and Condition Keys for AWS Services \(p. 686\)](#).

10. The ellipses (...) indicate that all the actions are included in the page, but we are showing only the rows with information relevant to this policy. When you view this page in the AWS Management Console, you see all the actions for your service.

11 (No access) – This policy includes an action that does not provide permissions.

12 Actions that provide permissions include a link to the action summary.

13 Resource – This column shows the resources that the policy defines for the service. IAM does not check whether the resource applies to each action. In this example, actions in the S3 service are allowed on only the developer_bucket Amazon S3 bucket resource. Depending on the information that the service provides to IAM, you might see an ARN such as arn:aws:s3:::developer_bucket/*, or you might see the defined resource type, such as BucketName = developer_bucket.

Note

This column can include a resource from a different service. If the policy statement that includes the resource does not include both actions and resources from the same service, then your policy includes mismatched resources. IAM does not warn you about mismatched resources when you create a policy, or when you view a policy in the service summary.

IAM also does not indicate whether the action applies to the resources, only whether the service matches. If this column includes a mismatched resource, then you should review your policy for errors. To better understand your policies, always test them with the [policy simulator \(p. 448\)](#).

14 Resource warning – For actions with resources that do not provide full permissions, you see one of the following warnings:

- **This action does not support resource-level permissions. This requires a wildcard (*) for the resource.** – This means that the policy includes resource-level permissions but must include "Resource": ["*"] to provide permissions for this action.
- **This action does not have an applicable resource.** – This means that the action is included in the policy without a supported resource.
- **This action does not have an applicable resource and condition.** – This means that the action is included in the policy without a supported resource and without a supported condition. In this case, there is also condition included in the policy for this service, but there are no conditions that apply to this action.

For the `ListAllMyBuckets` action, this policy includes the last warning because the action does not support resource-level permissions and does not support the `s3:x-amz-acl` condition key. If you fix either the resource problem or the condition problem, the remaining issue appears in a detailed warning.

15Request condition – This column tells whether the actions associated with the resource are subject to conditions. To learn more about those conditions, choose `{ } JSON` to review the JSON policy document.

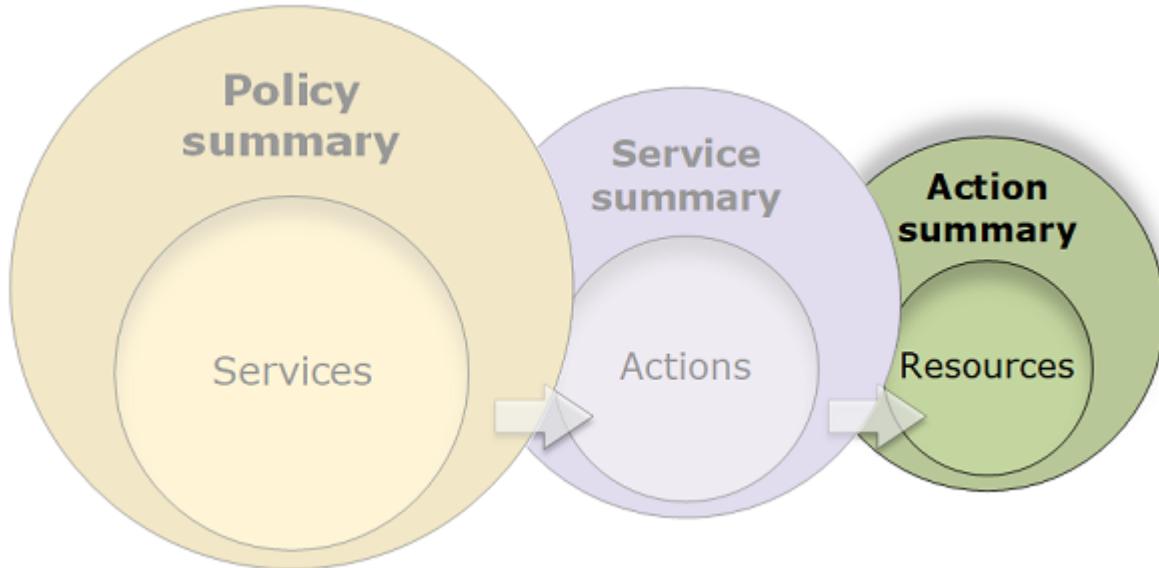
16Condition warning – For actions with conditions that do not provide full permissions, you see one of the following warnings:

- **<CONDITION_KEY> is not a supported condition key for this action.** – This means that the policy includes a condition key for the service that is not supported for this action.
- **Multiple condition keys are not supported for this action.** – This means that the policy includes more than one condition keys for the service that are not supported for this action.

For `GetObject`, this policy includes the `s3:x-amz-acl` condition key, which will not work with this action. Although the action supports the resource, the policy does not grant any permissions for this action because the condition will never be true for this action.

Action Summary (List of Resources)

Policies are summarized in three tables: the [policy summary](#) (p. 492), the [service summary](#) (p. 502), and the action summary. The *action summary* table includes a list of resources and the associated conditions that apply to the chosen action.



To view an action summary for each action that grants permissions, choose the link in the service summary. The action summary table includes details about the resource, including its **Region** and **Account**. You can also view the conditions that apply to each resource. This shows you conditions that apply to some resources but not others.

Viewing Action Summaries

You can view the action summary for any policy that is attached to a user on the **Users** page. You can view the action summary for any policy that is attached to a role on the **Roles** page. You can view

the action summary for managed policies on the **Policies** page. However, if you try to view the action summary for a managed policy from the **Users** page or the **Roles** page, you are redirected to the **Policies** page.

To view the action summary for a managed policy

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Policies**.
3. In the list of policies, choose the name of the policy that you want to view.
4. On the **Summary** page for the policy, view the **Permissions** tab to see the policy summary.
5. In the policy summary list of services, choose the name of the service that you want to view.
6. In the service summary list of actions, choose the name of the action that you want to view.

To view the action summary for a policy attached to a user

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. Choose **Users** from the navigation pane.
3. In the list of users, choose the name of the user whose policy you want to view.
4. On the **Summary** page for the user, view the **Permissions** tab to see the list of policies that are attached to the user directly or from a group.
5. In the table of policies for the user, expand the row of the policy that you want to view.
6. In the policy summary list of services, choose the name of the service that you want to view.

Note

If the policy that you select is an inline policy that is attached directly to the user, then the service summary table appears. If the policy is an inline policy attached from a group, then you are taken to the JSON policy document for that group. If the policy is a managed policy, then you are taken to the service summary for that policy on the **Policies** page.

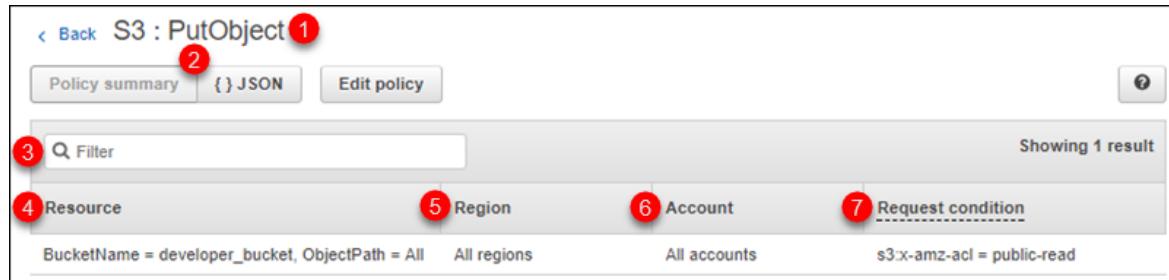
7. In the service summary list of actions, choose the name of the action that you want to view.

To view the action summary for a policy attached to a role

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Roles**.
3. In the list of roles, choose the name of the role whose policy you want to view.
4. On the **Summary** page for the role, view the **Permissions** tab to see the list of policies that are attached to the role.
5. In the table of policies for the role, expand the row of the policy that you want to view.
6. In the policy summary list of services, choose the name of the service that you want to view.
7. In the service summary list of actions, choose the name of the action that you want to view.

Understanding the Elements of an Action Summary

The example below is the action summary for the `PutObject` (Write) action from the Amazon S3 service summary (see [Service Summary \(List of Actions\) \(p. 502\)](#)). For this action, the policy defines multiple conditions on a single resource.



The action summary page includes the following information:

1. Next to the **Back** link appears the name of the service and action in the format **service: action** (in this case **S3: PutObject**). The action summary for this service includes the list of resources that are defined in the policy.
2. Choose **{ } JSON** to see additional details about the policy, such as viewing the multiple conditions that are applied to the actions. (If you are viewing the action summary for an inline policy that is attached directly to a user, the steps differ. To access the JSON policy document in that case, you must close the action summary dialog box and return to the policy summary.)
3. To view the summary for a specific resource, type keywords into the search box to reduce the list of available resources.
4. **Resource** – This column lists the resources that the policy defines for the chosen service. In this example, the **PutObject** action is allowed on all object paths, but on only the `developer_bucket` Amazon S3 bucket resource. Depending on the information that the service provides to IAM, you might see an ARN such as `arn:aws:s3:::developer_bucket/*`, or you might see the defined resource type, such as `BucketName = developer_bucket, ObjectPath = All`.
5. **Region** – This column shows the Region in which the resource is defined. Resources can be defined for all Regions, or a single Region. They cannot exist in more than one specific Region.
 - **All Regions** – The actions that are associated with the resource apply to all Regions. In this example, the action belongs to a global service, Amazon S3. Actions that belong to global services apply to all Regions.
 - Region text – The actions associated with the resource apply to one Region. For example, a policy can specify the `us-east-2` Region for a resource.
6. **Account** – This column indicates whether the services or actions associated with the resource apply to a specific account. Resources can exist in all accounts or a single account. They cannot exist in more than one specific account.
 - **All accounts** – The actions that are associated with the resource apply to all accounts. In this example, the action belongs to a global service, Amazon S3. Actions that belong to global services apply to all accounts.
 - **This account** – The actions that are associated with the resource apply only to the account that you are currently logged in to.
 - Account number – The actions that are associated with the resource apply to one account (one that you are not currently logged in to). For example, if a policy specifies the `123456789012` account for a resource, then the account number appears in the policy summary.
7. **Request condition** – This column shows whether the actions that are associated with the resource are subject to conditions. This example includes the `s3:x-amz-acl = public-read` condition. To learn more about those conditions, choose **{ } JSON** to review the JSON policy document.

Examples of Policy Summaries

The following examples include JSON policies with their associated [policy summaries](#) (p. 492), the [service summaries](#) (p. 502), and the [action summaries](#) (p. 507) to help you understand the permissions given through a policy.

Policy 1: DenyCustomerBucket

This policy demonstrates an allow and a deny for the same service.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "FullAccess",  
            "Effect": "Allow",  
            "Action": ["s3:*"],  
            "Resource": ["*"]  
        },  
        {  
            "Sid": "DenyCustomerBucket",  
            "Action": ["s3:*"],  
            "Effect": "Deny",  
            "Resource": ["arn:aws:s3:::customer", "arn:aws:s3:::customer/*"]  
        }  
    ]  
}
```

DenyCustomerBucket Policy Summary:

This policy defines some actions, resources, or conditions that do not provide permissions. To grant access, policies must have an action that has an applicable resource or condition. For details, choose Show remaining. [Learn more](#)

Service	Access level	Resource	Request condition
S3	Full: Read, Write, Permissions management Limited: List	Multiple	None
Allow (1 of 103 services) Show remaining 102			
S3	Full access	All resources	None

DenyCustomerBucket S3 (Explicit deny) Service Summary:

Action (66 of 69) Hide remaining 3	Resource	Request condition
List (1 of 4 actions)		
HeadBucket (No access)	⚠ This action does not support resource-level permissions. This requires a wildcard (*) for the resource.	None
ListAllMyBuckets(No access)	⚠ This action does not support resource-level permissions. This requires a wildcard (*) for the resource.	None
ListBucket	BucketName = customer	None
ListObjects (No access)	⚠ This action does not support resource-level permissions. This requires a wildcard (*) for the resource.	None
Read (30 of 30 actions)		
GetAccelerateConfiguration	BucketName = customer	None
GetAnalyticsConfiguration	BucketName = customer	None
GetBucketAcl	BucketName = customer	None
GetBucketCORS	BucketName = customer	None
GetBucketLocation	BucketName = customer	None
GetBucketLogging	BucketName = customer	None
GetBucketNotification	BucketName = customer	None
GetBucketPolicy	BucketName = customer	None
GetBucketRequestPayment	BucketName = customer	None
GetBucketTagging	BucketName = customer	None
GetBucketVersioning	BucketName = customer	None
GetBucketWebsite	BucketName = customer	None
GetInventoryConfiguration	BucketName = customer	None
GetIpConfiguration	BucketName = customer	None
GetLifecycleConfiguration	BucketName = customer	None
GetMetricsConfiguration	BucketName = customer	None
GetObject	BucketName = customer, ObjectPath = All	None
GetObjectAcl	BucketName = customer, ObjectPath = All	None
GetObjectTagging	BucketName = customer, ObjectPath = All	None
GetObjectTorrent	BucketName = customer, ObjectPath = All	None
GetObjectVersion	BucketName = customer, ObjectPath = All	None

GetObject (Read) Action Summary:

Resource	Region	Account	Request condition
BucketName = customer, ObjectPath = All	All regions	All accounts	None

Policy 2: DynamoDbRowCognitoID

This policy provides row-level access to Amazon DynamoDB based on the user's Amazon Cognito ID.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "dynamodb>DeleteItem",
                "dynamodb:GetItem",
                "dynamodb:PutItem",
                "dynamodb:UpdateItem"
            ],
            "Resource": [
                "arn:aws:dynamodb:us-west-1:123456789012:table/myDynamoTable"
            ],
            "Condition": {
                "ForAllValues:StringEquals": {
                    "dynamodb:LeadingKeys": [
                        "${cognito-identity.amazonaws.com:sub}"
                    ]
                }
            }
        }
    ]
}
```

DynamoDbRowCognitoID Policy Summary:

Service	Access level	Resource	Request condition
Allow (1 of 102 services) Show remaining 101			
DynamoDB	Limited: Read, Write	TableName = myDynamoTable	dynamodb:LeadingKeys = \${cognito-identity.amazonaws.com:sub}

DynamoDbRowCognitoID DynamoDB (Allow) Service Summary:

Action (4 of 25) Show remaining 21	Resource	Request condition
Read (1 of 14 actions)		
GetItem	TableName = myDynamoTable	dynamodb:LeadingKeys = \${cognito-identity.amazonaws.com:sub}
Write (3 of 10 actions)		
DeleteItem	TableName = myDynamoTable	dynamodb:LeadingKeys = \${cognito-identity.amazonaws.com:sub}
PutItem	TableName = myDynamoTable	dynamodb:LeadingKeys = \${cognito-identity.amazonaws.com:sub}
UpdateItem	TableName = myDynamoTable	dynamodb:LeadingKeys = \${cognito-identity.amazonaws.com:sub}

GetItem (List) Action Summary:

Resource	Region	Account	Request
TableName = myDynamoTable	us-west-1	123456789012	dynamodb:LeadingKeys = \${cognito-identity.amazonaws.com:sub}

Policy 3: MultipleResourceCondition

This policy includes multiple resources and conditions.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:PutObjectAcl"
            ],
            "Resource": ["arn:aws:s3:::Apple_bucket/*"],
            "Condition": {"StringEquals": {"s3:x-amz-acl": ["public-read"]}}
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:PutObjectAcl"
            ],
            "Resource": ["arn:aws:s3:::Orange_bucket/*"],
            "Condition": {"StringEquals": {
                "s3:x-amz-acl": ["custom"],
                "s3:x-amz-grant-full-control": ["1234"]
            }}
        }
    ]
}
```

MultipleResourceCondition Policy Summary:

Service	Access level	Resource	Request condition
Allow (1 of 100 services) Show remaining 99			
S3	Limited: Write, Permissions management	Multiple	Multiple

MultipleResourceCondition S3 (Allow) Service Summary:

Action (2 of 52 actions)	Show remaining 50	Resource	Request condition
Write (1 of 21 actions)			
PutObject		Multiple	Multiple
Permissions management (1 of 5 actions)			
PutObjectAcl		Multiple	Multiple

PutObject (Write) Action Summary:

Resource	Region	Account	Request condition
BucketName = Orange_bucket, ObjectPath = All	All regions	All accounts	Multiple
BucketName = Apple_bucket, ObjectPath = All	All regions	All accounts	s3:x-amz-acl = public-read

Policy 4: EC2_Troubleshoot

The following policy allows users to get a screenshot of a running Amazon EC2 instance, which can help with EC2 troubleshooting. This policy also permits viewing information about the items in the Amazon S3 developer bucket.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:GetConsoleScreenshot"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3>ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::developer"
            ]
        }
    ]
}
```

EC2_Troubleshoot Policy Summary:

Service	Access level	Resource	Request condition
Allow (2 of 102 services) Show remaining 100			
EC2	Limited: Read	All resources	None
S3	Limited: List	BucketName = developer	None

EC2_Troubleshoot S3 (Allow) Service Summary:

Action (1 of 52) Show remaining 51	Resource	Request condition
List (1 of 4 actions)		
ListBucket	BucketName = developer	None

ListBucket (List) Action Summary:

<input type="text"/> Filter			
Resource	Region	Account	Request
BucketName = developer	All regions	All accounts	None

Policy 5: Unrecognized_Service_Action

The following policy was intended to provide full access to DynamoDB, but that access fails because dynamodb is misspelled as dynamobd. This policy was intended to allow access to some Amazon EC2 actions in the us-east-2 Region, but deny that access to the ap-northeast-2 Region. However,

access to reboot instances in the `ap-northeast-2` Region is not explicitly denied because of the unrecognized `o` in the middle of the `RebootInstances` action. This example shows how you can use policy summaries to locate errors in your policies. To learn how to edit policies based on information in a policy summary, see [Editing Policies to Fix Warnings \(p. 493\)](#).

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "dynamodb:*"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Action": [
                "ec2:RunInstances",
                "ec2:StartInstances",
                "ec2:StopInstances",
                "ec2:RebootInstances"
            ],
            "Resource": "*",
            "Effect": "Deny",
            "Condition": {
                "StringEquals": {
                    "ec2:Region": "ap-northeast-2"
                }
            }
        },
        {
            "Action": [
                "ec2:RunInstances",
                "ec2:StartInstances",
                "ec2:StopInstances",
                "ec2:RebootInstances"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Condition": {
                "StringEquals": {
                    "ec2:Region": "us-east-2"
                }
            }
        }
    ]
}
```

Unrecognized_Service_Action Policy Summary:

Service	Access level	Resource	Request condition
<u>Unrecognized services</u>			
dynamodb	⚠		
<u>Explicit deny (1 of 103 services)</u>			
EC2	⚠ Limited: Write	All resources	ec2:Region = ap-northeast-2
<u>Allow (1 of 103 services)</u> Show remaining 102			
EC2	⚠ Limited: Write	All resources	ec2:Region = us-east-2

Unrecognized_Service_Action EC2 (Explicit deny) Service Summary:

Action (3 of 229) Show remaining 226	Resource	Request condition
Unrecognized actions		
RebootInstances ⚠		
Write (3 of 157 actions)		
RunInstances	All resources	ec2:Region = ap-northeast-2
StartInstances	All resources	ec2:Region = ap-northeast-2
StopInstances	All resources	ec2:Region = ap-northeast-2

Unrecognized_Service_Action StartInstances (Write) Action Summary:

Resource	Region	Account	Request condition
All resources	All regions	All accounts	ec2:Region = ap-northeast-2

Policy 6: CodeBuild_CodeCommit_CodeDeploy

This policy provides access to specific CodeBuild, CodeCommit, and CodeDeploy resources. Because these resources are specific to each service, they appear only with the matching service. If you include a resource that does not match any services in the `Action` element, then the resource appears in all action summaries.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Stmt1487980617000",
            "Effect": "Allow",
            "Action": [
                "codebuild:*",
                "codecommit:*",
                "codedeploy:*"
            ],
            "Resource": [
                "arn:aws:codebuild:us-east-2:123456789012:project/my-demo-project",
                "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo",
                "arn:aws:codedeploy:us-east-2:123456789012:application:WordPress_App",
                "arn:aws:codedeploy:us-east-2:123456789012:instance/AssetTag*"
            ]
        }
    ]
}
```

CodeBuild_CodeCommit_CodeDeploy Policy Summary:

Service	Access level	Resource	Request condition
Allow (3 of 103 services) Show remaining 100			
CodeBuild	Limited: List, Read, Write	arn:aws:codebuild:us-east-2:123456789012:project/my-demo-project	None
CodeCommit	Full: Read, Write Limited: List	arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo	None
CodeDeploy	Limited: List, Read, Write	Multiple	None

CodeBuild_CodeCommit_CodeDeploy CodeBuild (Allow) Service Summary:

Action (9 of 15) Show remaining 6	Resource	Request condition
List (1 of 3 actions)		
ListBuildsForProject	arn:aws:codebuild:us-east-2:123456789012:project/my-demo-project	None
Read (2 of 5 actions)		
BatchGetBuilds	arn:aws:codebuild:us-east-2:123456789012:project/my-demo-project	None
BatchGetProjects	arn:aws:codebuild:us-east-2:123456789012:project/my-demo-project	None
Write (6 of 7 actions)		
BatchDeleteBuilds	arn:aws:codebuild:us-east-2:123456789012:project/my-demo-project	None
CreateProject	arn:aws:codebuild:us-east-2:123456789012:project/my-demo-project	None
DeleteProject	arn:aws:codebuild:us-east-2:123456789012:project/my-demo-project	None
StartBuild	arn:aws:codebuild:us-east-2:123456789012:project/my-demo-project	None
StopBuild	arn:aws:codebuild:us-east-2:123456789012:project/my-demo-project	None
UpdateProject	arn:aws:codebuild:us-east-2:123456789012:project/my-demo-project	None

CodeBuild_CodeCommit_CodeDeploy StartBuild (Write) Action Summary:

Resource	Region	Account	Request condition
arn:aws:codebuild:us-east-2:123456789012:project/my-demo-project	us-east-2	123456789012	None

Permissions Required to Access IAM Resources

Resources are objects within a service. IAM resources include groups, users, roles, and policies. If you are signed in with AWS account root user credentials, you have no restrictions on administering IAM.

credentials or IAM resources. However, IAM users must explicitly be given permissions to administer credentials or IAM resources. You can do this by attaching an identity-based policy to the user.

Note

Throughout the AWS documentation, when we refer to an IAM policy without mentioning any of the specific categories, we mean an identity-based, customer managed policy. For details about policy categories, see [the section called "Policies & Permissions" \(p. 356\)](#).

Permissions for Administering IAM Identities

The permissions that are required to administer IAM groups, users, roles, and credentials usually correspond to the API actions for the task. For example, in order to create IAM users, you must have the `iam:CreateUser` permission that has the corresponding API command: `CreateUser`. To allow an IAM user to create other IAM users, you could attach an IAM policy like the following one to that user:

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": "iam:CreateUser",  
            "Resource": "*"  
        }  
    ]  
}
```

In a policy, the value of the `Resource` element depends on the action and what resources the action can affect. In the preceding example, the policy allows a user to create any user (* is a wildcard that matches all strings). In contrast, a policy that allows users to change only their own access keys (API actions `CreateAccessKey` and `UpdateAccessKey`) typically has a `Resource` element. In this case the ARN includes a variable (`#{aws:username}`) that resolves to the current user's name, as in the following example:

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "ListUsersForConsole",  
            "Effect": "Allow",  
            "Action": "iam>ListUsers",  
            "Resource": "arn:aws:iam::*:/*"  
        },  
        {  
            "Sid": "ViewAndUpdateAccessKeys",  
            "Effect": "Allow",  
            "Action": [  
                "iam:UpdateAccessKey",  
                "iam>CreateAccessKey",  
                "iam>ListAccessKeys"  
            ],  
            "Resource": "arn:aws:iam:::user/${aws:username}"  
        }  
    ]  
}
```

In the previous example, `#{aws:username}` is a variable that resolves to the user name of the current user. For more information about policy variables, see [IAM Policy Elements: Variables and Tags \(p. 628\)](#).

Using a wildcard character (*) in the action name often makes it easier to grant permissions for all the actions related to a specific task. For example, to allow users to perform any IAM action, you can use `iam:*` for the action. To allow users to perform any action related just to access keys, you can use `iam:AccessKey*` in the `Action` element of a policy statement. This gives the user permission to

perform the [CreateAccessKey](#), [DeleteAccessKey](#), [GetAccessKeyLastUsed](#), [ListAccessKeys](#), and [UpdateAccessKey](#) actions. (If an action is added to IAM in the future that has "AccessKey" in the name, using `iam:*AccessKey*` for the Action element will also give the user permission to that new action.) The following example shows a policy that allows users to perform all actions pertaining to their own access keys (replace ACCOUNT-ID-WITHOUT-HYPHENS with your AWS account ID):

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": "iam:*AccessKey*",  
            "Resource": "arn:aws:iam::ACCOUNT-ID-WITHOUT-HYPHENS:user/${aws:username}"  
        }  
    ]  
}
```

Some tasks, such as deleting a group, involve multiple actions: You must first remove users from the group, then detach or delete the group's policies, and then actually delete the group. If you want a user to be able to delete a group, you must be sure to give the user permissions to perform all of the related actions.

Permissions for Working in the AWS Management Console

The preceding examples show policies that allow a user to perform the actions with the [AWS CLI](#) or the [AWS SDKs](#).

As users work with the console, the console issues requests to IAM to list groups, users, roles, and policies, and to get the policies associated with a group, user, or role. The console also issues requests to get AWS account information and information about the principal. The principal is the user making requests in the console.

In general, to perform an action, you must have only the matching action included in a policy. To create a user, you need permission to call the `CreateUser` action. Often, when you use the console to perform an action, you must have permissions to display, list, get, or otherwise view resources in the console. This is necessary so that you can navigate through the console to make the specified action. For example, if user Jorge wants to use the console to change his own access keys, he goes to the IAM console and chooses **Users**. This action causes the console to make a `ListUsers` request. If Jorge doesn't have permission for the `iam>ListUsers` action, the console is denied access when it tries to list users. As a result, Jorge can't get to his own name and to his own access keys, even if he has permissions for the `CreateAccessKey` and `UpdateAccessKey` actions.

For example, if user Bob wants to use the console to change his own access keys, he goes to the IAM console and chooses **Users**. This action causes the console to make a `ListUsers` request. If Bob doesn't have permission for the `iam>ListUsers` action, the console is denied access when it tries to list users. As a result, Bob can't get to his own name and to his own access keys, even if he has permissions for the `CreateAccessKey` and `UpdateAccessKey` actions.

If you want to give users permissions to administer groups, users, roles, policies, and credentials with the AWS Management Console, you need to include permissions for the actions that the console performs. For some examples of policies that you can use to grant a user for these permissions, see [Example Policies for Administering IAM Resources \(p. 521\)](#).

Granting Permissions Across AWS Accounts

You can directly grant IAM users in your own account access to your resources. If users from another account need access to your resources, you can create an IAM role, which is an entity that includes

permissions but that isn't associated with a specific user. Users from other accounts can then use the role and access resources according to the permissions you've assigned to the role. For more information, see [Providing Access to an IAM User in Another AWS Account That You Own \(p. 179\)](#).

Note

Some services support resource-based policies as described in [Identity-Based Policies and Resource-Based Policies \(p. 379\)](#) (such as Amazon S3, Amazon SNS, and Amazon SQS). For those services, an alternative to using roles is to attach a policy to the resource (bucket, topic, or queue) that you want to share. The resource-based policy can specify the AWS account that has permissions to access the resource.

Permissions for One Service to Access Another

Many AWS services access other AWS services. For example, several AWS services—including Amazon EMR, Elastic Load Balancing, and Amazon EC2 Auto Scaling—manage Amazon EC2 instances. Other AWS services make use of Amazon S3 buckets, Amazon SNS topics, Amazon SQS queues, and so on.

The scenario for managing permissions in these cases varies by service. Here are some examples of how permissions are handled for different services:

- In Amazon EC2 Auto Scaling, users must have permission to use Auto Scaling, but don't need to be explicitly granted permission to manage Amazon EC2 instances.
- In AWS Data Pipeline, an IAM role determines what a pipeline can do; users need permission to assume the role. (For details, see [Granting Permissions to Pipelines with IAM](#) in the *AWS Data Pipeline Developer Guide*.)

For details about how to configure permissions properly so that an AWS service is able to accomplish the tasks you intend, refer to the documentation for the service you are calling. To learn how to create a role for a service, see [Creating a Role to Delegate Permissions to an AWS Service \(p. 236\)](#).

Configuring a service with an IAM role to work on your behalf

When you want to configure an AWS service to work on your behalf, you typically provide the ARN for an IAM role that defines what the service is allowed to do. AWS checks to ensure that you have permissions to pass a role to a service. For more information, see [Granting a User Permissions to Pass a Role to an AWS Service \(p. 258\)](#).

Required Actions

Actions are the things that you can do to a resource, such as viewing, creating, editing, and deleting that resource. Actions are defined by each AWS service.

To allow someone to perform an action, you must include the necessary actions in a policy that applies to the calling identity or the affected resource. In general, to provide the permission required to perform an action, you must include that action in your policy. For example, to create a user, you need add the `CreateUser` action to your policy.

In some cases, an action might require that you include additional related actions in your policy. For example, to provide permission for someone to create a directory in AWS Directory Service using the `ds:CreateDirectory` operation, you must include the following actions in their policy:

- `ds:CreateDirectory`
- `ec2:DescribeSubnets`
- `ec2:DescribeVpcs`
- `ec2:CreateSecurityGroup`

- `ec2:CreateNetworkInterface`
- `ec2:DescribeNetworkInterfaces`
- `ec2:AuthorizeSecurityGroupIngress`
- `ec2:AuthorizeSecurityGroupEgress`

When you create or edit a policy using the visual editor, you receive warnings and prompts to help you choose all of the required actions for your policy.

For more information about the permissions required to create a directory in AWS Directory Service, see [Example 2: Allow a User to Create a Directory](#).

Example Policies for Administering IAM Resources

Following are examples of IAM policies that allow users to perform tasks associated with managing IAM users, groups, and credentials. This includes policies that permit users manage their own passwords, access keys, and multi-factor authentication (MFA) devices.

For examples of policies that let users perform tasks with other AWS services, like Amazon S3, Amazon EC2, and DynamoDB, see [Example IAM Identity-Based Policies \(p. 394\)](#).

Topics

- [Allow a User to List the Account's Groups, Users, Policies, and More for Reporting Purposes \(p. 521\)](#)
- [Allow a User to Manage a Group's Membership \(p. 521\)](#)
- [Allow a User to Manage IAM Users \(p. 521\)](#)
- [Allow Users to Set Account Password Policy \(p. 523\)](#)
- [Allow Users to Generate and Retrieve IAM Credential Reports \(p. 523\)](#)
- [Allow All IAM Actions \(Admin Access\) \(p. 523\)](#)

Allow a User to List the Account's Groups, Users, Policies, and More for Reporting Purposes

The following policy allows the user to call any IAM action that starts with the string `Get` or `List`, and to generate reports. To view the example policy, see [IAM: Allows Read-Only Access to the IAM Console \(p. 427\)](#).

Allow a User to Manage a Group's Membership

The following policy allows the user to update the membership of the group called `MarketingGroup`. To view the example policy, see [IAM: Allows Managing a Group's Membership Programmatically and in the Console \(p. 425\)](#).

Allow a User to Manage IAM Users

The following policy allows a user to perform all the tasks associated with managing IAM users but not to perform actions on other entities, such as creating groups or policies. Allowed actions include these:

- Creating the user (the `CreateUser` action).
- Deleting the user. This task requires permissions to perform all of the following actions: `DeleteSigningCertificate`, `DeleteLoginProfile`, `RemoveUserFromGroup`, and `DeleteUser`.

- Listing users in the account and in groups (the [GetUser](#), [ListUsers](#) and [ListGroupsForUser](#) actions).
- Listing and removing policies for the user (the [ListUserPolicies](#), [ListAttachedUserPolicies](#), [DetachUserPolicy](#), [DeleteUserPolicy](#) actions)
- Renaming or changing the path for the user (the [UpdateUser](#) action). The `Resource` element must include an ARN that covers both the source path and the target path. For more information on paths, see [Friendly Names and Paths \(p. 575\)](#).

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowUsersToPerformUserActions",
            "Effect": "Allow",
            "Action": [
                "iam>ListPolicies",
                "iam>GetPolicy",
                "iam>UpdateUser",
                "iam>AttachUserPolicy",
                "iam>ListEntitiesForPolicy",
                "iam>DeleteUserPolicy",
                "iam>DeleteUser",
                "iam>ListUserPolicies",
                "iam>CreateUser",
                "iam>RemoveUserFromGroup",
                "iam>AddUserToGroup",
                "iam> GetUserPolicy",
                "iam>ListGroupsForUser",
                "iam>PutUserPolicy",
                "iam>ListAttachedUserPolicies",
                "iam>ListUsers",
                "iam> GetUser",
                "iam>DetachUserPolicy"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowUsersToSeeStatsOnIAMConsoleDashboard",
            "Effect": "Allow",
            "Action": [
                "iam>GetAccount*",
                "iam>ListAccount*"
            ],
            "Resource": "*"
        }
    ]
}
```

A number of the permissions included in the preceding policy allow the user to perform tasks in the AWS Management Console. Users who perform user-related tasks from the [AWS CLI](#), the [AWS SDKs](#), or the IAM HTTP query API only might not need certain permissions. For example, if users already know the ARN of policies to detach from a user, they do not need the `iam>ListAttachedUserPolicies` permission. The exact list of permissions that a user requires depends on the tasks that the user must perform while managing other users.

The following permissions in the policy allow access to user tasks via the AWS Management Console:

- `iam>GetAccount*`
- `iam>ListAccount*`

Allow Users to Set Account Password Policy

You might give some users permissions to get and update your AWS account's [password policy \(p. 101\)](#). To view the example policy, see [IAM: Allows Setting the Account Password Requirements Programmatically and in the Console \(p. 429\)](#).

Allow Users to Generate and Retrieve IAM Credential Reports

You can give users permission to generate and download a report that lists all users in your AWS account. The report also lists the status of various user credentials, including passwords, access keys, MFA devices, and signing certificates. For more information about credential reports, see [Getting Credential Reports for Your AWS Account \(p. 156\)](#). To view the example policy, see [IAM: Generate and Retrieve IAM Credential Reports \(p. 424\)](#).

Allow All IAM Actions (Admin Access)

You might give some users administrative permissions to perform all actions in IAM, including managing passwords, access keys, MFA devices, and user certificates. The following example policy grants these permissions.

Warning

When you give a user full access to IAM, there is no limit to the permissions that user can grant to him/herself or others. The user can create new IAM entities (users or roles) and grant those entities full access to all resources in your AWS account. When you give a user full access to IAM, you are effectively giving them full access to all resources in your AWS account. This includes access to delete all resources. You should grant these permissions to only trusted administrators, and you should enforce multi-factor authentication (MFA) for these administrators.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": "iam:*",  
            "Resource": "*"  
        }  
    ]  
}
```

What Is IAM Access Analyzer?

AWS IAM Access Analyzer helps you identify the resources in your account, such as Amazon S3 buckets or IAM roles, that are shared with an external entity. This lets you identify unintended access to your resources and data, which is a security risk. Access Analyzer identifies resources that are shared with external principals by using logic-based reasoning to analyze the resource-based policies in your AWS environment. For each instance of a resource that is shared outside of your account, Access Analyzer generates a finding. Findings include information about the access and the external principal that it is granted to. You can review findings to determine whether the access is intended and safe, or the access is unintended and a security risk.

Note

An external entity can be another AWS account, a root user, an IAM user or role, a federated user, an AWS service, an anonymous user, or other entity that you can use to create a filter. For more information, see [AWS JSON Policy Elements: Principal](#).

When you enable Access Analyzer, you create an analyzer for your account. Your account is the zone of trust for the analyzer. The analyzer monitors all of the [supported resources \(p. 525\)](#) within your zone of trust. Any access to resources by principals that are within your zone of trust is considered trusted. Once enabled, Access Analyzer analyzes the policies applied to all of the supported resources in your account. After the first analysis, Access Analyzer analyzes these policies periodically. If a new policy is added, or an existing policy is changed, Access Analyzer analyzes the new or updated policy within about 30 minutes.

When analyzing the policies, if Access Analyzer identifies one that grants access to an external principal that isn't within your zone of trust, it generates a finding. Each finding includes details about the resource, the external entity that has access to it, and the permissions granted so that you can take appropriate action. You can view the details included in the finding to determine whether the resource access is intentional or a potential risk that you should resolve. When you add a policy to a resource, or update an existing policy, Access Analyzer analyzes the policy. Access Analyzer also analyzes all resource-based policies periodically.

On rare occasions under certain conditions, Access Analyzer is not notified that a policy was added or updated. For example, a change to account-level block public access settings on an S3 bucket can take up to 12 hours. Also, if there is a delivery issue with AWS CloudTrail log delivery the policy change does not trigger a rescan of the resource that was reported in the finding. When this happens, Access Analyzer analyzes the new or updated policy during the next periodic scan, which is within 24 hours. If you want to confirm that a change you make to a policy resolves an access issue reported in a finding, you can rescan the resource reported in a finding by using the [Rescan](#) link in the Finding details page, or by using the [StartResourceScan](#) operation of the Access Analyzer API. To learn more, see [Resolving Findings \(p. 536\)](#).

Important

Access Analyzer analyzes only policies that are applied to resources in the same AWS Region that it's enabled in. To monitor all resources in your AWS environment, you must create an analyzer to enable Access Analyzer in each Region where you're using supported AWS resources.

Access Analyzer analyzes the following resource types:

- [Amazon Simple Storage Service Buckets \(p. 525\)](#)
- [AWS Identity and Access Management Roles \(p. 525\)](#)
- [AWS Key Management Service Keys \(p. 525\)](#)
- [AWS Lambda Functions and Layers \(p. 526\)](#)

- [Amazon Simple Queue Service Queues \(p. 527\)](#)

Supported Resource Types

Access Analyzer analyzes the resource-based policies that are applied to AWS resources in the Region where you enabled Access Analyzer. Only resource-based policies are analyzed. Review the information about each resource for details about how Access Analyzer generates findings for each resource type.

Supported resource types:

- [Amazon Simple Storage Service Buckets \(p. 525\)](#)
- [AWS Identity and Access Management Roles \(p. 525\)](#)
- [AWS Key Management Service Keys \(p. 525\)](#)
- [AWS Lambda Functions and Layers \(p. 526\)](#)
- [Amazon Simple Queue Service Queues \(p. 527\)](#)

Amazon Simple Storage Service Buckets

When Access Analyzer analyzes Amazon S3 buckets, it generates a finding when a bucket policy or ACL applied to a bucket grants access to an external entity. An external entity is a principal or other entity that you can use to [create a filter \(p. 534\)](#) that isn't within your zone of trust. For example, if a bucket policy grants access to another account or allows public access, Access Analyzer generates a finding. However, if you enable [Block public access](#) on your bucket, you can block access at the account level or the bucket level.

Amazon S3 *block public access* settings override the bucket policies that are applied to the bucket. Access Analyzer analyzes block public access settings at the bucket level whenever a policy changes. However, it evaluates the block public access settings at the account level only once every 6 hours. This means that Access Analyzer might not generate or resolve a finding for public access to a bucket for up to 6 hours. For example, if you have a bucket policy that allows public access, Access Analyzer generates a finding for that access. If you then enable block public access to block all public access to the bucket at the account level, Access Analyzer doesn't resolve the finding for the bucket policy for up to 6 hours, even though all public access to the bucket is blocked.

AWS Identity and Access Management Roles

For IAM roles, Access Analyzer analyzes [trust policies](#). In a role trust policy, you define the principals that you trust to assume the role. A role trust policy is a required resource-based policy that is attached to a role in IAM. Access Analyzer generates findings for roles within the zone of trust that can be accessed by an external entity that is outside your zone of trust.

Note

An IAM role is a global resource. If a role trust policy grants access to an external entity, Access Analyzer generates a finding in each enabled Region.

AWS Key Management Service Keys

For AWS KMS customer master keys (CMKs), Access Analyzer analyzes the key policies and grants applied to a key. Access Analyzer generates a finding if a key policy or grant allows an external entity to access the key. For example, if you use the [kms:CallerAccount](#) condition key in a policy statement to allow access to all users in a specific AWS account, and you specify an account other than the current account (in which Access Analyzer is enabled), Access Analyzer generates a finding. To learn more about KMS condition keys in IAM policy statements, see [AWS KMS Condition Keys](#).

When Access Analyzer analyzes a KMS key it reads key metadata, such as the key policy and list of grants. If the key policy doesn't allow the Access Analyzer role to read the key metadata, an Access Denied error finding is generated. For example, if the following example policy statement is the only policy applied to a key, it results in an Access Denied error finding in Access Analyzer:

```
{
    "Sid": "Allow access for Key Administrators",
    "Effect": "Allow",
    "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/Admin"
    },
    "Action": "kms:*",
    "Resource": "*"
}
```

Because this statement allows only the role named *Admin* from the AWS account 111122223333 to access the key, an Access Denied error finding is generated because Access Analyzer isn't able to fully analyze the key. An error finding is displayed in red text in the **Findings** table. The finding looks similar to the following:

```
{
    "error": "ACCESS_DENIED",
    "id": "12345678-1234-abcd-dcba-111122223333",
    "analyzedAt": "2019-09-16T14:24:33.352Z",
    "resource": "arn:aws:kms:us-west-2:1234567890:key/1a2b3c4d-5e6f-7a8b-9c0d-1a2b3c4d5e6f7g8a",
    "resourceType": "AWS::KMS::Key",
    "status": "ACTIVE",
    "updatedAt": "2019-09-16T14:24:33.352Z"
}
```

When you create a KMS CMK, the permissions granted to access the key depend on how you create the key. If you receive an Access Denied error finding for a key resource, apply the following policy statement to the key resource to grant Access Analyzer permission to access the key.

```
{
    "Sid": "Allow Access Analyzer access to key metadata",
    "Effect": "Allow",
    "Principal": {
        "AWS": "arn:aws:iam::111122223333:root"
    },
    "Action": [
        "kms:DescribeKey",
        "kms:GetKeyPolicy",
        "kms>List*"
    ],
    "Resource": "*"
},
```

After you receive an Access Denied finding for a KMS key resource, and then resolve the finding by updating the key policy, the finding is updated to a status of Resolved. If there are policy statements or key grants that grant permission to the key to an external entity, you might see additional findings for the key resource.

AWS Lambda Functions and Layers

For AWS Lambda functions, Access Analyzer analyzes policies, including condition statements in a policy, that grant access to the function to an external entity. Access Analyzer also analyzes permissions granted when using the [AddPermission](#) operation of the AWS Lambda API with an `EventSourceToken`.

Amazon Simple Queue Service Queues

For Amazon SQS queues, Access Analyzer analyzes policies, including condition statements in a policy, that allow an external entity access to a queue.

How Access Analyzer Works

This topic describes the concepts and terms that are used in Access Analyzer to help you become familiar with how Access Analyzer monitors access to your AWS resources.

IAM Access Analyzer is built on [Zelkova](#), which translates IAM policies into equivalent logical statements, and runs a suite of general-purpose and specialized logical solvers (satisfiability modulo theories) against the problem. Access Analyzer applies Zelkova repeatedly to a policy with increasingly specific queries to characterize classes of behaviors the policy allows, based on the content of the policy. To learn more about satisfiability modulo theories, see [Satisfiability Modulo Theories](#).

Access Analyzer does not examine access logs to determine whether an external entity accessed a resource within your zone of trust. It generates a finding when a resource-based policy allows access to a resource, even if the resource was not accessed by the external entity. Access Analyzer also does not consider the state of any external accounts when making its determination. That is, if it indicates that account 11112222333 can access your S3 bucket, it knows nothing about the state of users, roles, service control policies (SCP), and other relevant configurations in that account. This is for customer privacy – Access Analyzer doesn't consider who owns the other account. It is also for security – if the account is not owned by the Access Analyzer customer, it is still important to know that an external entity could gain access to their resources even if there are currently no principals in the account that could access the resources.

Access Analyzer considers only certain IAM condition keys that external users cannot directly influence, or that are otherwise impactful to authorization.

Access Analyzer does not currently report findings from AWS service principals or internal service accounts. In rare cases where Access Analyzer isn't able to fully determine whether a policy statement grants access to an external entity, it errs on the side of declaring a false positive finding. Access Analyzer is designed to provide a comprehensive view of the resource sharing in your account, and strives to minimize false negatives.

Getting Started with AWS IAM Access Analyzer

Use the information in this topic to learn about the requirements necessary to use and manage AWS IAM Access Analyzer, and then how to enable Access Analyzer. To learn more about the service-linked role for Access Analyzer, see [Using Service-Linked Roles for AWS IAM Access Analyzer \(p. 530\)](#).

Permissions Required to Use Access Analyzer

To successfully configure and use Access Analyzer, the account you use must be granted the required permissions. To access and use all Access Analyzer features, you can apply the IAMAccessAnalyzerFullAccess managed policy to the account. The full access policy grants the following permissions:

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

```
    "Effect": "Allow",
    "Action": [
        "access-analyzer:/*"
    ],
    "Resource": "*"
},
{
    "Effect": "Allow",
    "Action": "iam>CreateServiceLinkedRole",
    "Resource": "*",
    "Condition": {
        "StringEquals": {
            "iam:AWSServiceName": "access-analyzer.amazonaws.com"
        }
    }
},
{
    "Effect": "Allow",
    "Action": [
        "organizations:DescribeAccount",
        "organizations:DescribeOrganization",
        "organizations:DescribeOrganizationalUnit",
        "organizations>ListAccounts",
        "organizations>ListAccountsForParent",
        "organizations>ListAWSAccessForOrganization",
        "organizations>ListChildren",
        "organizations>ListDelegatedAdministrators",
        "organizations>ListOrganizationalUnitsForParent",
        "organizations>ListParents",
        "organizations>ListRoots"
    ],
    "Resource": "*"
}
]
```

A custom policy for managing Access Analyzer must include the following permissions:

- `access-analyzer: *`
- `iam>CreateServiceLinkedRole`

To allow read-only access to Access Analyzer, use the `IAMAccessAnalyzerReadOnlyAccess` managed policy. This policy grants the following permissions:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "access-analyzer:Get*",
                "access-analyzer>List*"
            ],
            "Resource": "*"
        }
    ]
}
```

Resources Defined by AWS IAM Access Analyzer

Access Analyzer defines the following resources:

Resource	ARN
analyzer	arn:\${Partition}:access-analyzer:\${Region}:\${Account}:analyzer/\${analyzerName}
archive-rule	arn:\${Partition}:access-analyzer:\${Region}:\${Account}:analyzer/\${analyzerName}/archive-rule/\${ruleName}

Required Access Analyzer Service Permissions

Access Analyzer uses a service-linked role named `AWSServiceRoleForAccessAnalyzer` to grant the service read-only access to analyze AWS resources with resource-based policies on your behalf. When you create an analyzer to enable Access Analyzer, the service creates the role in your account. For more information, see [Using Service-Linked Roles for AWS IAM Access Analyzer \(p. 530\)](#).

Note

Access Analyzer is Regional. You must enable Access Analyzer in each Region independently.

In some cases, after you enable Access Analyzer, the **Findings** page loads with no findings. This might be due to a delay in the console for populating your findings. You need to manually refresh the browser to view your findings. If you still don't see any findings, it's because you have no supported resources in your account that can be accessed by an external entity. If a policy that grants access to an external entity is applied to a resource, Access Analyzer generates a finding.

Note

It may take up to 30 minutes after a policy is modified for Access Analyzer to analyze the resource and then either generate a new finding or update an existing finding for the access to the resource.

Enabling Access Analyzer

To enable Access Analyzer in a Region, you must create an analyzer in that Region. You must create an analyzer in each Region in which you want to monitor access to your resources.

To create an analyzer

1. Open the IAM console at <https://console.aws.amazon.com/iam/>.
2. Choose **Access analyzer**.
3. Choose **Create analyzer**.
4. On the **Create analyzer** page, confirm that the Region displayed is the Region where you want to enable Access Analyzer.
5. Enter a name for the analyzer.
6. Optional. Add any tags that you want to apply to the analyzer.
7. Choose **Create Analyzer**.

When you create an analyzer to enable Access Analyzer, a service-linked role named `AWSServiceRoleForAccessAnalyzer` is created in your account.

Access Analyzer Quotas

Access Analyzer has the following quotas:

Resource	Default quota
Maximum analyzers with an account zone of trust	1

Resource	Default quota
Maximum archive rules per analyzer	100 Each archive rule can have up to 20 values per criterion.

Using Service-Linked Roles for AWS IAM Access Analyzer

AWS IAM Access Analyzer uses an IAM [service-linked role](#). A service-linked role is a unique type of IAM role that is linked directly to Access Analyzer. Service-linked roles are predefined by Access Analyzer and include all the permissions that the feature requires to call other AWS services on your behalf.

A service-linked role makes setting up Access Analyzer easier because you don't have to manually add the necessary permissions. Access Analyzer defines the permissions of its service-linked roles, and unless defined otherwise, only Access Analyzer can assume its roles. The defined permissions include the trust policy and the permissions policy, and that permissions policy cannot 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 Role** column. Choose a **Yes** with a link to view the service-linked role documentation for that service.

Service-Linked Role Permissions for AWS IAM Access Analyzer

AWS IAM Access Analyzer uses the service-linked role named **AWSServiceRoleForAccessAnalyzer** – Allow Access Analyzer to analyze resource metadata.

The **AWSServiceRoleForAccessAnalyzer** service-linked role trusts the following services to assume the role:

- `access-analyzer.amazonaws.com`

The role permissions policy allows Access Analyzer to complete the following actions on the specified resources:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetBucketPublicAccessBlock",
        "s3:GetBucketPolicyStatus",
        "s3:GetAccountPublicAccessBlock",
        "s3>ListAllMyBuckets",
        "s3:GetBucketAcl",
        "s3:GetBucketLocation",
        "s3:GetBucketPolicy",
        "iam:GetRole",
        "iam>ListRoles",
        "kms:DescribeKey",
        "kms:GetKeyPolicy",
        "kms>ListGrants",
        "kms>ListKeyPolicies",
        "kms>ListKeys",
        "kms:PutKeyPolicy"
      ]
    }
  ]
}
```

```
"ec2:DescribeVpcs",
"ec2:DescribeVpcEndpoints",
"ec2:DescribeByoipCidrs",
"ec2:DescribeAddresses",
"lambda>ListFunctions",
"lambda:GetPolicy",
"lambda>ListLayers",
"lambda>ListLayerVersions",
"lambda:GetLayerVersionPolicy",
"sqsc:GetQueueAttributes",
"sqsc>ListQueues",
"organizations>ListAWSServiceAccessForOrganization",
"organizations>ListDelegatedAdministrators",
"organizations>ListRoots",
"organizations>ListParents",
"organizations>ListChildren",
"organizations>ListOrganizationalUnitsForParent",
"organizations>ListAccountsForParent",
"organizations>ListAccounts",
"organizations>DescribeAccount",
"organizations>DescribeOrganization",
"organizations>DescribeOrganizationalUnit"
],
"Resource": "*"
}
]
}
```

You must configure permissions to allow an IAM entity (such as a user, group, or role) to create, edit, or delete a service-linked role. For more information, see [Service-Linked Role Permissions](#) in the *IAM User Guide*.

Creating a Service-Linked Role for Access Analyzer

You don't need to manually create a service-linked role. When you enable Access Analyzer in the AWS Management Console or the AWS API, Access Analyzer creates the service-linked role for you. The same service-linked role is used in all Regions in which you enable Access Analyzer.

Note

Access Analyzer is Regional. You must enable Access Analyzer in each Region independently.

If you delete this service-linked role, Access Analyzer recreates the role when you next create an analyzer.

You can also use the IAM console to create a service-linked role with the **Access Analyzer** use case. In the AWS CLI or the AWS API, create a service-linked role with the `access-analyzer.amazonaws.com` service name. For more information, see [Creating a Service-Linked Role](#) in the *IAM User Guide*. If you delete this service-linked role, you can use this same process to create the role again.

Editing a Service-Linked Role for Access Analyzer

Access Analyzer does not allow you to edit the `AWSServiceRoleForAccessAnalyzer` service-linked role. After you create a service-linked role, you cannot 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 Access Analyzer

If you no longer need to use a feature or service that requires a service-linked role, we recommend that you delete that role. That way you don't have an unused entity that isn't actively monitored or maintained. However, you must clean up the resources for your service-linked role before you can manually delete it.

Note

If Access Analyzer is using the role when you try to delete the resources, then the deletion might fail. If that happens, wait for a few minutes and try the operation again.

To delete Access Analyzer resources used by the AWSServiceRoleForAccessAnalyzer

1. Open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the **Access reports** section, under **Access analyzer**, choose **Analyzer details**.
3. Choose **Delete**.
4. To confirm that you want to delete the analyzer, enter **delete**, and then choose **Delete**.

To manually delete the service-linked role using IAM

Use the IAM console, the AWS CLI, or the AWS API to delete the AWSServiceRoleForAccessAnalyzer service-linked role. For more information, see [Deleting a Service-Linked Role](#) in the *IAM User Guide*.

Supported Regions for Access Analyzer Service-Linked Roles

Access Analyzer supports using service-linked roles in all of the Regions where the service is available. For more information, see [AWS Regions and Endpoints](#).

Access Analyzer Findings

Access Analyzer generates a finding for each instance of a resource-based policy that grants access to a resource in your zone of trust (your account) to an external entity. Any sharing that is within the zone of trust is considered safe, so Access Analyzer doesn't generate a finding. For example, if you grant permissions to an S3 bucket in your account to another AWS account, Access Analyzer generates a finding. But if you grant permission to a bucket in your account to an IAM role in your account, Access Analyzer doesn't generate a finding.

Topics

- [Working with Findings \(p. 532\)](#)
- [Review Findings \(p. 533\)](#)
- [Filtering Findings \(p. 534\)](#)
- [Archiving Findings \(p. 536\)](#)
- [Resolving Findings \(p. 536\)](#)

Working with Findings

Findings are generated only once for each instance of a resource that is shared outside of your zone of trust. Each time a resource-based policy is modified, Access Analyzer analyzes the policy. If the updated policy shares a resource that is already identified in a finding, but with different permissions or conditions, a new finding is generated for that instance of the resource sharing. If the access in the first finding is removed, that finding is updated to a status of Resolved.

The status of all findings remains Active until you archive them or remove the access that generated the finding. When you remove the access, the finding status is updated to Resolved.

Note

It may take up to 30 minutes after a policy is modified for Access Analyzer to analyze the resource and then update the finding.

You should review all of the findings in your account to determine whether the sharing is expected and approved. If the sharing identified in the finding is expected, you can archive the finding. When you archive a finding, the status is changed to Archived, and the finding is removed from the Active findings list. The finding is not deleted. You can view your archived findings at any time. Work through all of the findings in your account until you have zero active findings. After you get to zero findings, you know that any new Active findings that are generated are from a recent change in your environment.

Review Findings

After you [enable Access Analyzer \(p. 529\)](#), the next step is to review any findings to determine whether the access identified in the finding is intentional or unintentional. You can also review findings to determine common findings for access that is intended, and then [create an archive rule \(p. 537\)](#) to automatically archive those findings. You can also review archived and resolved findings.

To review findings

1. Open the IAM console at <https://console.aws.amazon.com/iam/>.
2. Choose **Access analyzer**.

Note

Findings are displayed only if you have permission to view findings for the analyzer.

All Active findings are displayed for the analyzer. To view other findings generated by the analyzer, choose the appropriate tab:

- Choose **Active** to view all active findings that were generated by the analyzer.
- Choose **Archived** to view only findings generated by the analyzer that have been archived. To learn more, see [Archiving Findings \(p. 536\)](#).
- Choose **Resolved** to view only findings that were generated by the analyzer that have been resolved. When you remediate the issue that generated the finding, the finding status is changed to Resolved.

Important

Resolved findings are deleted 90 days after the last update to the finding. Active and archived findings are not deleted unless you delete the analyzer that generated them.

- Choose **All** to view all findings with any status that were generated by the analyzer.

The **Findings** page displays the following details about the shared resource and policy statement that generated the finding:

Finding ID

The unique ID assigned to the finding. Choose the finding ID to display additional details about the resource and policy statement that generated the finding.

Resource

The type and partial name of the resource that has a policy applied to it that grants access to an external entity not within your zone of trust.

External principal

The principal, not within your zone of trust, that the analyzed policy grants access to. Valid values include:

- **AWS account** – All principals in the listed AWS account with permissions from that account's administrator can access the resource.
- **Any principal** – All principals in any AWS account that meet the conditions included in the **Conditions** column have permission to access the resource. For example, if a VPC is listed, it means

that any principal in any account that has permission to access the listed VPC can access the resource.

- **Canonical user** – All principals in the AWS account with the listed canonical user ID have permission to access the resource.
- **IAM role** – The listed IAM role has permission to access the resource.
- **IAM user** – The listed IAM user has permission to access the resource.

Condition

The condition from the policy statement that grants the access. For example, if the **Condition** field includes **Source VPC**, it means that the resource is shared with a principal that has access to the VPC listed. Conditions can be global or service-specific. [Global condition keys](#) have the `aws:` prefix.

Access level

The level of access granted to the external entity by the actions in the resource-based policy. View the details of the finding for more information. Access level values include the following:

- **List** – Permission to list resources within the service to determine whether an object exists. Actions with this level of access can list objects but cannot see the contents of a resource.
- **Read** – Permission to read but not edit the contents and attributes of resources in the service.
- **Write** – Permission to create, delete, or modify resources in the service.
- **Permissions** – Permission to grant or modify resource permissions in the service.
- **Tagging** – Permission to perform actions that only change the state of resource tags.

Updated

A timestamp for the most recent update to the finding status, or the time and date at which the finding was generated if no updates have been made.

Note

It may take up to 30 minutes after a policy is modified for Access Analyzer to again analyze the resource and then update the finding.

Status

The status of the finding, one of **Active**, **Archived**, or **Resolved**.

Filtering Findings

The default filtering for the page is to display all active findings. To view archived findings, choose the **Archived** tab. When you first start using Access Analyzer, there are no archived findings.

Use filters to display only the findings for a specific resource, account, principal, or other value. To create a filter, select the property to filter on, then choose a property value to filter on. For example, to create a filter that displays only findings for a specific AWS account, choose **AWS Account** for the property, then enter the account number for the AWS account that you want to view findings for.

To filter the findings displayed

1. Choose the **Filter active findings** field.
2. Choose the property to use to filter the findings displayed.
3. Choose the value to match for the property. Only findings with that value in the finding are displayed.

For example, if you choose **Resource** as the property, type part or all of the name of a bucket, then press Enter. Only findings for the bucket that matches the filter criteria are displayed.

You can add additional properties to further filter the findings displayed. When you add additional properties, only findings that match all conditions in the filter are displayed. Defining a filter to display findings that match one property OR another property is not supported.

The following properties are available for defining filters:

- **Resource** – To filter by resource, type all or part of the name of the resource.
- **Resource Type** – To filter by resource type, choose the type from the list displayed.
- **AWS Account** – Use this property to filter by AWS account that is granted access in the **Principal** section of a policy statement. To filter by AWS account, type all or part of the 12-digit AWS account ID, or all or part of the full account ARN of the external AWS user or role that has access to resources in the current account.
- **Canonical User** – To filter by canonical user, type the canonical user ID as defined for S3 buckets. To learn more, see [AWS Account Identifiers](#).
- **Federated User** – To filter by federated user, type all or part of the ARN of the federated identity. To learn more, see [Identity Providers and Federation](#).
- **Principal ARN** – Use this property to filter on the ARN of the principal (IAM user, role, or group) used in an `aws:PrincipalArn` condition key. To filter by Principal ARN, type all or part of the ARN of the IAM user, role, or group from an external AWS account reported in a finding.
- **Principal OrgID** – To filter by Principal OrgID, type all or part of the organization ID associated with the external principals that belong to the AWS organization specified as a condition in the finding. To learn more, see [AWS Global Condition Context Keys](#).
- **Principal Org Paths** – To filter by Principal Org Paths, type all or part of the ID for the AWS organization or organizational unit (OU) that allows access to all external principals that are account members of the specified organization or OU as a condition in the policy. To learn more, see [AWS Global Condition Context Keys](#).
- **Source Account** – To filter on source account, type all or part of the AWS account ID associated with the resources, as used in some cross-service permissions in AWS.
- **Source ARN** – To filter by Source ARN, type all or part of the ARN specified as a condition in the finding. To learn more, see To filter by Principal Org Paths, type all or part of the ID for the AWS organization or organizational unit (OU) that allows access to all external principals that are account members of the specified organization or OU as a condition in the policy. To learn more, see [AWS Global Condition Context Keys](#).
- **Source IP** – To filter by Source IP, type all or part of the IP address that allows external entities access to resources in the current account when using the specified IP address. To learn more, see [AWS Global Condition Context Keys](#).
- **Source VPC** – To filter by Source VPC, type all or part of the VPC ID that allows external entities access to resources in the current account when using the specified VPC. To learn more, see [AWS Global Condition Context Keys](#).
- **Source VPCE** – filter by Source VPCE, type all or part of the VPC endpoint ID that allows external entities access to resources in the current account when using the specified VPC endpoint. To learn more, see [AWS Global Condition Context Keys](#).
- **User ID** – To filter by User ID, type all or part of the user ID of the IAM user from an external AWS account who is allowed access to resource in the current account. To learn more, see [AWS Global Condition Context Keys](#).
- **KMS Key ID** – To filter by KMS Key ID, type all or part of the key ID for the KMS key specified as a condition for KMS-encrypted S3 object access in your current account.
- **Google Audience** – To filter by Google Audience, type all or part of the Google application ID specified as a condition for IAM role access in your current account. To learn more, see [IAM and AWS STS Condition Context Keys](#).
- **Cognito Audience** – To filter by Cognito Audience, type all or part of the Amazon Cognito identity pool ID specified as a condition for IAM role access in your current account. To learn more, see [IAM and AWS STS Condition Context Keys](#).

- **Caller Account** – The AWS account ID of the account that owns or contains the calling entity, such as an IAM role, user, or account root user. This is used by services calling KMS. To filter by caller account, type all or part of the AWS account ID.
- **Facebook App ID** – To filter by Facebook App ID, type all or part of the Facebook application ID (or site ID) specified as a condition to allow Login with Facebook federation access to an IAM role in your current account. To learn more, see [IAM and AWS STS Condition Context Keys](#).
- **Amazon App ID** – To filter by Amazon App ID, type all or part of the Amazon application ID (or site ID) specified as a condition to allow Login with Amazon federation access to an IAM role in your current account. To learn more, see [IAM and AWS STS Condition Context Keys](#).
- **Lambda Event Source Token** – To filter on Lambda Event Source Token passed in with Alexa integrations, type all or part of the token string.

Archiving Findings

When you get a finding for access to a resource that is intentional, such as an IAM role that is used by multiple users for approved workflows, you can archive the finding. When you archive a finding it is cleared from Active findings list, letting you focus on the findings you need to resolve. Archived findings aren't deleted. You can filter the Findings page to display your archived findings, and unarchive them at any time.

To archive findings from the **Findings** page

1. Select the check box next to one or more findings to archive.
2. Choose **Archive**.

A confirmation is displayed at the top of the screen.

To archive findings from the **Findings Details** page.

1. Choose the **Finding ID** for the finding to archive.
2. Choose **Archive**.

A confirmation is displayed at the top of the screen.

To unarchive findings, repeat the preceding steps, but choose **Unarchive** instead of **Archive**. When you unarchive a finding, the status is set to Active.

Resolving Findings

To resolve findings generated from access that you did not intend to allow, modify the policy statement to remove the permissions that allow access to the identified resource. For example, for findings on S3 buckets, use the Amazon S3 console to configure the permissions on the bucket. For IAM roles, use the IAM console to [modify the trust policy](#) for the listed IAM role. Use the console for the other supported resources to modify the policy statements that resulted in a generated finding.

After you make a change to resolve a finding, such as modifying a policy applied to an IAM role, Access Analyzer scans the resource again. If the resource is no longer shared outside of your zone of trust, the status of the finding is changed to Resolved. The finding is no longer displayed in the **Active findings** table, and instead is displayed in the **Resolved findings** table.

Note

This does not apply to Error findings. When Access Analyzer is not able to access a resource, it generates an error finding. If you resolve the issue that prevented Access Analyzer from

accessing the resource, the error finding is removed completely rather than changing to a resolved finding.

If the changes you made resulted in the resource being shared outside of your zone of trust, but in a different way, such as with a different principal or for a different permission, Access Analyzer generates a new Active finding.

Note

It may take up to 30 minutes after a policy is modified for Access Analyzer to again analyze the resource and then update the finding. Resolved findings are deleted 90 days after the last update to the finding status.

Archive Rules

Archive rules automatically archive new findings that meet the criteria you define when you create the rule. For example, you can create an archive rule to automatically archive any findings for a specific S3 bucket that you regularly grant access to. Or if you grant access to multiple resources to a specific principal, you can create a rule that automatically archives any new finding generated for access granted to that principal. This lets you focus only on active findings that may indicate a security risk.

Use the information provided in the finding details to identify the specific resource and external entity to use when creating or editing a rule. When you create an archive rule, only new findings that match the rule criteria are automatically archived. Existing findings are not automatically archived. When you create a rule, you can include up to 20 values per criterion in the rule.

Note

When you create or edit an archive rule, Access Analyzer does not validate the values you include in the filter for the rule. For example, if you add a rule to match an AWS Account, Access Analyzer accepts any value in the field, even if it is not a valid AWS account number.

To create an archive rule

1. Open the IAM console at <https://console.aws.amazon.com/iam/>.
2. Choose **Access analyzer**, then choose **Archive rules**.
3. Choose **Create archive rule**.
4. Enter a name for the rule if you want to change the default name.
5. In the **Rule** section, under **Criteria**, select a property to match for the rule.
6. Choose an operator for the property value, such as **contains**.

The operators available depend on the property you choose.

7. Optionally, add additional values for the property, or add additional criteria for the rule.
To add another value for a criterion, choose **Add another value**. To add another criterion for the rule, choose the **Add** button.
8. When finished adding criteria and values, choose **Create archive rule**.

For example, to create a rule that automatically archives any findings for S3 buckets: choose **Resource type**, and then choose **is** for the operator. Next choose **S3 bucket** from the **Select resource type** list, and then choose **Add**.

Continue to define criteria to customize the rule as appropriate for your environment, and then choose **Create archive rule**.

If you are creating a new rule and adding multiple criteria, you can remove a single criterion from the rule by choosing **Remove this criterion**. You can remove a value added for a criterion by choosing **Remove value**.

To edit an archive rule

1. Choose name of the rule to edit in the **Name**.

You can edit only one archive rule at a time.
2. Add new or remove the existing criteria and values for each criterion.
3. Choose **Save changes**.

To delete an archive rule

1. Select the check box for the rules to delete.

You can delete one, many, or all rules at the same time.
2. Choose **Delete**.
3. Type **delete** in the **Delete archive rule** confirmation dialog, and then choose **Delete**.

The rules are deleted only from the analyzer in the current Region. You must delete archive rules separately for each analyzer that you created in other Regions.

Monitoring AWS IAM Access Analyzer with Amazon EventBridge

Use the information in this topic to learn how to monitor Access Analyzer findings with Amazon EventBridge. EventBridge is the new version of Amazon CloudWatch Events.

Findings Events

Access Analyzer sends an event to EventBridge for each generated finding, for a change to the status of an existing finding, and when a finding is deleted. To receive findings and notifications about findings, you must create an event rule in Amazon EventBridge. When you create an event rule, you can also specify a target action to trigger based on the rule. For example, you could create an event rule that triggers an Amazon SNS topic when an event for a new finding is received from Access Analyzer.

Event Notification Frequency

Access Analyzer sends events for new findings and findings with status updates to EventBridge within about an hour from when the event occurs in your account. Access Analyzer also sends events to EventBridge when a resolved finding is deleted because the retention period has expired. For findings that are deleted because the analyzer that generated them is deleted, the event is sent to EventBridge approximately 24 hours after the analyzer was deleted. When a finding is deleted, the finding status is not changed. Instead, the `isDeleted` attribute is set to `true`.

Example Event

The following is an example Access Analyzer event sent to EventBridge. The `id` listed is the ID for the event in EventBridge. To learn more, see [Events and Event Patterns in EventBridge](#).

In the `detail` object, the values for the `accountId` and `region` attributes refer to the account and Region reported in the finding. The `isDeleted` attribute indicates whether the event was from the finding being deleted.

```
{
  "id": "22222222-dcba-4444-dcba-333333333333",
  "detail-type": "Access Analyzer Finding",
  "source": "aws.access-analyzer",
  "account": "111122223333",
  "time": "2019-11-21T01:22:33Z",
  "region": "us-west-2",
  "resources": [
    "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/MyAnalyzer"
  ],
  "detail": {
    "version": "1.0",
    "accountId": "111122223333",
    "region": "us-west-2",
    "isDeleted": false,
    COMPLETE_ACCESS_ANALYZER_GET_FINDING_RESPONSE
  }
}
```

The "id" is the finding ID. The "resources" array is a singleton with the ARN of the analyzer that generated the finding.

The following example shows data for an event that is sent to EventBridge from the `GetFinding` operation of the Access Analyzer API.

```
"version": "0",
"id": "22222222-dcba-4444-dcba-333333333333",
"status": "ACTIVE",
"resourceType": "AWS::S3::Bucket",
"resource": "arn:aws:s3:::my-bucket",
"createdAt": "2019-11-20T04:58:50Z",
"analyzedAt": "2019-11-21T01:22:22Z",
"updatedAt": "2019-11-21T01:14:07Z",
"principal": {"AWS": "999988887777"},
"action": ["s3:GetObject"],
"condition": {},
"isPublic": false
```

Access Analyzer also sends events to EventBridge for error findings. An error finding is a finding generated when Access Analyzer can't access a resource it tries to analyze. Events for error findings include an `error` attribute as shown in the following example.

```
"id": "22222222-dcba-4444-dcba-333333333333",
"status": "ACTIVE",
"resourceType": "AWS::S3::Bucket",
"resource": "arn:aws:s3:::my-bucket",
"error": "ACCESS_DENIED",
"createdAt": "2019-10-16T19:21:44.244Z",
"analyzedAt": "2019-10-16T19:21:44.244Z",
"updatedAt": "2019-10-16T19:21:44.244Z"
```

Creating an Event Rule with a Target

The following procedure describes how to create an event rule using the console.

Open the Amazon EventBridge console at <https://console.aws.amazon.com/events/>.

1. Choose **Create rule**.
2. Enter a **Name** and, optionally, a **Description**.

3. Under **Define pattern** choose **Event pattern**, then choose **Custom pattern**.
4. Copy the following example and then paste it into the **Event pattern** box.

```
{  
    "source": [  
        "aws.access-analyzer"  
    ],  
    "detail-type": [  
        "Access Analyzer Finding"  
    ]  
}
```

5. Choose **Save**.
6. Under **Select targets**, choose a **Target** action for the rule, such as an Amazon SNS topic or AWS Lambda function.
7. Choose the specific SNS topic or Lambda function to use when the target is triggered.

The target is triggered when an event is received that matches the event pattern defined in the rule.
8. Choose **Save** to create the rule.

To learn more about creating rules, see [Creating an EventBridge Rule That Triggers on an Event from an AWS Resource](#).

Create a Rule Using the CLI

1. Use the following to create a rule for Amazon EventBridge using the AWS CLI. Replace the rule name **TestRule** with the name for your rule.

```
aws events put-rule --name TestRule --event-pattern "{\"source\":[\"aws.access-analyzer\"]}"
```

2. You can customize the rule to trigger target actions only for a subset of generated findings, such as findings with specific attributes. The following example demonstrates how to create a rule that triggers a target action only for findings with a status of Active.

```
aws events put-rule --name TestRule --event-pattern "{\"source\":[\"aws.access-analyzer\"],\"detail-type\":[\"Access Analyzer Finding\"],\"detail\":{\"status\":[\"ACTIVE\"]}}"
```

3. To define a Lambda function as a target for the rule you created, use the following example command. Replace the Region and the function name in the ARN as appropriate for your environment.

```
aws events put-targets --rule TestRule --targets Id=1,Arn=arn:aws:lambda:us-east-1:111122223333:function:MyFunction
```

4. Add the permissions required to invoke the rule target. The following example demonstrates how to grant permissions to a Lambda function, following the preceding examples.

```
aws lambda add-permission --function-name MyFunction --statement-id 1 --action 'lambda:InvokeFunction' --principal events.amazonaws.com
```

Logging Access Analyzer API Calls with AWS CloudTrail

Access Analyzer is integrated with AWS CloudTrail, a service that provides a record of actions taken by a user, role, or an AWS service in Access Analyzer. CloudTrail captures all API calls for Access Analyzer as events. The calls captured include calls from the Access Analyzer console and code calls to the Access Analyzer API operations.

If you create a trail, you can enable continuous delivery of CloudTrail events to an Amazon S3 bucket, including events for Access Analyzer. 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 Access Analyzer, 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](#).

Access Analyzer Information in CloudTrail

CloudTrail is enabled on your AWS account when you create the account. When activity occurs in Access Analyzer, 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 Access Analyzer, 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 AWS 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 the following:

- [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 Access Analyzer actions are logged by CloudTrail and are documented in the [IAM Access Analyzer API Reference](#). For example, calls to the `CreateAnalyzer`, `CreateArchiveRule` and `ListFindings` actions generate entries in the CloudTrail log files.

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 AWS Identity and Access Management (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 Access Analyzer 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 aren't an ordered stack trace of the public API calls, so they don't appear in any specific order.

The following example shows a CloudTrail log entry that demonstrates the `CreateAnalyzer` operation made by a user named "Alice" on "June 14, 2018".

```
{  
    "eventVersion": "1.05",  
    "userIdentity": {  
        "type": "AssumedRole",  
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",  
        "arn": "arn:aws:iam::111122223333:user/Alice",  
        "accountId": "111122223333",  
        "accessKeyId": "AKIAI44QH8DHBEXAMPLE",  
        "sessionContext": {  
            "attributes": {  
                "mfaAuthenticated": "false",  
                "creationDate": "2018-06-14T22:54:20Z"  
            },  
            "sessionIssuer": {  
                "type": "Role",  
                "principalId": "AIDACKCEVSQ6C2EXAMPLE",  
                "arn": "arn:aws:iam::111122223333:user/Alice",  
                "accountId": "111122223333",  
                "userName": "Alice"  
            }  
        }  
    },  
    "eventTime": "2018-06-14T22:57:36Z",  
    "eventSource": "access-analyzer.amazonaws.com",  
    "eventName": "CreateAnalyzer",  
    "awsRegion": "us-west-2",  
    "sourceIPAddress": "198.51.100.179",  
    "userAgent": "aws-cli/1.16.205 Python/2.7.16 Darwin/17.7.0 botocore/1.12.195",  
    "requestParameters": {  
        "analyzerName": "test",  
        "type": "ACCOUNT",  
        "clientToken": "11111111-abcd-2222-abcd-222222222222"  
    },  
    "responseElements": {  
        "arn": "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/test"  
    },  
    "requestID": "22222222-dcba-4444-dcba-333333333333",  
    "eventID": "33333333-bcde-5555-bcde-444444444444",  
    "readOnly": false,  
    "eventType": "AwsApiCall",  
    "recipientAccountId": "111122223333"  
}
```

Troubleshooting IAM

If you encounter access-denied issues or similar difficulties when working with AWS Identity and Access Management (IAM), consult the topics in this section.

Topics

- [Troubleshooting General Issues \(p. 543\)](#)
- [Troubleshoot IAM Policies \(p. 547\)](#)
- [Troubleshooting U2F Security Keys \(p. 561\)](#)
- [Troubleshooting IAM Roles \(p. 563\)](#)
- [Troubleshooting Amazon EC2 and IAM \(p. 566\)](#)
- [Troubleshooting Amazon S3 and IAM \(p. 569\)](#)
- [Troubleshooting SAML 2.0 Federation with AWS \(p. 569\)](#)

Troubleshooting General Issues

Use the information here to help you diagnose and fix access-denied or other common issues when you work with AWS Identity and Access Management (IAM).

Topics

- [I Lost My Access Keys \(p. 543\)](#)
- [I Need to Access an Old Account \(p. 543\)](#)
- [I Can't Sign in to My Account \(p. 544\)](#)
- [I Get "Access Denied" When I Make a Request to an AWS Service \(p. 544\)](#)
- [I Get "Access Denied" When I Make a Request with Temporary Security Credentials \(p. 545\)](#)
- [Policy Variables Aren't Working \(p. 546\)](#)
- [Changes That I Make Are Not Always Immediately Visible \(p. 546\)](#)
- [I Am Not Authorized to Perform: iam>DeleteVirtualMFADevice \(p. 547\)](#)

I Lost My Access Keys

Access keys consist of two parts:

- **The access key identifier.** This is not a secret, and can be seen in the IAM console wherever access keys are listed, such as on the user summary page.
- **The secret access key.** This is provided when you initially create the access key pair. Just like a password, it **cannot be retrieved later**. If you lost your secret access key, then you must create a new access key pair. If you already have the [maximum number of access keys \(p. 582\)](#), you must delete an existing pair before you can create another.

For more information, see [Resetting Your Lost or Forgotten Passwords or Access Keys \(p. 118\)](#).

I Need to Access an Old Account

When you first created your AWS account, you provided an email address and password. These are your AWS account root user credentials. If you have an old AWS account that you can no longer access because you have lost or forgotten the password, you can recover the password. For more information, see [Resetting Your Lost or Forgotten Passwords or Access Keys \(p. 118\)](#).

If you no longer have access to the email, you should first try to recover access to the email. If that doesn't work, you can contact AWS Customer Service.

You can try to recover access to your email using one of the following options:

- If you own the domain where the email address is hosted, you can add the email address back to your email server. Alternatively, you can set up a catch-all for your email account. The catch-all setting on your domain "catches all" messages that are sent to email addresses that do not exist in the mail server. It redirects those messages to a specific email address. For example, assume that your AWS account root user email address is `paulo@sample-domain.com`, but you changed your only domain email address to `paulo.santos@sample-domain.com`. In that case, you could set your new email as the catch-all. That way, when someone like AWS sends a message to `paulo@sample-domain.com` or any other `text@sample-domain.com`, you receive that message at `paulo.santos@sample-domain.com`.
- If the email address on the account is part of your corporate email system, we recommend that you contact your IT system administrators. They might be able to help you regain access to the email address.

If you're still not able to access your AWS account, you can find alternate support options at [Contact us](#) by expanding the **I'm an AWS customer and I'm looking for billing or account support** menu. When you contact AWS Support, you must provide the following information:

- All the details that are listed on the account, including your full name, phone number, address, email address, and last four digits of the credit card. You might need to create a new AWS account for the purpose of contacting AWS Support. This is necessary to assist in the investigation of your request.
- The reason that you're unable to access the email account to receive password reset instructions.
- After you recover your account, close any accounts that you are not using. It's a good idea not to have open accounts in your name that could result in charges to you. For more information, see [Closing an Account](#) in the *Billing and Cost Management User Guide*.

I Can't Sign in to My Account

When you first created your AWS account, you provided an email address and password. These are your AWS account root user credentials. If you can no longer access your account because you have lost or forgotten your password, you can recover that password. For more information, see [Resetting Your Lost or Forgotten Passwords or Access Keys \(p. 118\)](#).

If you provided your account email address and password, AWS sometimes requires you to provide a one-time verification code. To retrieve the verification code, check the email that is associated with your AWS account for a message from Amazon Web Services. The email address ends in @amazon.com or @aws.amazon.com. Follow the directions in the message. If you don't see the message in your account, check your spam folder. If you no longer have access to the email, see [I Need to Access an Old Account \(p. 543\)](#).

I Get "Access Denied" When I Make a Request to an AWS Service

- Verify that you have the identity-based policy permission to call the action and resource that you have requested. If any conditions are set, you must also meet those conditions when you send the request. For information about viewing or modifying policies for an IAM user, group, or role, see [Managing IAM Policies \(p. 442\)](#).
- Are you trying to access a service that supports [resource-based policies \(p. 379\)](#), such as Amazon S3, Amazon SNS, or Amazon SQS? If so, verify that the policy specifies you as a principal and grants you access. If you make a request to a service within your account, either your identity-based policies or

the resource-based policies can grant you permission. If you make a request to a service in a different account, then both your identity-based policies and the resource-based policies must grant you permission. To view the services that support resource-based policies, see [AWS Services That Work with IAM \(p. 586\)](#).

- If your policy includes a condition with a key–value pair, review it carefully. Examples include the `aws:RequestTag/tag-key` (p. 665) global condition key, the AWS KMS `kms:EncryptionContext:encryption_context_key`, and the `ResourceTag/tag-key` condition key supported by multiple services. Make sure that the key name does not match multiple results. Because condition key names are not case sensitive, a condition that checks for a key named `foo` matches `foo`, `Foo`, or `FOO`. If your request includes multiple key–value pairs with key names that differ only by case, then your access might be unexpectedly denied. For more information, see [IAM JSON Policy Elements: Condition \(p. 611\)](#).
- If you have a [permissions boundary \(p. 370\)](#), verify that the policy that is used for the permissions boundary allows your request. If your identity-based policies allow the request, but your permissions boundary does not, then the request is denied. A permissions boundary controls the maximum permissions that an IAM principal (user or role) can have. Resource-based policies are not limited by permissions boundaries. Permissions boundaries are not common. For more information about how AWS evaluates policies, see [Policy Evaluation Logic \(p. 636\)](#).
- If you are signing requests manually (without using the [AWS SDKs](#)), verify that you have correctly signed the request.

I Get "Access Denied" When I Make a Request with Temporary Security Credentials

- First, make sure that you are not denied access for a reason that is unrelated to your temporary credentials. For more information, see [I Get "Access Denied" When I Make a Request to an AWS Service \(p. 544\)](#).
- Verify that the service accepts temporary security credentials, see [AWS Services That Work with IAM \(p. 586\)](#).
- Verify that your requests are being signed correctly and that the request is well-formed. For details, see your [toolkit](#) documentation or [Using Temporary Credentials With AWS Resources \(p. 319\)](#).
- Verify that your temporary security credentials haven't expired. For more information, see [Temporary Security Credentials \(p. 307\)](#).
- Verify that the IAM user or role has the correct permissions. Permissions for temporary security credentials are derived from an IAM user or role. As a result, the permissions are limited to those that are granted to the role whose temporary credentials you have assumed. For more information about how permissions for temporary security credentials are determined, see [Controlling Permissions for Temporary Security Credentials \(p. 322\)](#).
- If you assumed a role, your role session might be limited by session policies. When you [request temporary security credentials \(p. 309\)](#) programmatically using AWS STS, you can optionally pass inline or managed [session policies \(p. 358\)](#). Session policies are advanced policies that you pass as a parameter when you programmatically create a temporary credential session for a role. You can pass a single JSON inline session policy document using the `Policy` parameter. You can use the `PolicyArns` parameter to specify up to 10 managed session policies. The resulting session's permissions are the intersection of the role's identity-based policies and the session policies. Alternatively, if your administrator or a custom program provides you with temporary credentials, they might have included a session policy to limit your access.
- If you are a federated user, your session might be limited by session policies. You become a federated user by signing in to AWS as an IAM user and then requesting a federation token. For more information about federated users, see [GetFederationToken—Federation Through a Custom Identity Broker \(p. 314\)](#). If you or your identity broker passed session policies while requesting a federation token, then your session is limited by those policies. The resulting session's permissions are the

intersection of your IAM user identity-based policies and the session policies. For more information about session policies, see [Session Policies \(p. 358\)](#).

- If you are accessing a resource that has a resource-based policy by using a role, verify that the policy grants permissions to the role. For example, the following policy allows MyRole from account 111122223333 to access MyBucket.

```
{  
    "Version": "2012-10-17",  
    "Statement": [{  
        "Sid": "S3BucketPolicy",  
        "Effect": "Allow",  
        "Principal": {"AWS": ["arn:aws:iam::111122223333:role/MyRole"]},  
        "Action": ["s3:PutObject"],  
        "Resource": ["arn:aws:s3:::MyBucket/*"]  
    }]  
}
```

Policy Variables Aren't Working

- Verify that all policies that include variables include the following version number in the policy: "Version": "2012-10-17". Without the correct version number, the variables are not replaced during evaluation. Instead, the variables are evaluated literally. Any policies that don't include variables will still work if you include the latest version number.

A `Version` policy element is different from a policy version. The `Version` policy element is used within a policy and defines the version of the policy language. A policy version, on the other hand, is created when you make changes to a customer managed policy in IAM. The changed policy doesn't overwrite the existing policy. Instead, IAM creates a new version of the managed policy. To learn more about the `Version` policy element see [IAM JSON Policy Elements: Version \(p. 598\)](#). To learn more about policy versions, see [the section called "Versioning IAM Policies" \(p. 465\)](#).

- Verify that your policy variables are in the right case. For details, see [IAM Policy Elements: Variables and Tags \(p. 628\)](#).

Changes That I Make Are Not Always Immediately Visible

As a service that is accessed through computers in data centers around the world, IAM uses a distributed computing model called [eventual consistency](#). Any change that you make in IAM (or other AWS services) takes time to become visible from all possible endpoints. Some of the delay results from the time it takes to send the data from server to server, from replication zone to replication zone, and from Region to Region around the world. IAM also uses caching to improve performance, but in some cases this can add time: The change might not be visible until the previously cached data times out.

You must design your global applications to account for these potential delays. Ensure that they work as expected, even when a change made in one location is not instantly visible at another. Such changes include creating or updating users, groups, roles, or policies. We recommend that you do not include such IAM changes in the critical, high-availability code paths of your application. Instead, make IAM changes in a separate initialization or setup routine that you run less frequently. Also, be sure to verify that the changes have been propagated before production workflows depend on them.

For more information about how some other AWS services are affected by this, consult the following resources:

- **Amazon DynamoDB:** [What is the consistency model of Amazon DynamoDB?](#) in the *DynamoDB FAQ*, and [Read Consistency](#) in the Amazon DynamoDB Developer Guide.
- **Amazon EC2:** [EC2 Eventual Consistency](#) in the *Amazon EC2 API Reference*.
- **Amazon EMR:** [Ensuring Consistency When Using Amazon S3 and Amazon Elastic MapReduce for ETL Workflows](#) in the AWS Big Data Blog
- **Amazon Redshift:** [Managing Data Consistency](#) in the *Amazon Redshift Database Developer Guide*
- **Amazon S3:** [Amazon S3 Data Consistency Model](#) in the *Amazon Simple Storage Service Developer Guide*

I Am Not Authorized to Perform: iam>DeleteVirtualMFADevice

You might receive the following error when you attempt to assign or remove a virtual MFA device for yourself or others:

```
User: arn:aws:iam::123456789012:user/Diego is not authorized to perform:  
iam>DeleteVirtualMFADevice on resource: arn:aws:iam::123456789012:mfa/Diego with an  
explicit deny
```

This could happen if someone previously began assigning a virtual MFA device to a user in the IAM console and then cancelled the process. This creates an MFA device for the user in IAM but never activates it. You must delete the existing MFA device before you can associate a new device with the user.

AWS recommends a policy that allows a user to delete their own virtual MFA device only if they are authenticated using MFA. For more information, see [AWS: Allows MFA-Authenticated IAM Users to Manage Their Own Credentials on the My Security Credentials Page \(p. 398\)](#).

To fix this issue, an administrator should **not** edit policy permissions. Instead, the administrator must use the AWS CLI or AWS API to remove the existing but deactivated device.

To delete an existing but deactivated MFA device

1. View the virtual MFA devices in your account.
 - AWS CLI: [aws iam list-virtual-mfa-devices](#)
 - AWS API: [ListVirtualMFADevices](#)
2. In the response, locate the ARN of the virtual device for the user you are trying to fix.
3. Delete the device.
 - AWS CLI: [aws iam delete-virtual-mfa-device](#)
 - AWS API: [DeleteVirtualMFADevice](#)

Troubleshoot IAM Policies

A [policy \(p. 356\)](#) is an entity in AWS that, when attached to an identity or resource, defines their permissions. AWS evaluates these policies when a principal, such as a user, makes a request. Permissions in the policies determine whether the request is allowed or denied. Policies are stored in AWS as JSON documents that are attached to principals as *identity-based policies* or to resources as *resource-based policies*. You can attach an identity-based policy to a principal (or identity), such as an IAM group, user, or role. Identity-based policies include AWS managed policies, customer managed policies, and inline policies. You can create and edit customer managed policies in the AWS Management Console using the **Visual editor** tab or the **JSON** tab. When you view a policy in the AWS Management Console, you can see

a summary of the permissions that are granted by that policy. You can use the visual editor and policy summaries to help you diagnose and fix common errors encountered while managing IAM policies.

Keep in mind that all IAM policies are stored using syntax that begins with the rules of [JavaScript Object Notation \(JSON\)](#). You do not have to understand this syntax to create or manage your policies. You can create and edit a policy using the visual editor in the AWS Management Console. To learn more about JSON syntax in IAM policies, see [Grammar of the IAM JSON Policy Language \(p. 651\)](#).

Troubleshooting IAM Policy Topics

- [Troubleshoot Using the Visual Editor \(p. 548\)](#)
 - [Policy Restructuring \(p. 548\)](#)
 - [Choosing a Resource ARN in the Visual Editor \(p. 549\)](#)
 - [Denying Permissions in the Visual Editor \(p. 549\)](#)
 - [Specifying Multiple Services in the Visual Editor \(p. 550\)](#)
 - [Reducing the Size of Your Policy in the Visual Editor \(p. 550\)](#)
 - [Fixing Unrecognized Services, Actions, or Resource Types in the Visual Editor \(p. 550\)](#)
- [Troubleshoot Using Policy Summaries \(p. 551\)](#)
 - [Missing Policy Summary \(p. 551\)](#)
 - [Policy Summary Includes Unrecognized Services, Actions, or Resource Types \(p. 552\)](#)
 - [Service Does Not Support IAM Policy Summaries \(p. 553\)](#)
 - [My Policy Does Not Grant the Expected Permissions \(p. 554\)](#)
- [Troubleshoot Policy Management \(p. 557\)](#)
 - [Attaching or Detaching a Policy in an IAM Account \(p. 557\)](#)
 - [Changing Policies for Your IAM Identities Based on Their Activity \(p. 557\)](#)
- [Troubleshoot JSON Policy Documents \(p. 558\)](#)
 - [More Than One JSON Policy Object \(p. 558\)](#)
 - [More Than One JSON Statement Element \(p. 558\)](#)
 - [More Than One Effect, Action, or Resource Element in a JSON Statement Element \(p. 559\)](#)
 - [Missing JSON Version Element \(p. 561\)](#)

Troubleshoot Using the Visual Editor

When you create or edit a customer managed policy, you can use information in the **Visual editor** tab to help you troubleshoot errors in your policy. To view an example of using the visual editor to create a policy, see [the section called “Controlling Access to Identities” \(p. 383\)](#).

Policy Restructuring

When you create a policy, AWS validates, processes, and transforms the policy before storing it. When AWS returns the policy in response to a user query or displays it in the console, AWS transforms the policy back into a human-readable format without changing the permissions granted by the policy. This can result in differences in what you see in the policy visual editor or **JSON** tab: Visual editor permission blocks can be added, removed, or reordered, and content within a block can be optimized. In the **JSON** tab, insignificant white space can be removed, and elements within JSON maps can be reordered. In addition, AWS account IDs within the principal elements can be replaced by the ARN of the AWS account root user. Because of these possible changes, you should not compare JSON policy documents as strings.

When you create a customer managed policy in the AWS Management Console, you can choose to work entirely in the **JSON** tab. If you never make any changes in the **Visual editor** tab and choose **Review policy** from the **JSON** tab, the policy is less likely to be restructured. However, if you create a policy and

use the **Visual editor** tab to make any modifications, or if you choose **Review policy** from the **Visual editor** tab, then IAM might restructure the policy to optimize its appearance in the visual editor.

This restructuring exists only in your editing session and is not saved automatically.

If your policy is restructured in your editing session, IAM determines whether to save the restructuring based on the following situations:

On this tab	If you edit your policy	And then choose <i>Review policy</i> from this tab	When you choose <i>Save changes</i>
Visual editor	Edited	Visual editor	The policy is restructured
Visual editor	Edited	JSON	The policy is restructured
Visual editor	Not Edited	Visual editor	The policy is restructured
JSON	Edited	Visual editor	The policy is restructured
JSON	Edited	JSON	The policy structure is not changed
JSON	Not Edited	JSON	The policy structure is not changed

IAM might restructure complex policies or policies that have permission blocks or statements that allow multiple services, resource types, or condition keys.

Choosing a Resource ARN in the Visual Editor

When you create or edit a policy using the visual editor, you must first choose a service, and then choose actions from that service. If the service and actions that you selected support choosing [specific resources \(p. 388\)](#), then the visual editor lists the supported resource types. You can then choose **Add ARN** to provide the details about your resource. You can choose from the following options for adding an ARN for a resource type.

- **Use the ARN builder** – Based on the resource type, you might see different fields to build your ARN. You can also choose **Any** to provide permissions for any value for the specified setting. For example, if you selected the Amazon EC2 **Read** access level group, then the actions in your policy support the **instance** resource type. You must provide the **Region**, **Account**, and **InstanceId** values for your resource. If you provide your account ID but choose **Any** for the Region and instance ID, then the policy grants permissions to any instance in your account.
- **Type or paste the ARN** – You can specify resources by their [Amazon Resource Name \(ARN\)](#). You can include a wildcard character (*) in any field of the ARN (between each pair of colons). For more information, see [IAM JSON Policy Elements: Resource \(p. 609\)](#).

Denying Permissions in the Visual Editor

By default, the policy that you create using the visual editor allows the actions that you choose. To deny the chosen actions instead, choose **Switch to deny permissions**. Because requests are *denied by default*, we recommend as a security best practice that you allow permissions to only those actions and

resources that a user needs. This is sometimes called "whitelisting." You should create a statement to deny permissions ("blacklisting") only if you want to override a permission separately that is allowed by another statement or policy. We recommend that you limit the number of deny permissions to a minimum because they can increase the difficulty of troubleshooting permissions. For more information about how IAM evaluates policy logic, see [Policy Evaluation Logic \(p. 636\)](#).

Note

By default, only the AWS account root user has access to all the resources in that account. So if you are not signed in as the root user, you must have permissions granted by a policy.

Specifying Multiple Services in the Visual Editor

When you use the visual editor to construct a policy, you can select only one service at a time. This is a best practice because the visual editor then allows you to choose from the actions for that one service. You then choose from the resources supported by that service and the selected actions. This makes it easier to create and troubleshoot your policy.

If you are familiar with the JSON syntax, you can also use a wildcard character (*) to manually specify multiple services. For example, type `Code*` to provide permissions for all services beginning with Code, such as CodeBuild and CodeCommit. However, you must then type the actions and resource ARNs to complete your policy. Additionally, when you save your policy, it might be [restructured \(p. 548\)](#) to include each service in a separate permission block.

Alternatively, to use JSON syntax (such as wildcards) for services, create, edit, and save your policy using the **JSON** tab.

Reducing the Size of Your Policy in the Visual Editor

When you use the visual editor to create a policy, IAM creates a JSON document to store your policy. You can view this document by switching to the **JSON** tab. If this JSON document exceeds the size limit of a policy, the visual editor displays an error message and does not allow you to review and save your policy. To view the IAM limitation on the size of a managed policy, see [IAM and STS Character Limits \(p. 583\)](#).

To reduce the size of your policy in the visual editor, edit your policy or move permission blocks to another policy. The error message includes the number of characters that your policy document contains, and you can use this information to help you reduce the size of your policy.

Fixing Unrecognized Services, Actions, or Resource Types in the Visual Editor

When you create or edit a policy in the visual editor, you might see a warning that your policy includes an unrecognized service, action, or resource type.

Note

IAM reviews service names, actions, and resource types for services that support policy summaries. However, your policy summary might include a resource value or condition that does not exist. Always test your policies with the [policy simulator \(p. 448\)](#).

If your policy includes unrecognized services, actions or resource types, one of the following errors has occurred:

- **Preview service** – Services that are in preview do not support the visual editor. If you are participating in the preview, you can ignore the warning and continue, though you must manually type the actions and resource ARNs to complete your policy. Alternatively, you can choose the **JSON** tab to type or paste a JSON policy document.
- **Custom service** – Custom services do not support the visual editor. If you are using a custom service, you can ignore the warning and continue, though you must manually type the actions and resource ARNs to complete your policy. Alternatively, you can choose the **JSON** tab to type or paste a JSON policy document.

- **Service does not support the visual editor** – If your policy includes a generally available (GA) service that does not support the visual editor, you can ignore the warning and continue, though you must manually type the actions and resource ARNs to complete your policy. Alternatively, you can choose the **JSON** tab to type or paste a JSON policy document.

Generally available services are services that are released publicly and are not preview or custom services. If an unrecognized service is generally available and the name is spelled correctly, then the service does not support the visual editor. To learn how to request visual editor or policy summary support for a GA service, see [Service Does Not Support IAM Policy Summaries \(p. 553\)](#).

- **Action does not support the visual editor** – If your policy includes a supported service with an unsupported action, you can ignore the warning and continue, though you must manually type the resource ARNs to complete your policy. Alternatively, you can choose the **JSON** tab to type or paste a JSON policy document.

If your policy includes a supported service with an unsupported action, then the service does not fully support the visual editor. To learn how to request visual editor or policy summary support for a GA service, see [Service Does Not Support IAM Policy Summaries \(p. 553\)](#).

- **Resource type does not support the visual editor** – If your policy includes a supported action with an unsupported resource type, you can ignore the warning and continue. However, IAM cannot confirm that you have included resources for all of your selected actions, and you might see additional warnings.
- **Typo** – When you manually type a service, action, or resource in the visual editor, you can create a policy that includes a typo. As a best practice, use the visual editor by selecting from the list of services and actions, and then complete the resource section according to the prompts. However, if a service does not fully support the visual editor, you might have to manually type parts of your policy.

If you are certain that your policy contains none of the errors above, then your policy might include a typo. Check for misspelled service, action, and resource type names. For example, you might use `s2` instead of `s3` and `ListMyBuckets` instead of `ListAllMyBuckets`. Another common action typo is the inclusion of unnecessary text in ARNs, such as `arn:aws:s3: : :*`, or missing colons in actions, such as `AWSAuthRuntimeService.AuthenticatePassword`. You can evaluate a policy that might include typos by choosing **Review policy** to review the policy summary and confirm whether the policy provides the permissions you intended.

Troubleshoot Using Policy Summaries

You can diagnose and resolve issues related to policy summaries.

Missing Policy Summary

The IAM console includes *policy summary* tables that describe the access level, resources, and conditions that are allowed or denied for each service in a policy. Policies are summarized in three tables: the [policy summary \(p. 492\)](#), the [service summary \(p. 502\)](#), and the [action summary \(p. 507\)](#). The *policy summary* table includes a list of services and summaries of the permissions that are defined by the chosen policy. You can view the [policy summary \(p. 491\)](#) for any policies that are attached to a user on the **Users** page. You can view the policy summary for managed policies on the **Policies** page. If AWS is unable to render a summary for a policy, then you see the JSON policy document instead of the summary, and receive the following error:

A summary for this policy cannot be generated. You can still view or edit the JSON policy document.

If your policy does not include a summary, one of the following errors has occurred:

- **Unsupported policy element** – IAM does not support generating policy summaries for policies that include one of the following [policy elements \(p. 598\)](#):
 - Principal

- `NotPrincipal`
- `NotResource`
- **No policy permissions** – If a policy does not provide any effective permissions, then the policy summary cannot be generated. For example, if a policy includes a single statement with the element `"NotAction": "*"`, then it grants access to all actions except "all actions" (*). This means it grants Deny or Allow access to nothing.

Note

You must be careful when using these policy elements such as `NotPrincipal`, `NotAction`, and `NotResource`. For information about using policy elements, see [IAM JSON Policy Elements Reference \(p. 598\)](#).

You can create a policy that does not provide effective permissions if you provide mismatched services and resources. This can occur if you specify actions in one service and resources from another service. In this case, the policy summary does appear. The only indication that there is a problem is that the resource column in the summary can include a resource from a different service. If this column includes a mismatched resource, then you should review your policy for errors. To better understand your policies, always test them with the [policy simulator \(p. 448\)](#).

Policy Summary Includes Unrecognized Services, Actions, or Resource Types

In the IAM console, if a [policy summary \(p. 491\)](#) includes a warning symbol (), then the policy might include an unrecognized service, action or resource type. To learn about warnings within a policy summary, see [Policy Summary \(List of Services\) \(p. 492\)](#).

Note

IAM reviews service names, actions, and resource types for services that support policy summaries. However, your policy summary might include a resource value or condition that does not exist. Always test your policies with the [policy simulator \(p. 448\)](#).

If your policy includes unrecognized services, actions or resource types, one of the following errors has occurred:

- **Preview service** – Services that are in preview do not support policy summaries.
- **Custom service** – Custom services do not support policy summaries.
- **Service does not support summaries** – If your policy includes a generally available (GA) service that does not support policy summaries, then the service is included in the **Unrecognized services** section of the policy summary table. Generally available services are services that are released publicly and are not preview or custom services. If an unrecognized service is generally available and the name is spelled correctly, then the service does not support IAM policy summaries. To learn how to request policy summary support for a GA service, see [Service Does Not Support IAM Policy Summaries \(p. 553\)](#).
- **Action does not support summaries** – If your policy includes a supported service with an unsupported action, then the action is included in the **Unrecognized actions** section of the service summary table. To learn about warnings within a service summary, see [Service Summary \(List of Actions\) \(p. 502\)](#).
- **Resource type does not support summaries** – If your policy includes a supported action with an unsupported resource type, then the resource is included in the **Unrecognized resource types** section of the service summary table. To learn about warnings within a service summary, see [Service Summary \(List of Actions\) \(p. 502\)](#).
- **Typo** – Because the policy validator in AWS checks only that the JSON is syntactically correct, you can create a policy that includes a typo. If you are certain that your policy contains none of the errors above, then your policy might include a typo. Check for misspelled service, action, and resource type names. For example, you might use `s2` instead of `s3` and

ListMyBuckets instead of ListAllMyBuckets. Another common action typo is the inclusion of unnecessary text in ARNs, such as arn:aws:s3: : *, or missing colons in actions, such as AWSAuthRuntimeService.AuthenticatePassword. You can evaluate a policy that might include typos by using the [policy simulator \(p. 448\)](#) to confirm whether the policy provides the permissions you intended.

Service Does Not Support IAM Policy Summaries

When a generally available (GA) service or action is not recognized by IAM policy summaries or the visual editor, it is possible that the service does not support these features. Generally available services are services that are released publicly and are not previewed or custom services. If an unrecognized service is generally available and the name is spelled correctly, then the service does not support these features. If your policy includes a supported service with an unsupported action, then the service does not fully support IAM policy summaries.

To request that a service add IAM policy summary or visual editor support

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. Locate the policy that includes the unsupported service:
 - If the policy is a managed policy, choose **Policies** in the navigation pane. In the list of policies, choose the name of the policy that you want to view.
 - If the policy is an inline policy attached to the user, choose **Users** in the navigation pane. In the list of users, choose the name of the user whose policy you want to view. In the table of policies for the user, expand the header for the policy summary that you want to view.
3. In the left side on the AWS Management Console footer, choose **Feedback**. In the **Tell us about your experience:** box, type **I request that the <ServiceName> service add support for IAM policy summaries and the visual editor.** If you want more than one service to support summaries, type **I request that the <ServiceName1>, <ServiceName2>, and <ServiceName3> services add support for IAM policy summaries and the visual editor.**

To request that a service add IAM policy summary support for a missing action

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. Locate the policy that includes the unsupported service:
 - If the policy is a managed policy, choose **Policies** in the navigation pane. In the list of policies, choose the name of the policy that you want to view.
 - If the policy is an inline policy attached to the user, choose **Users** in the navigation pane. In the list of users, choose the name of the user whose policy you want to view. In the table of policies for the user, choose the name of the policy that you want to view to expand the policy summary.
3. In the policy summary, choose the name of the service that includes an unsupported action.
4. In the left side on the AWS Management Console footer, choose **Feedback**. In the **Tell us about your experience:** box, type **I request that the <ServiceName> service add IAM policy summary and the visual editor support for the <ActionName> action.** If you want to report more than one unsupported action, type **I request that the <ServiceName> service add IAM policy summary and the visual editor support for the <ActionName1>, <ActionName2>, and <ActionName3> actions.**

To request that a different service includes missing actions, repeat the last three steps.

My Policy Does Not Grant the Expected Permissions

To assign permissions to a user, group, role, or resource, you create a *policy*, which is a document that defines permissions. The policy document includes the following elements:

- **Effect** – whether the policy allows or denies access
- **Action** – the list of actions that are allowed or denied by the policy
- **Resource** – the list of resources on which the actions can occur
- **Condition (Optional)** – the circumstances under which the policy grants permission

To learn about these and other policy elements, see [IAM JSON Policy Elements Reference \(p. 598\)](#).

To grant access, your policy must define an action with a supported resource. If your policy also includes a condition, that condition must include a [global condition key \(p. 665\)](#) or must apply to the action. To learn which resources are supported by an action, see the [AWS documentation](#) for your service. To learn which conditions are supported by an action, see [Actions, Resources, and Condition Keys for AWS Services \(p. 686\)](#).

To learn whether your policy defines an action, resource, or condition that does not grant permissions, you can view the [policy summary \(p. 492\)](#) for your policy using the IAM console at <https://console.aws.amazon.com/iam/>. You can use policy summaries to identify and correct problems in your policy.

There are several reasons why an element might not grant permissions despite being defined in the IAM policy:

- [An action is defined without an applicable resource \(p. 554\)](#)
- [A resource is defined without an applicable action \(p. 555\)](#)
- [A condition is defined without an applicable action \(p. 555\)](#)

To view examples of policy summaries that include warnings, see [the section called “Policy Summary \(List of Services\)” \(p. 492\)](#).

An Action Is Defined Without an Applicable Resource

The policy below defines all `ec2:Describe*` actions and defines a specific resource. None of the `ec2:Describe` actions are granted because none of these actions support resource-level permissions. Resource-level permissions mean that the action supports resources using [ARNs](#) in the policy's [Resource \(p. 609\)](#) element. If an action does not support resource-level permissions, then that statement in the policy must use a wildcard (*) in the Resource element. To learn which services support resource-level permissions, see [AWS Services That Work with IAM \(p. 586\)](#).

```
{  
    "Version": "2012-10-17",  
    "Statement": [{  
        "Effect": "Allow",  
        "Action": "ec2:Describe*",  
        "Resource": "arn:aws:ec2:us-east-2:ACCOUNT-ID:instance/*"  
    }]  
}
```

This policy does not provide any permissions, and the policy summary includes the following error:

`This policy does not grant any permissions. To grant access, policies must have an action that has an applicable resource or condition.`

To fix this policy, you must use * in the Resource element.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {"Effect": "Allow",  
         "Action": "ec2:Describe*",  
         "Resource": "*"}  
    ]  
}
```

A Resource Is Defined Without an Applicable Action

The policy below defines an Amazon S3 bucket resource but does not include an S3 action that can be performed on that resource. This policy also grants full access to all Amazon CloudFront actions.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {"Effect": "Allow",  
         "Action": "cloudfront:*",  
         "Resource": [  
             "arn:aws:cloudfront:*",  
             "arn:aws:s3:::examplebucket"  
         ]}  
    ]  
}
```

This policy provides permissions for all CloudFront actions. But because the policy defines the S3 examplebucket resource without defining any S3 actions, the policy summary includes the following warning:

This policy defines some actions, resources, or conditions that do not provide permissions. To grant access, policies must have an action that has an applicable resource or condition.

To fix this policy to provide S3 bucket permissions, you must define S3 actions that can be performed on a bucket resource.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {"Effect": "Allow",  
         "Action": [  
             "cloudfront:*",  
             "s3:CreateBucket",  
             "s3>ListBucket*",  
             "s3:PutBucket*",  
             "s3:GetBucket*"  
         ],  
         "Resource": [  
             "arn:aws:cloudfront:*",  
             "arn:aws:s3:::examplebucket"  
         ]}  
    ]  
}
```

Alternately, to fix this policy to provide only CloudFront permissions, remove the S3 resource.

A Condition Is Defined Without an Applicable Action

The policy below defines two Amazon S3 actions for all S3 resources, if the S3 prefix equals custom and the version ID equals 1234. However, the s3:VersionId condition key is used for object version

tagging and is not supported by the defined bucket actions. To learn which conditions are supported by an action, see [Actions, Resources, and Condition Keys for AWS Services \(p. 686\)](#) and follow the link to the service documentation for condition keys.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3>ListBucketVersions",
                "s3>ListBucket"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "s3:prefix": [
                        "custom"
                    ],
                    "s3:VersionId": [
                        "1234"
                    ]
                }
            }
        }
    ]
}
```

This policy provides permissions for the `s3>ListBucketVersions` action and the `s3>ListBucket` action if the bucket name includes the `custom` prefix. But because the `s3:VersionId` condition is not supported by any of the defined actions, the policy summary includes the following error:

This policy does not grant any permissions. To grant access, policies must have an action that has an applicable resource or condition.

To fix this policy to use S3 object version tagging, you must define an S3 action that supports the `s3:VersionId` condition key.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3>ListBucketVersions",
                "s3>ListBucket",
                "s3>GetObjectVersion"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "s3:prefix": [
                        "custom"
                    ],
                    "s3:VersionId": [
                        "1234"
                    ]
                }
            }
        }
    ]
}
```

This policy provides permissions for every action and condition in the policy. However, the policy still does not provide any permissions because there is no case where a single action matches both conditions. Instead, you must create two separate statements that each include only actions with the conditions to which they apply.

To fix this policy, create two statements. The first statement includes the actions that support the `s3:prefix` condition, and the second statement includes the actions that support the `s3:VersionId` condition.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": [  
                "s3>ListBucketVersions",  
                "s3>ListBucket"  
            ],  
            "Resource": "*",  
            "Condition": {  
                "StringEquals": {  
                    "s3:prefix": "custom"  
                }  
            }  
        },  
        {  
            "Effect": "Allow",  
            "Action": "s3GetObjectVersion",  
            "Resource": "*",  
            "Condition": {  
                "StringEquals": {  
                    "s3:VersionId": "1234"  
                }  
            }  
        }  
    ]  
}
```

Troubleshoot Policy Management

You can diagnose and resolve issues relating to policy management.

Attaching or Detaching a Policy in an IAM Account

Some AWS managed policies are linked to a service. These policies are used only with a [service-linked role](#) (p. 176) for that service. In the IAM console, when you view the **Summary** page for a policy, the page includes a banner to indicate that the policy is linked to a service. You cannot attach this policy to a user, group, or role within IAM. When you create a service-linked role for the service, this policy is automatically attached to your new role. Because the policy is required, you cannot detach the policy from the service-linked role.

Changing Policies for Your IAM Identities Based on Their Activity

You can update policies for your IAM identities (users, groups, and roles) based on their activity. To do this, view your account's events in CloudTrail **Event history**. CloudTrail event logs include detailed event information that you can use to change the policy's permissions. You might find that a user or role is attempting to perform an action in AWS and that request is denied. In that case, you can consider whether the user or role should have permission to perform the action. If so, you can add the action and even the ARN of the resource that they attempted to access to their policy. Alternatively, if the user or

role has permissions that they are not using, you might consider removing those permissions from their policy. Make sure that your policies grant the [least privilege \(p. 61\)](#) that is needed to perform only the necessary actions. For more information about using CloudTrail, see [Viewing CloudTrail Events in the CloudTrail Console](#) in the [AWS CloudTrail User Guide](#).

Troubleshoot JSON Policy Documents

You can diagnose and resolve issues relating to JSON policy documents.

More Than One JSON Policy Object

An IAM policy must consist of one and only one JSON object. You denote an object by placing {} braces around it. Although you can nest other objects within a JSON object by embedding additional {} braces within the outer pair, a policy can contain only one outermost pair of {} braces. The following example is incorrect because it contains two objects at the top level (called out in red):

```
{  
    "Version": "2012-10-17",  
    "Statement":  
    {  
        "Effect": "Allow",  
        "Action": "ec2:Describe*",  
        "Resource": "*"  
    }  
}  
  
{  
    "Statement": {  
        "Effect": "Allow",  
        "Action": "s3:*",  
        "Resource": "arn:aws:s3:::my-bucket/*"  
    }  
}
```

You can, however, meet the intention of the previous example with the use of correct policy grammar. Instead of including two complete policy objects each with its own Statement element, you can combine the two blocks into a single Statement element. The Statement element has an array of two objects as its value, as shown in the following example (called out in bold):

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": "ec2:Describe*",  
            "Resource": "*"  
        },  
        {  
            "Effect": "Allow",  
            "Action": "s3:*",  
            "Resource": "arn:aws:s3:::my-bucket/*"  
        }  
    ]  
}
```

More Than One JSON Statement Element

This error might at first appear to be a variation on the previous section. However, syntactically it is a different type of error. The following example has only one policy object as denoted by a single pair of {} braces at the top level. However, that object contains two Statement elements within it.

An IAM policy must contain only one `Statement` element, consisting of the name (`Statement`) appearing to the left of a colon, followed by its value on the right. The value of a `Statement` element must be an object, denoted by {} braces, containing one `Effect` element, one `Action` element, and one `Resource` element. The following example is incorrect because it contains two `Statement` elements in the policy object (called out in *red*):

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "ec2:Describe*",
        "Resource": "*"
    },
    "Statement": {
        "Effect": "Allow",
        "Action": "s3:*",
        "Resource": "arn:aws:s3:::my-bucket/*"
    }
}
```

A value object can be an array of multiple value objects. To solve this problem, combine the two `Statement` elements into one element with an object array, as shown in the following example (called out in **bold**):

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:Describe*",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": "arn:aws:s3:::my-bucket/*"
        }
    ]
}
```

The value of the `Statement` element is an object array. The array in this example consists of two objects, each of which is by itself a correct value for a `Statement` element. Each object in the array is separated by commas.

More Than One Effect, Action, or Resource Element in a JSON Statement Element

On the value side of the `Statement` name/value pair, the object must consist of only one `Effect` element, one `Action` element, and one `Resource` element. The following policy is incorrect because it has two `Effect` elements in the value object of the `Statement`:

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Deny",
        "Effect": "Allow",
        "Action": "ec2:*",
        "Resource": "*"
    }
}
```

Note

The policy engine does not allow such errors in new or edited policies. However, the policy engine continues to permit policies that were saved before the engine was updated. The behavior of existing policies with the error is as follows:

- Multiple `Effect` elements: only the last `Effect` element is observed. The others are ignored.
- Multiple `Action` elements: all `Action` elements are combined internally and treated as if they were a single list.
- Multiple `Resource` elements: all `Resource` elements are combined internally and treated as if they were a single list.

The policy engine does not allow you to save any policy with syntax errors. You must correct the errors in the policy before you can save it. The [Policy Validator \(p. 448\)](#) tool can help you to find all older policies with errors and can recommend corrections for them.

In each case, the solution is to remove the incorrect extra element. For `Effect` elements, this is straightforward: if you want the previous example to *deny* permissions to Amazon EC2 instances, then you must remove the line `"Effect": "Allow"`, from the policy, as follows:

```
{  
    "Version": "2012-10-17",  
    "Statement": {  
        "Effect": "Deny",  
        "Action": "ec2:*",  
        "Resource": "*"  
    }  
}
```

However, if the duplicate element is `Action` or `Resource`, then the resolution can be more complicated. You might have multiple actions that you want to allow (or deny) permission to, or you might want to control access to multiple resources. For example, the following example is incorrect because it has multiple `Resource` elements (called out in *red*):

```
{  
    "Version": "2012-10-17",  
    "Statement": {  
        "Effect": "Allow",  
        "Action": "s3:*",  
        "Resource        "Resource    }  
}
```

Each of the required elements in a `Statement` element's value object can be present only once. The solution is to place each value in an array. The following example illustrates this by making the two separate resource elements into one `Resource` element with an array as the value object (called out in **bold**):

```
{  
    "Version": "2012-10-17",  
    "Statement": {  
        "Effect": "Allow",  
        "Action": "s3:*",  
        "Resource": [  
            "arn:aws:s3:::my-bucket",  
            "arn:aws:s3:::my-bucket/*"  
        ]  
    }  
}
```

```
}
```

Missing JSON Version Element

A `Version` policy element is different from a policy version. The `Version` policy element is used within a policy and defines the version of the policy language. A policy version, on the other hand, is created when you make changes to a customer managed policy in IAM. The changed policy doesn't overwrite the existing policy. Instead, IAM creates a new version of the managed policy. To learn more about the `Version` policy element see [IAM JSON Policy Elements: Version \(p. 598\)](#). To learn more about policy versions, see [the section called "Versioning IAM Policies" \(p. 465\)](#).

As AWS features evolve, new capabilities are added to IAM policies to support those features. Sometimes, an update to the policy syntax includes a new version number. If you use newer features of the policy grammar in your policy, then you must tell the policy parsing engine which version you are using. The default policy version is "2008-10-17." If you want to use any policy feature that was introduced later, then you must specify the version number that supports the feature you want. We recommend that you *always* include the latest policy syntax version number, which is currently "`Version": "2012-10-17"`. For example, the following policy is incorrect because it uses a policy variable `$(...)` in the ARN for a resource. But it fails to specify a policy syntax version that supports policy variables (called out in **red**):

```
{
  "Statement": [
    {
      "Action": "iam:*AccessKey*",
      "Effect": "Allow",
      "Resource": "arn:aws:iam::123456789012:user/${aws:username}"
    }
  ]
}
```

Adding a `Version` element at the top of the policy with the value `2012-10-17`, the first IAM API version that supports policy variables, solves this problem (called out in **bold**):

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": "iam:*AccessKey*",
      "Effect": "Allow",
      "Resource": "arn:aws:iam::123456789012:user/${aws:username}"
    }
  ]
}
```

Troubleshooting U2F Security Keys

Use the information here to help you diagnose common issues that you might encounter when working with U2F security keys.

Topics

- [I Can't Enable My U2F Security Key \(p. 562\)](#)
- [I Can't Sign in Using My U2F Security Key \(p. 562\)](#)
- [I Lost or Broke My U2F Key \(p. 562\)](#)

- [Other Issues \(p. 563\)](#)

I Can't Enable My U2F Security Key

Consult the following solutions depending on your status as an IAM user or system administrator

IAM Users

If you can't enable your U2F security key, check the following:

- Are you using a supported configuration?

For information on devices and browsers you can use with U2F and AWS, see [Supported Configurations for Using U2F Security Keys \(p. 129\)](#).

- Are you using Mozilla Firefox?

Most Firefox versions that support U2F do not enable support by default. To enable support for U2F in Firefox, do the following:

1. From the Firefox address bar, type `about:config`.
2. In the Search bar of the screen that opens, type `u2f`.
3. Choose `security.webauth.u2f` and change its value to `true`.

- Are you using any browser plugins?

AWS does not support the use of plugins to add U2F browser support. Instead, use a browser that offers native support of the U2F standard.

Even if you're using a supported browser, you may have a plugin that is incompatible with U2F. An incompatible plugin may prevent you from enabling and using your U2F security key. You should disable any plugins that might be incompatible and restart your browser. Then retry enabling the U2F security key.

- Do you have the appropriate permissions?

If you don't have any of the above compatibility issues, you may not have the appropriate permissions. Contact your system administrator.

System Administrators

If you're an administrator and your IAM users can't enable their U2F security keys despite using a supported configuration, make sure they have the appropriate permissions. For a detailed example, see [Tutorial: Enable Your Users to Configure Their Own Credentials and MFA Settings \(p. 56\)](#).

I Can't Sign in Using My U2F Security Key

If you're an IAM user and you can't sign in to the AWS Management Console using U2F, first see [Supported Configurations for Using U2F Security Keys \(p. 129\)](#). If you're using a supported configuration but cannot sign in, contact your system administrator for assistance.

I Lost or Broke My U2F Key

Only *one* MFA device (virtual, U2F security key, or hardware) is assigned to a user at a time. Replacing a U2F security key is similar to replacing a hardware MFA device. For information on what to do if you lose or break any type of MFA device, see [What If an MFA Device Is Lost or Stops Working? \(p. 144\)](#).

Other Issues

If you have an issue with U2F security keys that is not covered here, do one of the following:

- IAM users: Contact your system administrator.
- AWS account root users: Contact [AWS Support](#).

Troubleshooting IAM Roles

Use the information here to help you diagnose and fix common issues that you might encounter when working with IAM roles.

Topics

- [I Can't Assume a Role \(p. 563\)](#)
- [A New Role Appeared in My AWS Account \(p. 564\)](#)
- [I Can't Edit or Delete a Role in My AWS Account \(p. 564\)](#)
- [I'm Not Authorized to Perform: iam:PassRole \(p. 565\)](#)
- [Why Can't I Assume a Role with a 12-Hour Session? \(AWS CLI, AWS API\) \(p. 565\)](#)
- [My Role Has a Policy That Allows Me to Perform an Action, But I Get "Access Denied" \(p. 565\)](#)

I Can't Assume a Role

Check the following:

- Make sure to use the exact name of your role, because role names are case sensitive.
- Verify that your IAM policy grants you permission to call `sts:AssumeRole` for the role that you want to assume. The `Action` element of your IAM policy must allow you to call the `AssumeRole` action. In addition, the `Resource` element of your IAM policy must specify the role that you want to assume. For example, the `Resource` element can specify a role by its Amazon Resource Name (ARN) or by a wildcard (*). For example, at least one policy applicable to you must grant permissions similar to the following:

```
"Effect": "Allow",
"Action": "sts:AssumeRole",
"Resource": "arn:aws:iam::account_id_number:role/role-name-you-want-to-assume"
```

- Verify that your IAM identity is tagged with any tags that the IAM policy requires. For example, in the following policy permissions, the `Condition` element requires that you, as the principal requesting to assume the role, must have a specific tag. You must be tagged with `department = HR` or `department = CS`. Otherwise, you cannot assume the role. To learn about tagging IAM users and roles, see [the section called "Tagging Users and Roles" \(p. 295\)](#).

```
"Effect": "Allow",
"Action": "sts:AssumeRole",
"Resource": "*",
"Condition": {"StringEquals": {"aws:PrincipalTag/department": [
    "HR",
    "CS"
]}}
```

- Verify that you meet all the conditions that are specified in the role's trust policy. A `Condition` can specify an expiration date, an external ID, or that a request must come only from specific IP addresses.

Consider the following example: If the current date is any time after the specified date, then the policy never matches and cannot grant you the permission to assume the role.

```
"Effect": "Allow",
"Action": "sts:AssumeRole",
"Resource": "arn:aws:iam::account_id_number:role/role-name-you-want-to-assume"
"Condition": {
    "DateLessThan" : {
        "aws:CurrentTime" : "2016-05-01T12:00:00Z"
    }
}
```

- Verify that the AWS account from which you are calling `AssumeRole` is a trusted entity for the role that you are assuming. Trusted entities are defined as a `Principal` in a role's trust policy. The following example is a trust policy that is attached to the role that you want to assume. In this example, the account ID with the IAM user that you signed in with must be 123456789012. If your account number is not listed in the `Principal` element of the role's trust policy, then you cannot assume the role. It does not matter what permissions are granted to you in access policies. Note that the example policy limits permissions to actions that occur between July 1, 2017 and December 31, 2017 (UTC), inclusive. If you log in before or after those dates, then the policy does not match, and you cannot assume the role.

```
"Effect": "Allow",
"Principal": { "AWS": "arn:aws:iam::123456789012:root" },
>Action": "sts:AssumeRole",
"Condition": {
    "DateGreaterThanOrEqual": {"aws:CurrentTime": "2017-07-01T00:00:00Z"},
    "DateLessThan": {"aws:CurrentTime": "2017-12-31T23:59:59Z"}
}
```

A New Role Appeared in My AWS Account

Some AWS services require that you use a unique type of service role that is linked directly to the service. This [service-linked role \(p. 176\)](#) is predefined by the service and includes all the permissions that the service requires. This makes setting up a service easier because you don't have to manually add the necessary permissions. For general information about service-linked roles, see [Using Service-Linked Roles \(p. 221\)](#).

You might already be using a service when it begins supporting service-linked roles. If so, you might receive an email telling you about a new role in your account. This role includes all the permissions that the service needs to perform actions on your behalf. You don't need to take any action to support this role. However, you should not delete the role from your account. Doing so could remove permissions that the service needs to access AWS resources. You can view the service-linked roles in your account by going to the **IAM Roles** page of the IAM console. Service-linked roles appear with **(Service-linked role)** in the **Trusted entities** column of the table.

For information about which services support service-linked roles, see [AWS Services That Work with IAM \(p. 586\)](#) and look for the services that have **Yes** in the **Service-Linked Role** column. For information about using the service-linked role for a service, choose the **Yes** link.

I Can't Edit or Delete a Role in My AWS Account

You cannot delete or edit the permissions for a [service-linked role \(p. 176\)](#) in IAM. These roles include predefined trusts and permissions that are required by the service in order to perform actions on your behalf. You can use the IAM console, AWS CLI, or API to edit only the description of a service-linked role. You can view the service-linked roles in your account by going to the **IAM Roles** page in the console. Service-linked roles appear with **(Service-linked role)** in the **Trusted entities** column of the table. A

banner on the role's **Summary** page also indicates that the role is a service-linked role. You can manage and delete these roles only through the linked service, if that service supports the action. Be careful when modifying or deleting a service-linked role because doing so could remove permissions that the service needs to access AWS resources.

For information about which services support service-linked roles, see [AWS Services That Work with IAM \(p. 586\)](#) and look for the services that have **Yes** in the **Service-Linked Role** column.

I'm Not Authorized to Perform: iam:PassRole

When you create a service-linked role, you must have permission to pass that role to the service. Some services automatically create a service-linked role in your account when you perform an action in that service. For example, Amazon EC2 Auto Scaling creates the `AWSServiceRoleForAutoScaling` service-linked role for you the first time that you create an Auto Scaling group. If you try to create an Auto Scaling group without the `PassRole` permission, you receive the following error:

```
ClientError: An error occurred (AccessDenied) when calling the
PutLifecycleHook operation: User: arn:aws:sts::111122223333:assumed-role/
Testrole/Diego is not authorized to perform: iam:PassRole on resource:
arn:aws:iam::111122223333:role/aws-service-role/autoscaling.amazonaws.com/
AWSServiceRoleForAutoScaling
```

To fix this error, ask your administrator to add the `iam:PassRole` permission for you.

To learn which services support service-linked roles, see [AWS Services That Work with IAM \(p. 586\)](#). To learn whether a service automatically creates a service-linked role for you, choose the **Yes** link to view the service-linked role documentation for the service.

Why Can't I Assume a Role with a 12-Hour Session? (AWS CLI, AWS API)

When you use the AWS STS `AssumeRole*` API or `assume-role*` CLI operations to assume a role, you can specify a value for the `DurationSeconds` parameter. You can specify a value from 900 seconds (15 minutes) up to the **Maximum CLI/API session duration** setting for the role. If you specify a value higher than this setting, the operation fails. This setting can have a maximum value of 12 hours. For example, if you specify a session duration of 12 hours, but your administrator set the maximum session duration to 6 hours, your operation fails. To learn how to view the maximum value for your role, see [View the Maximum Session Duration Setting for a Role \(p. 255\)](#).

If you use [role chaining \(p. 177\)](#) (using a role to assume a second role), your session is limited to a maximum of one hour. If you then use the `DurationSeconds` parameter to provide a value greater than one hour, the operation fails.

My Role Has a Policy That Allows Me to Perform an Action, But I Get "Access Denied"

Your role session might be limited by session policies. When you [request temporary security credentials \(p. 309\)](#) programmatically using AWS STS, you can optionally pass inline or managed [session policies \(p. 358\)](#). Session policies are advanced policies that you pass as a parameter when you programmatically create a temporary credential session for a role. You can pass a single JSON inline session policy document using the `Policy` parameter. You can use the `PolicyArns` parameter to specify up to 10 managed session policies. The resulting session's permissions are the intersection of the role's identity-based policies and the session policies. Alternatively, if your administrator or a custom program provides you with temporary credentials, they might have included a session policy to limit your access.

Troubleshooting Amazon EC2 and IAM

Use the information here to help you troubleshoot and fix access denied or other issues that you might encounter when working with Amazon EC2 and IAM.

Topics

- [When Attempting to Launch an Instance, I Don't See the Role I Expected to See in the Amazon EC2 Console IAM Role List \(p. 566\)](#)
- [The Credentials on My Instance Are for the Wrong Role \(p. 566\)](#)
- [When I Attempt to Call the AddRoleToInstanceProfile, I Get an AccessDenied Error \(p. 567\)](#)
- [Amazon EC2: When I Attempt to Launch an Instance with a Role, I Get an AccessDenied Error \(p. 567\)](#)
- [I Can't Access the Temporary Security Credentials on My EC2 Instance \(p. 567\)](#)
- [What Do the Errors from the info Document in the IAM Subtree Mean? \(p. 568\)](#)

When Attempting to Launch an Instance, I Don't See the Role I Expected to See in the Amazon EC2 Console IAM Role List

Check the following:

- If you are signed in as an IAM user, verify that you have permission to call `ListInstanceProfiles`. For information about the permissions necessary to work with roles, see "Permissions Required for Using Roles with Amazon EC2" in [Using an IAM Role to Grant Permissions to Applications Running on Amazon EC2 Instances \(p. 269\)](#). For information about adding permissions to a user, see [Managing IAM Policies \(p. 442\)](#).

If you cannot modify your own permissions, you must contact an administrator who can work with IAM in order to update your permissions.

- If you created a role by using the IAM CLI or API, verify that you created an instance profile and added the role to that instance profile. Also, if you name your role and instance profile differently, you won't see the correct role name in the list of IAM roles in the Amazon EC2 console. The **IAM Role** list in the Amazon EC2 console lists the names of instance profiles, not the names of roles. You will have to select the name of the instance profile that contains the role you want. For details about instance profiles, see [Using Instance Profiles \(p. 276\)](#).

Note

If you use the IAM console to create roles, you don't need to work with instance profiles. For each role that you create in the IAM console, an instance profile is created with the same name as the role, and the role is automatically added to that instance profile. An instance profile can contain only one IAM role, and that limit cannot be increased.

The Credentials on My Instance Are for the Wrong Role

The role in the instance profile might have been replaced recently. If so, your application will need to wait for the next automatically scheduled credential rotation before credentials for your role become available.

When I Attempt to Call the AddRoleToInstanceProfile, I Get an AccessDenied Error

If you are making requests as an IAM user, verify that you have the following permissions:

- `iam:AddRoleToInstanceProfile` with the resource matching the instance profile ARN (for example, `arn:aws:iam::999999999999:instance-profile/ExampleInstanceProfile`).

For more information about the permissions necessary to work with roles, see "How Do I Get Started?" in [Using an IAM Role to Grant Permissions to Applications Running on Amazon EC2 Instances \(p. 269\)](#). For information about adding permissions to a user, see [Managing IAM Policies \(p. 442\)](#).

Amazon EC2: When I Attempt to Launch an Instance with a Role, I Get an AccessDenied Error

Check the following:

- Launch an instance without an instance profile. This will help ensure that the problem is limited to IAM roles for Amazon EC2 instances.
- If you are making requests as an IAM user, verify that you have the following permissions:
 - `ec2:RunInstances` with a wildcard resource ("*")
 - `iam:PassRole` with the resource matching the role ARN (for example, `arn:aws:iam::999999999999:role/ExampleRoleName`)
- Call the IAM `GetInstanceProfile` action to ensure that you are using a valid instance profile name or a valid instance profile ARN. For more information, see [Using IAM roles with Amazon EC2 instances](#).
- Call the IAM `GetInstanceProfile` action to ensure that the instance profile has a role. Empty instance profiles will fail with an `AccessDenied` error. For more information about creating a role, see [Creating IAM Roles \(p. 228\)](#).

For more information about the permissions necessary to work with roles, see "How Do I Get Started?" in [Using an IAM Role to Grant Permissions to Applications Running on Amazon EC2 Instances \(p. 269\)](#). For information about adding permissions to a user, see [Managing IAM Policies \(p. 442\)](#).

I Can't Access the Temporary Security Credentials on My EC2 Instance

To access temporary security credentials on your EC2 instance, you must first use the IAM console to create a role. Then you launch an EC2 instance that uses that role and examine the running instance. For more information, see [How Do I Get Started?](#) in [Using an IAM Role to Grant Permissions to Applications Running on Amazon EC2 Instances \(p. 269\)](#).

If you still can't access your temporary security credentials on your EC2 instance, check the following:

- Can you access another part of the instance metadata service (IMDS)? If not, check that you have no firewall rules blocking access to requests to the IMDS.

```
[ec2-user@domU-12-31-39-0A-8D-DE ~]$ GET http://169.254.169.254/latest/meta-data/hostname; echo
```

- Does the `iam` subtree of the IMDS exist? If not, verify that your instance has an IAM instance profile associated with it by calling the EC2 `DescribeInstances` API operation or using the `aws ec2 describe-instances` CLI command.

```
[ec2-user@domU-12-31-39-0A-8D-DE ~]$ GET http://169.254.169.254/latest/meta-data/iam; echo
```

- Check the `info` document in the IAM subtree for an error. If you have an error, see [What Do the Errors from the `info` Document in the IAM Subtree Mean? \(p. 568\)](#) for more information.

```
[ec2-user@domU-12-31-39-0A-8D-DE ~]$ GET http://169.254.169.254/latest/meta-data/iam/info; echo
```

What Do the Errors from the `info` Document in the IAM Subtree Mean?

The `iam/info` Document Indicates "Code": "InstanceProfileNotFound"

Your IAM instance profile has been deleted and Amazon EC2 can no longer provide credentials to your instance. You must attach a valid instance profile to your Amazon EC2 instance.

If an instance profile with that name exists, check that the instance profile wasn't deleted and another was created with the same name:

1. Call the IAM `GetInstanceProfile` operation to get the `InstanceProfileId`.
2. Call the Amazon EC2 `DescribeInstances` operation to get the `IamInstanceProfileId` for the instance.
3. Verify that the `InstanceProfileId` from the IAM operation matches the `IamInstanceProfileId` from the Amazon EC2 operation.

If the IDs are different, then the instance profile attached to your instances is no longer valid. You must attach a valid instance profile to the instance.

The `iam/info` Document Indicates a Success but Indicates "Message": "Instance Profile does not contain a role..."

The role has been removed from the instance profile by the IAM `RemoveRoleFromInstanceProfile` action. You can use the IAM `AddRoleToInstanceProfile` action to attach a role to the instance profile. Your application will need to wait until the next scheduled refresh to access the credentials for the role.

The `iam/security-credentials/[role-name]` Document Indicates "Code": "AssumeRoleUnauthorizedAccess"

Amazon EC2 does not have permission to assume the role. Permission to assume the role is controlled by the trust policy attached to the role, like the example that follows. Use the IAM `UpdateAssumeRolePolicy` API to update the trust policy.

```
{"Version": "2012-10-17", "Statement": [{"Effect": "Allow", "Principal": {"Service": ["ec2.amazonaws.com"]}, "Action": ["sts:AssumeRole"]}]}
```

Your application will need to wait until the next automatically scheduled refresh to access the credentials for the role.

Troubleshooting Amazon S3 and IAM

Use the information here to help you troubleshoot and fix issues that you might encounter when working with Amazon S3 and IAM.

How Do I Grant Anonymous Access to an Amazon S3 Bucket?

You use an Amazon S3 bucket policy that specifies a wildcard (*) in the `principal` element, which means anyone can access the bucket. With anonymous access, anyone (including users without an AWS account) will be able to access the bucket. For a sample policy, see [Example Cases for Amazon S3 Bucket Policies](#) in the *Amazon Simple Storage Service Developer Guide*.

I'm Signed in as an AWS Account Root User; Why Can't I Access an Amazon S3 Bucket Under My Account?

In some cases, you might have an IAM user with full access to IAM and Amazon S3. If the IAM user assigns a bucket policy to an Amazon S3 bucket and doesn't specify the AWS account root user as a principal, the root user is denied access to that bucket. However, as the root user, you can still access the bucket. To do that, modify the bucket policy to allow root user access from the Amazon S3 console or the AWS CLI.

Troubleshooting SAML 2.0 Federation with AWS

Use the information here to help you diagnose and fix issues that you might encounter when working with SAML 2.0 and federation with IAM.

Topics

- [Error: Your Request Included an Invalid SAML Response. To Logout, Click Here. \(p. 570\)](#)
- [Error: RoleSessionName is Required in AuthnResponse \(Service: AWSSecurityTokenService; Status Code: 400; Error Code: InvalidIdentityToken\) \(p. 570\)](#)
- [Error: Not Authorized to Perform sts:AssumeRoleWithSAML \(Service: AWSSecurityTokenService; Status Code: 403; Error Code: AccessDenied\) \(p. 570\)](#)
- [Error: RoleSessionName in AuthnResponse Must Match \[a-zA-Z_0-9+=,.@-\]{2,64} \(Service: AWSSecurityTokenService; Status Code: 400; Error Code: InvalidIdentityToken\) \(p. 571\)](#)
- [Error: Response Signature Invalid \(Service: AWSSecurityTokenService; Status Code: 400; Error Code: InvalidIdentityToken\) \(p. 571\)](#)
- [Error: Failed to Assume Role: Issuer Not Present in Specified Provider \(Service: AWSOpenIdDiscoveryService; Status Code: 400; Error Code: AuthSamlInvalidSamlResponseException\) \(p. 571\)](#)

- [Error: Could Not Parse Metadata. \(p. 571\)](#)
- [Error: Specified Provider Doesn't Exist. \(p. 572\)](#)
- [Error: Requested DurationSeconds Exceeds MaxSessionDuration Set for This Role. \(p. 572\)](#)
- [How to View a SAML Response in Your Browser for Troubleshooting \(p. 572\)](#)

Error: Your Request Included an Invalid SAML Response. To Logout, Click Here.

This error can occur when the SAML response from the identity provider does not include an attribute with the Name set to `https://aws.amazon.com/SAML/Attributes/Role`. The attribute must contain one or more `AttributeValue` elements, each containing a comma-separated pair of strings:

- The ARN of a role that the user can be mapped to
- The ARN of the SAML provider

For more information, see [Configuring SAML Assertions for the Authentication Response \(p. 206\)](#). To view the SAML response in your browser, follow the steps listed in [How to View a SAML Response in Your Browser for Troubleshooting \(p. 572\)](#).

Error: RoleSessionName is Required in AuthnResponse (Service: AWSSecurityTokenService; Status Code: 400; Error Code: InvalidIdentityToken)

This error can occur when the SAML response from the identity provider does not include an attribute with the Name set to `https://aws.amazon.com/SAML/Attributes/RoleSessionName`. The attribute value is an identifier for the user and is typically a user ID or an email address.

For more information, see [Configuring SAML Assertions for the Authentication Response \(p. 206\)](#). To view the SAML response in your browser, follow the steps listed in [How to View a SAML Response in Your Browser for Troubleshooting \(p. 572\)](#).

Error: Not Authorized to Perform sts:AssumeRoleWithSAML (Service: AWSSecurityTokenService; Status Code: 403; Error Code: AccessDenied)

This error can occur if the IAM role specified in the SAML response is misspelled or does not exist. Make sure to use the exact name of your role, because role names are case sensitive. Correct the name of the role in the SAML service provider configuration.

You are allowed access only if your role trust policy includes the `sts:AssumeRoleWithSAML` action. If your SAML assertion is configured to use the [PrincipalTag attribute \(p. 208\)](#), your trust policy must also include the `sts:TagSession` action. For more information about session tags, see [Passing Session Tags in AWS STS \(p. 299\)](#).

This error can also occur if the federated users do not have permissions to assume the role. The role must have a trust policy that specifies the ARN of the IAM SAML identity provider as the Principal. The role

also contains conditions that control which users can assume the role. Ensure that your users meet the requirements of the conditions.

This error can also occur if the SAML response does not include a `Subject` containing a `NameID`.

For more information, see [Establish Permissions in AWS for Federated Users](#) and [Configuring SAML Assertions for the Authentication Response \(p. 206\)](#). To view the SAML response in your browser, follow the steps listed in [How to View a SAML Response in Your Browser for Troubleshooting \(p. 572\)](#).

Error: RoleSessionName in AuthnResponse Must Match [a-zA-Z_0-9+=,.@-]{2,64} (Service: AWSecurityTokenService; Status Code: 400; Error Code: InvalidIdentityToken)

This error can occur if the `RoleSessionName` attribute value is too long or contains invalid characters. The maximum valid length is 64 characters.

For more information, see [Configuring SAML Assertions for the Authentication Response \(p. 206\)](#). To view the SAML response in your browser, follow the steps listed in [How to View a SAML Response in Your Browser for Troubleshooting \(p. 572\)](#).

Error: Response Signature Invalid (Service: AWSecurityTokenService; Status Code: 400; Error Code: InvalidIdentityToken)

This error can occur when federation metadata of the identity provider does not match the metadata of the IAM identity provider. For example, the metadata file for the identity service provider might have changed to update an expired certificate. Download the updated SAML metadata file from your identity service provider. Then update it in the AWS identity provider entity that you define in IAM with the `aws iam update-saml-provider` cross-platform CLI command or the `Update-IAMSAMLProvider` PowerShell cmdlet.

Error: Failed to Assume Role: Issuer Not Present in Specified Provider (Service: AWSOpenIdDiscoveryService; Status Code: 400; Error Code: AuthSamlInvalidSamlResponseException)

This error can occur if the issuer in the SAML response does not match the issuer declared in the federation metadata file. The metadata file was uploaded to AWS when you created the identity provider in IAM.

Error: Could Not Parse Metadata.

This error can occur if your metadata file is not formatted properly.

When you [create or manage a SAML identity provider \(p. 202\)](#) in the AWS Management Console, you must retrieve the SAML metadata document from your identity provider. This metadata file includes the issuer's name, expiration information, and keys that can be used to validate the SAML authentication

response (assertions) that are received from the IdP. The metadata file must be encoded in UTF-8 format without a byte order mark (BOM). Also, the x.509 certificate that is included as part of the SAML metadata document must use a key size of at least 1024 bits. If the key size is smaller, the IdP creation fails with an "Unable to parse metadata" error. To remove the BOM, you can encode the file as UTF-8 using a text editing tool, such as Notepad++.

Error: Specified Provider Doesn't Exist.

This error can occur if the name of the provider that you specify in the SAML assertion does not match the name of the provider configured in IAM. For more information about viewing the provider name, see [Creating IAM SAML Identity Providers \(p. 201\)](#).

Error: Requested DurationSeconds Exceeds MaxSessionDuration Set for This Role.

This error can occur if you assume a role from the AWS CLI or API.

When you use the [assume-role-with-saml](#) CLI or [AssumeRoleWithSAML](#) API operations to assume a role, you can specify a value for the `DurationSeconds` parameter. You can specify a value from 900 seconds (15 minutes) up to the maximum session duration setting for the role. If you specify a value higher than this setting, the operation fails. For example, if you specify a session duration of 12 hours, but your administrator set the maximum session duration to 6 hours, your operation fails. To learn how to view the maximum value for your role, see [View the Maximum Session Duration Setting for a Role \(p. 255\)](#).

How to View a SAML Response in Your Browser for Troubleshooting

The following procedures describe how to view the SAML response from your service provider from in your browser when troubleshooting a SAML 2.0-related issue.

For all browsers, go to the page where you can reproduce the issue. Then follow the steps for the appropriate browser:

Topics

- [Google Chrome \(p. 572\)](#)
- [Mozilla Firefox \(p. 573\)](#)
- [Apple Safari \(p. 573\)](#)
- [Microsoft Internet Explorer \(p. 573\)](#)
- [What to Do with the Base64-Encoded SAML Response \(p. 573\)](#)

Google Chrome

To view a SAML response in Chrome

These steps were tested using version 54.0.2840.87m. If you use another version, you might need to adapt the steps accordingly.

1. Press **F12** to start the developer console.
2. Select the **Network** tab, and then select **Preserve log**.
3. Reproduce the issue.
4. Look for a **SAML Post** in the developer console pane. Select that row, and then view the **Headers** tab at the bottom. Look for the **SAMLResponse** attribute that contains the encoded request.

Mozilla Firefox

To view a SAML response in Firefox

This procedure was tested on version 37.0.2 of Mozilla Firefox. If you use another version, you might need to adapt the steps accordingly.

1. Press **F12** to start the developer console.
2. In the upper right of the developer tools window, click options (the small gear icon). Under **Common Preferences** select **Enable persistent logs**.
3. Select the **Network** tab.
4. Reproduce the issue.
5. Look for a **POST SAML** in the table. Select that row. In the **Form Data** window on the right, select the **Params** tab and find the **SAMLResponse** element.

Apple Safari

To view a SAML response in Safari

These steps were tested using version 8.0.6 (10600.6.3). If you use another version, you might need to adapt the steps accordingly.

1. Enable Web Inspector in Safari. Open the **Preferences** window, select the **Advanced** tab, and then select **Show Develop menu in the menu bar**.
2. Now you can open Web Inspector. Click **Develop**, then select **Show Web Inspector**.
3. Select the **Resources** tab.
4. Reproduce the issue.
5. Look for a **saml-signin.aws.amazon.com** request.
6. Scroll down to find Request Data with the name **SAMLResponse**. The associated value is the Base64-encoded response.

Microsoft Internet Explorer

To view a SAML response in Internet Explorer

The best way to analyze network traffic in Internet Explorer is through the use of a third-party tool.

- Follow the steps at <http://social.technet.microsoft.com/wiki/contents/articles/3286.ad-fs-2-0-how-to-use-fiddler-web-debugger-to-analyze-a-ws-federation-passive-sign-in.aspx> to download and install Fiddler and capture the data.

What to Do with the Base64-Encoded SAML Response

Once you find the Base64-encoded SAML response element in your browser, copy it and use your favorite Base-64 decoding tool to extract the XML tagged response.

Security Tip

Because the SAML response data that you are viewing might contain sensitive security data, we recommend that you do not use an *online* base64 decoder. Instead use a tool installed on your local computer that does not send your SAML data over the network.

Built-in option for Windows systems (PowerShell):

```
PS C:\> [System.Text.Encoding]::UTF8.GetString([System.Convert]::FromBase64String("base64encodedtext"))
```

Built-in option for MacOS and Linux systems:

```
$ echo "base64encodedtext" | base64 --decode
```

Reference Information for AWS Identity and Access Management

Use the topics in this section to find detailed reference material for various aspects of IAM and AWS STS.

Topics

- [IAM Identifiers \(p. 575\)](#)
- [IAM and STS Limits \(p. 581\)](#)
- [AWS Services That Work with IAM \(p. 586\)](#)
- [IAM JSON Policy Reference \(p. 597\)](#)

IAM Identifiers

IAM uses a few different identifiers for users, groups, roles, policies, and server certificates. This section describes the identifiers and when you use each.

Topics

- [Friendly Names and Paths \(p. 575\)](#)
- [IAM ARNs \(p. 576\)](#)
- [Unique Identifiers \(p. 579\)](#)

Friendly Names and Paths

When you create a user, a role, a group, or a policy, or when you upload a server certificate, you give it a friendly name. Examples include Bob, TestApp1, Developers, ManageCredentialsPermissions, or ProdServerCert.

If you are using the IAM API or AWS Command Line Interface (AWS CLI) to create IAM entities, you can also give the entity an optional path. You can use a single path, or nest multiple paths as if they were a folder structure. For example, you could use the nested path /division_abc/subdivision_xyz/product_1234/engineering/ to match your company's organizational structure. You could then create a policy to allow all users in that path to access the policy simulator API. To view this policy, see [IAM: Access the Policy Simulator API Based on User Path \(p. 429\)](#). For additional examples of how you might use paths, see [IAM ARNs \(p. 576\)](#).

When you use AWS CloudFormation to create resources, you can specify a path for users, groups, and roles, but not policies.

Just because you give a user and group the same path doesn't automatically put that user in that group. For example, you might create a Developers group and specify its path as /division_abc/subdivision_xyz/product_1234/engineering/. Just because you create a user named Bob and give him that same path doesn't automatically put Bob in the Developers group. IAM doesn't enforce any boundaries between users or groups based on their paths. Users with different paths can use the same resources (assuming they've been granted permission to those resources). For information about limitations on names, see [IAM and STS Limits \(p. 581\)](#).

IAM ARNs

Most resources have a friendly name (for example, a user named Bob or a group named Developers). However, the permissions policy language requires you to specify the resource or resources using the following *Amazon Resource Name (ARN)* format.

```
arn:partition:service:region:account:resource
```

Where:

- *partition* identifies the partition that the resource is in. For standard AWS Regions, the partition is aws. If you have resources in other partitions, the partition is aws-*partitionname*. For example, the partition for resources in the China (Beijing) Region is aws-cn. You cannot [delegate access \(p. 293\)](#) between accounts in different partitions.
- *service* identifies the AWS product. For IAM resources, this is always iam.
- *region* is the Region the resource resides in. For IAM resources, this is always kept blank.
- *account* is the AWS account ID with no hyphens (for example, 123456789012).
- *resource* is the portion that identifies the specific resource by name.

You can specify IAM and AWS STS ARNs using the following syntax. The Region portion of the ARN is blank because IAM resources are global.

Syntax:

```
arn:aws:iam::account-id:root
arn:aws:iam::account-id:user/user-name-with-path
arn:aws:iam::account-id:group/group-name-with-path
arn:aws:iam::account-id:role/role-name-with-path
arn:aws:iam::account-id:policy/policy-name-with-path
arn:aws:iam::account-id:instance-profile/instance-profile-name-with-path
arn:aws:sts::account-id:federated-user/user-name
arn:aws:sts::account-id:assumed-role/role-name/role-session-name
arn:aws:iam::account-id:mfa/virtual-device-name-with-path
arn:aws:iam::account-id:u2f/u2f-token-id
arn:aws:iam::account-id:server-certificate/certificate-name-with-path
arn:aws:iam::account-id:saml-provider/provider-name
arn:aws:iam::account-id:oidc-provider/provider-name
```

Many of the following examples include paths in the resource part of the ARN. Paths cannot be created or manipulated in the AWS Management Console. To use paths you must work with the resource by using the AWS API, the AWS CLI, or the Tools for Windows PowerShell.

Examples:

```
arn:aws:iam::123456789012:root
arn:aws:iam::123456789012:user/JohnDoe
arn:aws:iam::123456789012:user/division_abc/subdivision_xyz/JaneDoe
arn:aws:iam::123456789012:group/Developers
arn:aws:iam::123456789012:group/division_abc/subdivision_xyz/product_A/Developers
arn:aws:iam::123456789012:role/S3Access
arn:aws:iam::123456789012:role/application_abc/component_xyz/S3Access
arn:aws:iam::123456789012:policy/UsersManageOwnCredentials
arn:aws:iam::123456789012:policy/division_abc/subdivision_xyz/UsersManageOwnCredentials
arn:aws:iam::123456789012:instance-profile/Webserver
arn:aws:sts::123456789012:federated-user/JohnDoe
arn:aws:sts::123456789012:assumed-role/Accounting-Role/JaneDoe
arn:aws:iam::123456789012:mfa/JaneDoeMFA
arn:aws:iam::123456789012:u2f/user/JohnDoe/default (U2F security key)
```

```
arn:aws:iam::123456789012:server-certificate/ProdServerCert
arn:aws:iam::123456789012:server-certificate/division_abc/subdivision_xyz/ProdServerCert
arn:aws:iam::123456789012:saml-provider/ADFSProvider
arn:aws:iam::123456789012:oidc-provider/GoogleProvider
```

The following examples provide more detail to help you understand the ARN format for different types of IAM and AWS STS resources.

- An IAM user in the account:

```
arn:aws:iam::123456789012:user/JohnDoe
```

- Another user with a path reflecting an organization chart:

```
arn:aws:iam::123456789012:user/division_abc/subdivision_xyz/JaneDoe
```

- An IAM group:

```
arn:aws:iam::123456789012:group/Developers
```

- An IAM group with a path:

```
arn:aws:iam::123456789012:group/division_abc/subdivision_xyz/product_A/Developers
```

- An IAM role:

```
arn:aws:iam::123456789012:role/S3Access
```

- A managed policy:

```
arn:aws:iam::123456789012:policy/ManageCredentialsPermissions
```

- An instance profile that can be associated with an EC2 instance:

```
arn:aws:iam::123456789012:instance-profile/Webserver
```

- A federated user identified in IAM as "Paulo":

```
arn:aws:sts::123456789012:federated-user/Paulo
```

- The active session of someone assuming the role of "Accounting-Role", with a role session name of "Mary":

```
arn:aws:sts::123456789012:assumed-role/Accounting-Role/Mary
```

- The multi-factor authentication device assigned to the user named Jorge:

```
arn:aws:iam::123456789012:mfa/Jorge
```

- A server certificate:

```
arn:aws:iam::123456789012:server-certificate/ProdServerCert
```

- A server certificate with a path that reflects an organization chart:

```
arn:aws:iam::123456789012:server-certificate/division_abc/subdivision_xyz/ProdServerCert
```

- Identity providers (SAML and OIDC):

```
arn:aws:iam::123456789012:saml-provider/ADFSProvider  
arn:aws:iam::123456789012:oidc-provider/GoogleProvider
```

Another important ARN is the root user ARN. Although this is not an IAM resource, you should be familiar with the format of this ARN. It is often used in the [Principal element \(p. 601\)](#) of a policy.

- The AWS account - the account itself:

```
arn:aws:iam::123456789012:root
```

The following example shows a policy that you could assign to Richard to allow him to manage his own access keys. Notice that the resource is the IAM user Richard.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "ManageRichardAccessKeys",  
            "Effect": "Allow",  
            "Action": [  
                "iam:*AccessKey*",  
                "iam:GetUser"  
            ],  
            "Resource": "arn:aws:iam::*:user/division_abc/subdivision_xyz/Richard"  
        },  
        {  
            "Sid": "ListForConsole",  
            "Effect": "Allow",  
            "Action": "iam>ListUsers",  
            "Resource": "*"  
        }  
    ]  
}
```

Note

When you use ARNs to identify resources in an IAM policy, you can include *policy variables*. Policy variables can include placeholders for runtime information (such as the user's name) as part of the ARN. For more information, see [IAM Policy Elements: Variables and Tags \(p. 628\)](#)

You can use wildcards in the *resource* portion of the ARN to specify multiple users or groups or policies. For example, to specify all users working on product_1234, you would use:

```
arn:aws:iam::123456789012:user/division_abc/subdivision_xyz/product_1234/*
```

Let's say you have users whose names start with the string app_. You could refer to them all with the following ARN.

```
arn:aws:iam::123456789012:user/division_abc/subdivision_xyz/product_1234/app_*
```

To specify all users, groups, or policies in your AWS account, use a wildcard after the `user/`, `group/`, or `policy` part of the ARN, respectively.

```
arn:aws:iam::123456789012:user/*
```

```
arn:aws:iam::123456789012:group/*
arn:aws:iam::123456789012:policy/*
```

Don't use a wildcard in the `user/`, `group/`, or `policy` part of the ARN. For example, the following is not allowed:

```
arn:aws:iam::123456789012:u*
```

Example Use of Paths and ARNs for a Project-Based Group

Paths cannot be created or manipulated in the AWS Management Console. To use paths you must work with the resource by using the AWS API, the AWS CLI, or the Tools for Windows PowerShell.

In this example, Jules in the `Marketing_Admin` group creates a project-based group within the `/marketing/` path. Jules assigns users from different parts of the company to the group. This example illustrates that a user's path isn't related to the groups the user is in.

The marketing group has a new product they'll be launching, so Jules creates a new group in the `/marketing/` path called `Widget_Launch`. Jules then assigns the following policy to the group, which gives the group access to objects in the part of the `example_bucket` that is designated to this particular launch.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:*",
      "Resource": "arn:aws:s3:::example_bucket/marketing/newproductlaunch/widget/*"
    },
    {
      "Effect": "Allow",
      "Action": "s3>ListBucket*",
      "Resource": "arn:aws:s3:::example_bucket",
      "Condition": {"StringLike": {"s3:prefix": "marketing/newproductlaunch/widget/*"}}
    }
  ]
}
```

Jules then assigns the users who are working on this launch to the group. This includes Patricia and Eli from the `/marketing/` path. It also includes Chris and Chloe from the `/sales/` path, and Alice and Jim from the `/legal/` path.

Unique Identifiers

When IAM creates a user, group, role, policy, instance profile, or server certificate, it assigns to each entity a unique ID that looks like this:

AIDAJQABLZS4A3QDU576Q

For the most part, you use friendly names and [ARNs](#) when you work with IAM entities. That way you don't need to know the unique ID for a specific entity. However, the unique ID can sometimes be useful when it isn't practical to use friendly names.

One example pertains to reusing friendly names in your AWS account. Within your account, a friendly name for a user, group, or policy must be unique. For example, you might create an IAM user named David. Your company uses Amazon S3 and has a bucket with folders for each employee. The bucket

has a resource-based policy (a bucket policy) that lets users access only their own folders in the bucket. Suppose that the employee named David leaves your company and you delete the corresponding IAM user. But later another employee named David starts and you create a new IAM user named David. If the bucket policy specifies the David IAM user, the policy allows the new David to access information that was left by the former David.

However, every IAM user has a unique ID, even if you create a new IAM user that reuses a friendly name that you deleted before. In the example, the old IAM user David and the new IAM user David have different unique IDs. You can create resource policies for Amazon S3 buckets that grant access by unique ID and not just by user name. Doing so reduces the chance that you could inadvertently grant access to information that an employee should not have.

Another example where user IDs can be useful is if you maintain your own database (or other store) of IAM user information. The unique ID can provide a unique identifier for each IAM user you create. This is so even if over time you have IAM users that reuse a name, as in the previous example.

Understanding Unique ID Prefixes

IAM uses the following prefixes to indicate what type of entity each unique ID applies to.

Prefix	Entity Type
AAGA	Action group
ACCA	Context-specific credential
AGPA	Group
AIDA	IAM user
AIPA	Amazon EC2 instance profile
AKIA	Access key
ANPA	Managed policy
ANVA	Version in a managed policy
APKA	Public key
AROA	Role
ASCA	Certificate
ASIA	Temporary (AWS STS) keys

Getting the Unique Identifier

The unique ID for an IAM entity is not available in the IAM console. To get the unique ID, you can use the following AWS CLI commands or IAM API calls.

AWS CLI:

- [get-caller-identity](#)
- [get-group](#)
- [get-role](#)
- [get-user](#)

- [get-policy](#)
- [get-instance-profile](#)
- [get-server-certificate](#)

IAM API:

- [GetCallerIdentity](#)
- [GetGroup](#)
- [GetRole](#)
- [GetUser](#)
- [GetPolicy](#)
- [GetInstanceProfile](#)
- [GetServerCertificate](#)

IAM and STS Limits

Objects in AWS Identity and Access Management (IAM) and AWS Security Token Service (STS) have size limitations. These services also limit how you name an object, the number of objects you can create, and the number of characters you can use when you pass an object.

Note

To get account-level information about IAM usage and limit quotas, use the [GetAccountSummary](#) API operation or the [get-account-summary](#) AWS CLI command.

IAM Name Limits

The following are restrictions on IAM names:

- Policy documents can contain only the following Unicode characters: horizontal tab (U+0009), linefeed (U+000A), carriage return (U+000D), and characters in the range U+0020 to U+00FF.
- Names of users, groups, roles, policies, instance profiles, and server certificates must be alphanumeric, including the following common characters: plus (+), equal (=), comma (,), period (.), at (@), underscore (_), and hyphen (-).
- Names of users, groups, roles, and instance profiles must be unique within the account. They are not distinguished by case, for example, you cannot create groups named both **ADMINS** and **admins**.
- The external ID value that a third party uses to assume a role must have a minimum of 2 characters and a maximum of 1,224 characters. The value must be alphanumeric without white space. It can also include the following symbols: plus (+), equal (=), comma (,), period (.), at (@), colon (:), forward slash (/), and hyphen (-). For more information about the external ID, see [How to Use an External ID When Granting Access to Your AWS Resources to a Third Party \(p. 233\)](#).
- Path names must begin and end with a forward slash (/).
- Policy names for [inline policies \(p. 364\)](#) must be unique to the user, group, or role they are embedded in. The names can contain any Basic Latin (ASCII) characters minus the following reserved characters: backward slash (\), forward slash (/), asterisk (*), question mark (?), and white space. These characters are reserved according to [RFC 3986](#).
- User passwords (login profiles) can contain any Basic Latin (ASCII) characters.
- AWS account ID aliases must be unique across AWS products, and must be alphanumeric following DNS naming conventions. An alias must be lowercase, it must not start or end with a hyphen, it cannot contain two consecutive hyphens, and it cannot be a 12-digit number.

For a list of Basic Latin (ASCII) characters, go to the [Library of Congress Basic Latin \(ASCII\) Code Table](#).

IAM Object Limits

AWS allows you to request an increase to default IAM entity limits. To learn how to request a limit increase to these default limits, see [AWS Service Limits](#) in the *Amazon Web Services General Reference* documentation.

Default limits for IAM entities:

Resource	Default Limit
Customer managed policies in an AWS account	1500
Groups in an AWS account	300
Roles in an AWS account	1000
Managed policies attached to an IAM role	10
Managed policies attached to an IAM user	10
Virtual MFA devices (assigned or unassigned) in an AWS account	Equal to the user quota for the account
Instance profiles in an AWS account	1000
Server certificates stored in an AWS account	20

You cannot request a limit increase for the following limits.

Limits for IAM entities:

Resource	Limit
Access keys assigned to an IAM user	2
Access keys assigned to the AWS account root user	2
Aliases for an AWS account	1
Groups an IAM user can be a member of	10
IAM users in a group	Equal to the user quota for the account
Users in an AWS account	5000 (If you need to add a large number of users, consider using temporary security credentials (p. 307) .)
Identity providers (IdPs) associated with an IAM SAML provider object	10
Keys per SAML provider	10
Login profiles for an IAM user	1
Managed policies attached to an IAM group	10
Permissions boundaries for an IAM user	1
Permissions boundaries for an IAM role	1

Resource	Limit
MFA devices in use by an IAM user	1
MFA devices in use by the AWS account root user	1
Roles in an instance profile	1
SAML providers in an AWS account	100
Signing certificates assigned to an IAM user	2
SSH public keys assigned to an IAM user	5
Tags that can be attached to an IAM role	50
Tags that can be attached to an IAM user	50
Versions of a managed policy that can be stored	5

IAM and STS Character Limits

The following are the maximum character counts and size limits for IAM and AWS STS:

Description	Limit
Path	512 characters
User name	64 characters
Group name	128 characters
Role name	64 characters Important If you intend to use a role with the Switch Role feature in the AWS console, then the combined Path and RoleName cannot exceed 64 characters.
Tag key	128 characters This character limit applies to user tags, role tags, and session tags (p. 299) .
Tag value	256 characters This character limit applies to user tags, role tags, and session tags (p. 299) . Tag values can be empty. That is, tag values can have a length of 0 characters.
Instance profile name	128 characters
Unique IDs created by IAM, for example:	128 characters
<ul style="list-style-type: none"> • User IDs that begin with AIDA • Group IDs that begin with AGPA 	

Description	Limit
<ul style="list-style-type: none"> Role IDs that begin with <code>AROA</code> Managed policy IDs that begin with <code>ANPA</code> Server certificate IDs that begin with <code>ASCA</code> <p>Note This is not intended to be an exhaustive list, nor is it a guarantee that IDs of a certain type begin only with the specified letter combination.</p>	
Policy name	128 characters
Password for a login profile	1 to 128 characters
Alias for an AWS account ID	3 to 63 characters
Role trust policy JSON text (the policy that determines who is allowed to assume the role)	2,048 characters
Role session name	64 characters
Role session duration	<p>12 hours</p> <p>When you assume a role from the AWS CLI or API, you can use the <code>duration-seconds</code> CLI parameter or the <code>DurationSeconds</code> API parameter to request a longer role session. You can specify a value from 900 seconds (15 minutes) up to the maximum session duration setting for the role, which can range from 1 to 12 hours. The maximum session duration setting does not limit sessions assumed by AWS services. To learn how to view the maximum value for your role, see View the Maximum Session Duration Setting for a Role (p. 255). If you don't specify a value for the <code>DurationSeconds</code> parameter, your security credentials are valid for one hour.</p>

Description	Limit
Role session policies (p. 358)	<ul style="list-style-type: none"> You can pass only one JSON policy document when you programmatically create a temporary session for a role or federated user. The size of the passed JSON policy document and all session policy ARNs combined cannot exceed 2,048 characters. You can pass multiple managed policy ARNs when you create a session. However, the size of each managed policy cannot exceed 2,048 characters. An AWS conversion compresses the passed session policies and session tags into a packed binary format that has a separate limit. You can pass session policies using the AWS CLI or AWS API. The <code>PackedPolicySize</code> response element indicates by percentage how close the policies and tags for your request are to the upper size limit.
Role session tags (p. 299)	<ul style="list-style-type: none"> Session tags must meet the tag key limit of 128 characters and the tag value limit of 256 characters. You can pass up to 50 session tags. An AWS conversion compresses the passed session policies and session tags into a packed binary format that has a separate limit. You can pass session tags using the AWS CLI or AWS API. The <code>PackedPolicySize</code> response element indicates by percentage how close the policies and tags for your request are to the upper size limit.
For inline policies (p. 364)	<p>You can add as many inline policies as you want to an IAM user, role, or group. But the total aggregate policy size (the sum size of all inline policies) per entity cannot exceed the following limits:</p> <ul style="list-style-type: none"> User policy size cannot exceed 2,048 characters. Role policy size cannot exceed 10,240 characters. Group policy size cannot exceed 5,120 characters. <p>Note IAM does not count white space when calculating the size of a policy against these limitations.</p>

Description	Limit
For managed policies (p. 364)	<ul style="list-style-type: none"> You can add up to 10 managed policies to an IAM user, role, or group. The size of each managed policy cannot exceed 6,144 characters. <p>Note IAM does not count white space when calculating the size of a policy against this limitation.</p>

AWS Services That Work with IAM

The AWS services listed below are grouped by their [AWS product categories](#) and include information about what IAM features they support:

- Service** – You can choose the name of a service to view the AWS documentation about IAM authorization and access for that service.
- Actions** – You can specify individual actions in a policy. If the service does not support this feature, then **All actions** is selected in the [visual editor \(p. 445\)](#). In a JSON policy document, you must use * in the Action element. For a list of actions in each service, see [Actions, Resources, and Condition Keys for AWS Services \(p. 686\)](#).
- Resource-level permissions** – You can use [ARNs](#) to specify individual resources in the policy. If the service does not support this feature, then **All resources** is chosen in the [policy visual editor \(p. 445\)](#). In a JSON policy document, you must use * in the Resource element. Some actions, such as List* actions, do not support specifying an ARN because they are designed to return multiple resources. If a service supports this feature for some resources but not others, it is indicated by yellow cells in the table. See the documentation for that service for more information.
- Resource-based policies** – You can attach resource-based policies to a resource within the service. Resource-based policies include a Principal element to specify which IAM identities can access that resource. For more information, see [Identity-Based Policies and Resource-Based Policies \(p. 379\)](#).
- Authorization based on tags** – You can use [resource tags](#) in the condition of a policy to control access to a resource in the service. You do this using the [aws:ResourceTag \(p. 672\)](#) global condition key or service-specific tags, such as [ec2:ResourceTag](#). For more information about defining permissions based on attributes such as tags, see [What Is ABAC for AWS? \(p. 12\)](#).
- Temporary credentials** – Users signed in with federation, a cross-account role, or a [service role \(p. 176\)](#) can access the service. Temporary security credentials are obtained by calling AWS STS API operations like [AssumeRole](#) or [GetFederationToken](#). For more information, see [Temporary Security Credentials \(p. 307\)](#).
- Service-linked roles** – A [service-linked role \(p. 176\)](#) gives the service permission to access resources in other services to complete an action on your behalf. Choose the Yes link to see the documentation for services that support these roles. For more information, see [Using Service-Linked Roles \(p. 221\)](#).
- More information** – If a service doesn't fully support a feature, you can review the footnotes for an entry to view the limitations and links to related information.

Compute Services

Service	Actions	Resource-level permissions	Resource-based policies	Authorization based on tags	Temporary credentials	Service-linked roles
AWS Batch	Yes	Yes	No	No	Yes	No
Amazon Elastic Compute Cloud (Amazon EC2)	Yes	Yes	No	Yes	Yes	Yes ¹
Amazon EC2 Auto Scaling	Yes	Yes	No	Yes	Yes	Yes
Amazon EC2 Image Builder	Yes	Yes	No	Yes	Yes	No
AWS Elastic Beanstalk	Yes	Yes	No	Yes	Yes	Yes
Amazon Elastic Container Registry (Amazon ECR)	Yes	Yes	Yes	Yes	Yes	No
Amazon Elastic Container Service (Amazon ECS)	Yes	Yes ²	No	Yes	Yes	Yes
Amazon Elastic Kubernetes Service (Amazon EKS)	Yes	Yes	No	Yes	Yes	Yes
Amazon Elastic Inference	Yes	Yes	Yes	No	No	No
Elastic Load Balancing	Yes	Yes	No	Yes	Yes	Yes
AWS Lambda	Yes	Yes	Yes	No	Yes	Yes ³
Amazon Lightsail	Yes	Yes	No	Yes	Yes	No
AWS Outposts	Yes	Yes	No	Yes	Yes	No
AWS Serverless Application Repository	Yes	Yes	Yes	No	Yes	No

¹ Amazon EC2 service-linked roles cannot be created using the AWS Management Console, and can be used only for the following features: [Scheduled Instances](#), [Spot Instance Requests](#), [Spot Fleet Requests](#)

² Only some Amazon EC2 actions [support resource-level permissions](#).

³ AWS Lambda doesn't have service-linked roles, but Lambda@Edge does. For more information, see [Service-Linked Roles for Lambda@Edge](#) in the Amazon CloudFront Developer Guide.

Storage Services

Service	Actions	Resource-level permissions	Resource-based policies	Authorization based on tags	Temporary credentials	Service-linked roles
AWS Backup	Yes	Yes	Yes	No	Yes	No
AWS Backup Storage	Yes	Yes	No	No	Yes	No

Amazon Elastic Block Store (Amazon EBS)	Yes	Yes	No	Yes	Yes	No
Amazon Elastic File System (Amazon EFS)	Yes	Yes	No	Yes	Yes	Yes
Amazon FSx	Yes	Yes	Yes	Yes	Yes	Yes
Amazon S3 Glacier	Yes	Yes	Yes	Yes	Yes	No
AWS Import/Export	Yes	No	No	No	Yes	No
AWS Migration Hub	Yes	Yes	No	No	Yes	No
Amazon Simple Storage Service (Amazon S3)	Yes	Yes	Yes	Yes ¹	Yes	No
AWS Snowball	Yes	No	No	No	Yes	No
AWS Snowball Edge	Yes	No	No	No	No	No
AWS Storage Gateway	Yes	Yes	No	Yes	Yes	No

¹ Amazon S3 supports tag-based authorization for only object resources.

Database Services

Service	Actions	Resource-level permissions	Resource-based policies	Authorization based on tags	Temporary credentials	Service-linked roles
Amazon DynamoDB	Yes	Yes	No	No	Yes	Yes
Amazon ElastiCache	Yes	No ¹	No	No	Yes	Yes
AWS Managed Apache Cassandra Service (MCS)	Yes	Yes	No	No	Yes	No
Amazon Quantum Ledger Database (Amazon QLDB)	Yes	Yes	No	Yes	Yes	No
Amazon Redshift	Yes	Yes	No	No	Yes	Yes
Amazon Relational Database Service (Amazon RDS)	Yes	Yes	No	Yes	Yes	Yes
Amazon RDS Data API	Yes	No	No	No	Yes	No
Amazon SimpleDB	Yes	Yes	No	No	Yes	No

¹ You cannot specify ElastiCache resource ARNs in a policy, but when seeding a cluster or replication, you can specify an Amazon S3 ARN with ElastiCache actions. group.

Developer Tools Services

Service	Actions	Resource-level permissions	Resource-based policies	Authorization based on tags	Temporary credentials	Service-linked roles
AWS Cloud9	Yes	Yes	Yes	Yes	Yes	Yes
CodeBuild	Yes	Yes	Yes ¹	Yes ²	Yes	No
CodeCommit	Yes	Yes	No	Yes	Yes	No
AWS CodeDeploy	Yes	Yes	No	No	Yes	No
CodePipeline	Yes	Yes	No	Yes	Yes	No
AWS CodeStar	Yes	Yes ¹	No	Yes	Yes	No
AWS CodeStar Notifications	Yes	Yes	No	Yes	Yes	Yes
AWS X-Ray	Yes	No	No	No	Yes	No

¹ CodeBuild supports cross-account resource sharing via AWS RAM.

² CodeBuild supports authorization based on tags for project-based actions.

Security, Identity, and Compliance Services

Service	Actions	Resource-level permissions	Resource-based policies	Authorization based on tags	Temporary credentials	Service-linked roles
AWS Certificate Manager Private Certificate Authority (ACM)	Yes	Yes	No	Yes	Yes	No
AWS Artifact	Yes	Yes	No	No	Yes	No
AWS Certificate Manager (ACM)	Yes	Yes	No	Yes	Yes	No
AWS CloudHSM	Yes	Yes	No	Yes	Yes	Yes
AWS CloudHSM Classic	Yes	No	No	No	No	No
Amazon Cognito	Yes	Yes	No	Yes	Yes	Yes
Amazon Detective	Yes	Yes	No	No	Yes	No
AWS Directory Service	Yes	Yes	No	Yes	Yes	No
AWS Firewall Manager	Yes	Yes	Yes	Yes	Yes	Yes
Amazon GuardDuty	Yes	Yes	No	No	Yes	Yes
AWS Identity and Access Management (IAM)	Yes	Yes	Yes ¹	Yes ² (p. 389)	Yes ³	No
IAM Access Analyzer	Yes	Yes	No	Yes	Yes	Yes

Amazon Inspector	Yes	No	No	No	Yes	Yes
AWS Key Management Service (AWS KMS)	Yes	Yes	Yes	No	Yes	Yes
Amazon Macie	Yes	No	No	No	Yes	Yes
AWS Resource Access Manager (AWS RAM)	Yes	Yes	No	Yes	Yes	No
AWS Secrets Manager	Yes	Yes	Yes	Yes	Yes	No
AWS Security Hub	Yes	Yes	No	Yes	Yes	Yes
AWS Single Sign-On (AWS SSO)	Yes	No	No	No	Yes	Yes
AWS SSO Directory	Yes	No	No	No	Yes	No
AWS Security Token Service (AWS STS)	Yes	Yes ⁴	No	Yes	Yes ⁵	No
AWS Shield Advanced	Yes	Yes	No	No	Yes	No
AWS WAF	Yes	Yes	No	No	Yes	Yes
AWS WAFV2	Yes	Yes	No	Yes	Yes	No

¹ IAM supports only one type of resource-based policy called a role *trust policy*, which is attached to an IAM role. For more information, see [Granting a User Permissions to Switch Roles \(p. 255\)](#).

² IAM supports tag-based access control for only user and role resources.

³ Only some of the API actions for IAM can be called with temporary credentials. For more information, see [Comparing your API options](#)

⁴ AWS STS does not have "resources," but does allow restricting access in a similar way to users. For more information, see [Denying Access to Temporary Security Credentials by Name](#).

⁵ Only some of the API operations for AWS STS support calling with temporary credentials. For more information, see [Comparing your API options](#).

Machine Learning Services

Service	Actions	Resource-level permissions	Resource-based policies	Authorization based on tags	Temporary credentials	Service-linked roles
Amazon CodeGuru Profiler	Yes	Yes	No	No	Yes	No
Amazon CodeGuru Reviewer	Yes	Yes	No	Yes	Yes	Yes
Amazon Comprehend	Yes	No	No	Yes	Yes	No
AWS DeepRacer	Yes	No	No	No	Yes	Yes
Forecast	Yes	Yes	No	No	Yes	No
Amazon Fraud Detector	Yes	No	No	No	Yes	No

Amazon Kendra	Yes	Yes	No	No	Yes	No
Amazon Lex	Yes	Yes	No	No	Yes	Yes
Amazon Machine Learning	Yes	Yes	No	Yes	Yes	No
Amazon Personalize	Yes	Yes	No	No	Yes	No
Amazon Polly	Yes	Yes	No	No	Yes	No
Amazon Rekognition	Yes	Yes	No	No	Yes	No
Amazon SageMaker	Yes	Yes	No	Yes	Yes	No
Amazon Textract	Yes	Yes	No	No	No	No
Amazon Transcribe	Yes	No	No	No	Yes	No
Amazon Translate	Yes	No	No	No	Yes	No

Management and Governance Services

Service	Actions	Resource-level permissions	Resource-based policies	Authorization based on tags	Temporary credentials	Service-linked roles
Application Auto Scaling	Yes	No	No	No	Yes	Yes
AWS Auto Scaling	Yes	No	No	No	Yes	Yes
AWS CloudFormation	Yes	Yes	No	Yes	Yes	No
AWS CloudTrail	Yes	Yes	No	No	Yes	No
Amazon CloudWatch	Yes	Yes	No	Yes	Yes	Yes ¹
Amazon CloudWatch Events	Yes	Yes	No	Yes	Yes	No
Amazon CloudWatch Logs	Yes	Yes	Yes	Yes	Yes	No
Amazon CloudWatch Synthetics	Yes	Yes	No	No	Yes	No
AWS Compute Optimizer	Yes	No	No	No	Yes	Yes
AWS Config	Yes	Yes ²	No	Yes	Yes	Yes
Amazon Data Lifecycle Manager	Yes	Yes	No	Yes	Yes	No
AWS Health	Yes	Yes	No	No	Yes	No
AWS OpsWorks	Yes	Yes	No	Yes	Yes	No
AWS OpsWorks for Chef Automate	Yes	Yes	No	Yes	Yes	No
AWS Organizations	Yes	Yes	No	No	Yes	Yes
AWS Resource Groups	Yes	Yes	No	Yes	Yes ³	No
Resource Groups Tagging API	Yes	No	No	No	Yes	No

AWS Service Catalog	Yes	No	No	Yes ⁴	Yes	No
AWS Systems Manager	Yes	Yes	No	Yes	Yes	Yes
AWS Trusted Advisor	Yes ⁵	Yes	No	No	Yes	Yes
AWS Well-Architected Tool	Yes	Yes	No	No	Yes	No

¹ Amazon CloudWatch service-linked roles cannot be created using the AWS Management Console, and support only the [Alarm Actions](#) feature.

² AWS Config supports resource-level permissions for multi-account multi-Region data aggregation and AWS Config Rules. For a list of supported resources, see the [Multi-Account Multi-Region Data Aggregation](#) section and [AWS Config Rules](#) section of [AWS Config API Guide](#).

³ Users can assume a role with a policy that allows AWS Resource Groups operations.

⁴ AWS Service Catalog supports tag-based access control for only actions that match API operations with one resource in the input.

⁵ API access to Trusted Advisor is through the AWS Support API and is controlled by AWS Support IAM policies.

Migration and Transfer Services

Service	Actions	Resource-level permissions	Resource-based policies	Authorization-based on tags	Temporary credentials	Service-linked roles
AWS Application Discovery Service	Yes	No	No	No	No	Yes
AWS Database Migration Service	Yes	Yes	Yes ¹	Yes	Yes	No
AWS Migration Hub	Yes	Yes	No	No	Yes	No
AWS Server Migration Service	Yes	No	No	No	Yes	Yes

¹ You can create and modify policies that are attached to AWS KMS encryption keys you create to encrypt data migrated to supported target endpoints. The supported target endpoints include Amazon Redshift and Amazon S3. For more information, see [Creating and Using AWS KMS Keys to Encrypt Amazon Redshift Target Data](#) and [Creating AWS KMS Keys to Encrypt Amazon S3 Target Objects](#) in the [AWS Database Migration Service User Guide](#).

Mobile Services

Service	Actions	Resource-level permissions	Resource-based policies	Authorization-based on tags	Temporary credentials	Service-linked roles
AWS Amplify	Yes	Yes	No	Yes	Yes	No
AWS Device Farm	Yes	Yes	No	Yes	Yes	No

Networking and Content Delivery Services

Service	Actions	Resource-level permissions	Resource-based policies	Authorization based on tags	Temporary credentials	Service-linked roles
Amazon API Gateway	Yes	Yes	Yes	Yes	Yes	Yes
AWS App Mesh	Yes	Yes	No	Yes	Yes	Yes
Amazon CloudFront	Yes ¹	Yes	No	Yes	Yes	Yes ⁴
AWS Cloud Map	Yes	Yes	No	No	Yes	No
AWS Direct Connect	Yes	Yes	No	Yes	Yes	No
AWS Global Accelerator	Yes	Yes	No	No	Yes	Yes
Network Manager	Yes	Yes	Yes	Yes	Yes	Yes
Amazon Route 53	Yes	Yes	No	No	Yes	No
Amazon Route 53 Resolver	Yes	Yes	No	Yes	Yes	No
Amazon Virtual Private Cloud (Amazon VPC)	Yes	Yes ²	Yes ³	No	Yes	No

¹ CloudFront does not support action-level permissions for creating CloudFront key pairs. You must use an AWS account root user to create a CloudFront key pair. For more information, see [Creating CloudFront Key Pairs for Your Trusted Signers](#) in the *Amazon CloudFront Developer Guide*.

² In an IAM user policy, you cannot restrict permissions to a specific Amazon VPC endpoint. Any Action element that includes the ec2 : *VpcEndpoint* or ec2 : DescribePrefixLists API actions must specify "Resource": "*". For more information, see [Controlling the Use of Endpoints](#) in the *Amazon VPC User Guide*.

³ Amazon VPC supports attaching a single resource policy to a VPC endpoint to restrict what can be accessed through that endpoint. For more information about using resource-based policies to control access to resources from specific Amazon VPC endpoints, see [Using Endpoint Policies](#) in the *Amazon VPC User Guide*.

⁴ Amazon CloudFront doesn't have service-linked roles, but Lambda@Edge does. For more information, see [Service-Linked Roles for Lambda@Edge](#) in the *Amazon CloudFront Developer Guide*.

Media Services

Service	Actions	Resource-level permissions	Resource-based policies	Authorization based on tags	Temporary credentials	Service-linked roles
Amazon Elastic Transcoder	Yes	Yes	No	No	Yes	No
AWS Elemental MediaConnect	Yes	Yes	No	No	Yes	No
AWS Elemental MediaConvert	Yes	Yes	No	Yes	Yes	No
AWS Elemental MediaLive	Yes	Yes	Yes	Yes	Yes	No

AWS Elemental MediaPackage	Yes	Yes	No	Yes	Yes	No
AWS Elemental MediaStore	Yes	Yes	Yes	No	Yes	No
AWS Elemental MediaTailor	Yes	Yes	No	Yes	Yes	No
Kinesis Video Streams	Yes	Yes	No	Yes	Yes	No

Analytics Services

Service	Actions	Resource-level permissions	Resource-based policies	Authorization based on tags	Temporary credentials	Service-linked roles
Amazon Athena	Yes	Yes	No	Yes	Yes	No
Amazon CloudSearch	Yes	Yes	No	No	Yes	No
AWS Data Exchange	Yes	Yes	No	Yes	Yes	No
AWS Data Pipeline	Yes	No	No	Yes	Yes	No
Amazon Elasticsearch Service	Yes	Yes	Yes	No	Yes	Yes
Amazon EMR	Yes	No	No	Yes	Yes	Yes
AWS Glue	Yes	Yes	Yes	Yes	Yes	No
Amazon Kinesis Data Analytics	Yes	Yes	No	Yes	Yes	No
Amazon Kinesis Data Firehose	Yes	Yes	No	Yes	Yes	No
Amazon Kinesis Data Streams	Yes	Yes	No	No	Yes	No
AWS Lake Formation	Yes	No	No	No	Yes	Yes
Amazon Managed Streaming for Apache Kafka (MSK)	Yes	Yes	No	Yes	Yes	No
Amazon QuickSight	Yes	Yes	No	No	Yes	No

Application Integration Services

Service	Actions	Resource-level permissions	Resource-based policies	Authorization based on tags	Temporary credentials	Service-linked roles
Amazon EventBridge	Yes	Yes	No	Yes	Yes	No
Amazon EventBridge Schemas	Yes	Yes	No	Yes	Yes	No
Amazon MQ	Yes	Yes	No	Yes	Yes	No
Amazon Simple Notification Service (Amazon SNS)	Yes	Yes	Yes	No	Yes	No

Amazon Simple Queue Service (Amazon SQS)	Yes	Yes	Yes	No	Yes	No
AWS Step Functions	Yes	Yes	No	Yes	Yes	No
Amazon Simple Workflow Service (Amazon SWF)	Yes	Yes	No	Yes	Yes	No

Business Applications Services

Service	Actions	Resource-level permissions	Resource-based policies	Authorization-based on tags	Temporary credentials	Service-linked roles
Alexa for Business	Yes	Yes	No	No	Yes	No
Amazon Chime	Yes	No	No	No	Yes	Yes
Amazon WorkMail	Yes	No	No	No	Yes	Yes

Satellite Services

Service	Actions	Resource-level permissions	Resource-based policies	Authorization-based on tags	Temporary credentials	Service-linked roles
AWS Ground Station	Yes	Yes	No	Yes	Yes	No

Internet of Things Services

Service	Actions	Resource-level permissions	Resource-based policies	Authorization-based on tags	Temporary credentials	Service-linked roles
AWS IoT	Yes	Yes	Yes ¹	Yes	Yes	No
AWS IoT Analytics	Yes	Yes	No	Yes	Yes	No
AWS IoT Events	Yes	Yes	No	Yes	Yes	No
AWS IoT Greengrass	Yes	Yes	No	Yes	Yes	No
AWS IoT Things Graph	Yes	No	No	No	Yes	No
AWS IoT SiteWise	Yes	Yes	No	No	Yes	No

¹ Devices connected to AWS IoT are authenticated by using X.509 certificates or using Amazon Cognito Identities. You can attach AWS IoT policies to an X.509 certificate or Amazon Cognito Identity to control what the device is authorized to do. For more information, see [Security and Identity for AWS IoT](#) in the [AWS IoT Developer Guide](#).

Robotics Services

Service	Actions	Resource-level permissions	Resource-based policies	Authorization based on tags	Temporary credentials	Service-linked roles
RoboMaker	Yes	Yes	No	Yes	No	Yes

Blockchain Services

Service	Actions	Resource-level permissions	Resource-based policies	Authorization based on tags	Temporary credentials	Service-linked roles
Amazon Managed Blockchain	Yes	Yes	No	No	Yes	No

Game Development Services

Service	Actions	Resource-level permissions	Resource-based policies	Authorization based on tags	Temporary credentials	Service-linked roles
Amazon GameLift	Yes	Yes	No	No	Yes	No

AR & VR Services

Service	Actions	Resource-level permissions	Resource-based policies	Authorization based on tags	Temporary credentials	Service-linked roles
Amazon Sumerian	Yes	Yes	No	No	Yes	No

Customer Engagement Services

Service	Actions	Resource-level permissions	Resource-based policies	Authorization based on tags	Temporary credentials	Service-linked roles
Amazon Connect	Yes	Yes	No	No	Yes	Yes
Amazon Pinpoint	Yes	Yes	No	Yes	Yes	No
Amazon Simple Email Service (Amazon SES)	Yes	Yes ¹	Yes	Yes	Yes ²	No

¹ You can only use resource-level permissions in policy statements that refer to actions related to sending email, such as `ses:SendEmail` or `ses:SendRawEmail`. For policy statements that refer to any other actions, the Resource element can only contain `*`.

² Only the Amazon SES API supports temporary security credentials. The Amazon SES SMTP interface does not support SMTP credentials that are derived from temporary security credentials.

End User Computing Services

Service	Actions	Resource-level permissions	Resource-based policies	Authorization based on tags	Temporary based credentials	Service-linked roles
Amazon AppStream	Yes	No	No	No	Yes	No
Amazon AppStream 2.0	Yes	Yes	No	Yes	Yes	No
Amazon WAM	Yes	No	No	No	Yes	No
Amazon WorkDocs	Yes	No	No	No	Yes	No
Amazon WorkLink	Yes	Yes	Yes	No	Yes	Yes
Amazon WorkSpaces	Yes	Yes	No	Yes	Yes	No

Additional Resources

Service	Actions	Resource-level permissions	Resource-based policies	Authorization based on tags	Temporary based credentials	Service-linked roles
AWS Billing and Cost Management	Yes	No	No	No	Yes	No
AWS Marketplace	Yes	No	No	No	Yes	No
AWS Private Marketplace	Yes	No	No	No	No	No
AWS Support	Yes	No	No	No	Yes	Yes

IAM JSON Policy Reference

This section presents detailed syntax, descriptions, and examples of the elements, variables, and evaluation logic of JSON policies in IAM. For more general information, see [Overview of JSON Policies \(p. 361\)](#).

This reference includes the following sections.

- [IAM JSON Policy Elements Reference \(p. 598\)](#) — Learn more about the elements that you can use when you create a policy. View additional policy examples and learn about conditions, supported data types, and how they are used in various services.
- [Policy Evaluation Logic \(p. 636\)](#) — This section describes AWS requests, how they are authenticated, and how AWS uses policies to determine access to resources.
- [Grammar of the IAM JSON Policy Language \(p. 651\)](#) — This section presents a formal grammar for the language that is used to create policies in IAM.

- [AWS Managed Policies for Job Functions \(p. 656\)](#) — This section lists all the AWS managed policies that directly map to common job functions in the IT industry. Use these policies to grant the permissions that are needed to carry out the tasks expected of someone in a specific job function. These policies consolidate permissions for many services into a single policy.
- [AWS Global Condition Context Keys \(p. 665\)](#) — This section includes a list of all the AWS global condition keys that you can use to limit permissions in an IAM policy.
- [IAM and AWS STS Condition Context Keys \(p. 676\)](#) — This section includes a list of all the IAM and AWS STS condition keys that you can use to limit permissions in an IAM policy.
- [Actions, Resources, and Condition Keys for AWS Services \(p. 686\)](#) — This section presents a list of all the AWS API operations that you can use as permissions in an IAM policy. It also includes the service-specific condition keys that can be used to further refine the request.

IAM JSON Policy Elements Reference

JSON policy documents are made up of elements. The elements are listed here in the general order you use them in a policy. The order of the elements doesn't matter—for example, the `Resource` element can come before the `Action` element. You're not required to specify any `Condition` elements in the policy. To learn more about the general structure and purpose of a JSON policy document, see [Overview of JSON Policies \(p. 361\)](#).

Some JSON policy elements are mutually exclusive. This means that you cannot create a policy that uses both. For example, you cannot use both `Action` and `NotAction` in the same policy statement. Other pairs that are mutually exclusive include `Principal/NotPrincipal` and `Resource/NotResource`.

The details of what goes into a policy vary for each service, depending on what actions the service makes available, what types of resources it contains, and so on. When you're writing policies for a specific service, it's helpful to see examples of policies for that service. For a list of all the services that support IAM, and for links to the documentation in those services that discusses IAM and policies, see [AWS Services That Work with IAM \(p. 586\)](#).

Topics

- [IAM JSON Policy Elements: Version \(p. 598\)](#)
- [IAM JSON Policy Elements: Id \(p. 599\)](#)
- [IAM JSON Policy Elements: Statement \(p. 599\)](#)
- [IAM JSON Policy Elements: Sid \(p. 600\)](#)
- [IAM JSON Policy Elements: Effect \(p. 600\)](#)
- [AWS JSON Policy Elements: Principal \(p. 601\)](#)
- [AWS JSON Policy Elements: NotPrincipal \(p. 604\)](#)
- [IAM JSON Policy Elements: Action \(p. 606\)](#)
- [IAM JSON Policy Elements: NotAction \(p. 607\)](#)
- [IAM JSON Policy Elements: Resource \(p. 609\)](#)
- [IAM JSON Policy Elements: NotResource \(p. 610\)](#)
- [IAM JSON Policy Elements: Condition \(p. 611\)](#)
- [IAM Policy Elements: Variables and Tags \(p. 628\)](#)
- [IAM JSON Policy Elements: Supported Data Types \(p. 635\)](#)

IAM JSON Policy Elements: Version

Disambiguation Note

This `Version` JSON policy element is different from a *policy version*. The `Version` policy element is used within a policy and defines the version of the policy language. A policy version,

on the other hand, is created when you make changes to a customer managed policy in IAM. The changed policy doesn't overwrite the existing policy. Instead, IAM creates a new version of the managed policy. If you were searching for information about the multiple version support available for managed policies, see [the section called "Versioning IAM Policies" \(p. 465\)](#).

The `Version` policy element specifies the language syntax rules that are to be used to process a policy. To use all of the available policy features, include the following `Version` element before the `Statement` element in all of your policies.

```
"Version": "2012-10-17"
```

IAM supports the following `Version` element values:

- 2012-10-17. This is the current version of the policy language, and you should always include a `Version` element and set it to 2012-10-17. Otherwise, you cannot use features such as [policy variables \(p. 628\)](#) that were introduced with this version.
- 2008-10-17. This was an earlier version of the policy language. You might see this version on older existing policies. Do not use this version for any new policies or when you update any existing policies.

If you do not include a `Version` element, the value defaults to 2008-10-17, but newer features, such as policy variables, will not work with your policy. For example, variables such as `#{aws:username}` aren't recognized as variables and are instead treated as literal strings in the policy.

IAM JSON Policy Elements: Id

The `Id` element specifies an optional identifier for the policy. The ID is used differently in different services.

For services that let you set an `Id` element, we recommend you use a UUID (GUID) for the value, or incorporate a UUID as part of the ID to ensure uniqueness.

```
"Id": "cd3ad3d9-2776-4ef1-a904-4c229d1642ee"
```

Note

Some AWS services (for example, Amazon SQS or Amazon SNS) might require this element and have uniqueness requirements for it. For service-specific information about writing policies, refer to the documentation for the service you're working with.

IAM JSON Policy Elements: Statement

The `Statement` element is the main element for a policy. This element is required. The `Statement` element can contain a single statement or an array of individual statements. Each individual statement block must be enclosed in curly braces `{ }`. For multiple statements, the array must be enclosed in square brackets `[]`.

```
"Statement": [{...},{...},{...}]
```

The following example shows a policy that contains an array of three statements inside a single `Statement` element. (The policy allows you to access your own "home folder" in the Amazon S3 console.) The policy includes the `aws:username` variable, which is replaced during policy evaluation with the user name from the request. For more information, see [Introduction \(p. 628\)](#).

```
{
```

```

    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3>ListAllMyBuckets",
                "s3>GetBucketLocation"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Effect": "Allow",
            "Action": "s3>ListBucket",
            "Resource": "arn:aws:s3:::BUCKET-NAME",
            "Condition": {"StringLike": {"s3:prefix": [
                "",
                "home/",
                "home/${aws:username}/"
            ]}}
        },
        {
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::BUCKET-NAME/home/${aws:username}",
                "arn:aws:s3:::BUCKET-NAME/home/${aws:username}/*"
            ]
        }
    ]
}

```

IAM JSON Policy Elements: Sid

The **Sid** (statement ID) is an optional identifier that you provide for the policy statement. You can assign a **Sid** value to each statement in a statement array. In services that let you specify an **ID** element, such as SQS and SNS, the **Sid** value is just a sub-ID of the policy document's ID. In IAM, the **Sid** value must be unique within a JSON policy.

```
"Sid": "1"
```

The **Sid** element supports uppercase letters, lowercase letters, and numbers.

In IAM, the **Sid** is not exposed in the IAM API. You can't retrieve a particular statement based on this ID.

Note

Some AWS services (for example, Amazon SQS or Amazon SNS) might require this element and have uniqueness requirements for it. For service-specific information about writing policies, refer to the documentation for the service you're working with.

IAM JSON Policy Elements: Effect

The **Effect** element is required and specifies whether the statement results in an allow or an explicit deny. Valid values for **Effect** are **Allow** and **Deny**.

```
"Effect": "Allow"
```

By default, access to resources is denied. To allow access to a resource, you must set the **Effect** element to **Allow**. To override an allow (for example, to override an allow that is otherwise in force), you set the **Effect** element to **Deny**. For more information, see [Policy Evaluation Logic \(p. 636\)](#).

AWS JSON Policy Elements: Principal

Use the `Principal` element in a policy to specify the principal that is allowed or denied access to a resource. You cannot use the `Principal` element in an IAM identity-based policy. You can use it in the trust policies for IAM roles and in resource-based policies. Resource-based policies are policies that you embed directly in an IAM resource. For example, you can embed policies in an Amazon S3 bucket or an AWS KMS customer master key (CMK).

You can specify any of the following principals in a policy:

- AWS account and root user
- IAM users
- Federated users (using web identity or SAML federation)
- IAM roles
- Assumed-role sessions
- AWS services
- Anonymous users (not recommended)

Use the `Principal` element in these ways:

- In IAM roles, use the `Principal` element in the role's trust policy to specify who can assume the role. For cross-account access, you must specify the 12-digit identifier of the trusted account. To learn whether principals in accounts outside of your zone of trust (trusted organization, OU, or account) have access to assume your roles, see [What is IAM Access Analyzer?](#).

Note

After you create the role, you can change the account to "*" to allow everyone to assume the role. If you do this, we strongly recommend that you limit who can access the role through other means, such as a `Condition` element that limits access to only certain IP addresses. Do not leave your role accessible to everyone!

- In resource-based policies, use the `Principal` element to specify the accounts or users who are allowed to access the resource.

Do not use the `Principal` element in policies that you attach to IAM users and groups. Similarly, you do not specify a principal in the permission policy for an IAM role. In those cases, the principal is implicitly the user that the policy is attached to (for IAM users) or the user who assumes the role (for role access policies). When the policy is attached to an IAM group, the principal is the IAM user in that group who is making the request.

Specifying a Principal

You specify a principal using the [Amazon Resource Name \(ARN\)](#) (p. 576) or other identifier of the principal. You cannot specify IAM groups and instance profiles as principals.

The following examples show different methods for specifying principals.

Specific AWS Accounts

When you use an AWS account identifier as the principal in a policy, you delegate authority to the account. Within that account, the permissions in the policy statement can be granted to all identities. This includes IAM users and roles in that account. When you specify an AWS account, you can use the account ARN (`arn:aws:iam::AWS-account-ID:root`), or a shortened form that consists of the `AWS:` prefix followed by the account ID.

For example, given an account ID of 123456789012, you can use either of the following methods to specify that account in the `Principal` element:

```
"Principal": { "AWS": "arn:aws:iam::123456789012:root" }
```

```
"Principal": { "AWS": "123456789012" }
```

You can also specify more than one AWS account as a principal using an array, with any combination of the methods that we previously mentioned.

```
"Principal": {  
    "AWS": [  
        "arn:aws:iam::123456789012:root",  
        "999999999999"  
    ]  
}
```

Some AWS services support additional options for specifying an account principal. For example, Amazon S3 lets you specify a [canonical user ID](#) using the following format:

```
"Principal": { "CanonicalUser":  
    "79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be" }
```

Individual IAM Users

You can specify an individual IAM user (or array of users) as the principal, as in the following examples.

Note

In a Principal element, the user name is case sensitive.

```
"Principal": { "AWS": "arn:aws:iam::AWS-account-ID:user/user-name" }
```

```
"Principal": {  
    "AWS": [  
        "arn:aws:iam::AWS-account-ID:user/user-name-1",  
        "arn:aws:iam::AWS-account-ID:user/UserName2"  
    ]  
}
```

When you specify users in a Principal element, you cannot use a wildcard (*) to mean "all users". Principals must always name a specific user or users.

Important

If your Principal element in a role trust policy contains an ARN that points to a specific IAM user, then that ARN is transformed to the user's unique principal ID when the policy is saved. This helps mitigate the risk of someone escalating their privileges by removing and recreating the user. You don't normally see this ID in the console, because there is also a reverse transformation back to the user's ARN when the trust policy is displayed. However, if you delete the user, then the relationship is broken. The policy no longer applies, even if you recreate the user. That's because the new user has a new principal ID that does not match the ID stored in the trust policy. When this happens, the principal ID shows up in the console because AWS can no longer map it back to a valid ARN. The result is that if you delete and recreate a user referenced in a trust policy's Principal element, you must edit the role to replace the now incorrect principal ID with the correct ARN. The ARN is once again transformed into the user's new principal ID when you save the policy.

Federated Web Identity Users

```
"Principal": { "Federated": "cognito-identity.amazonaws.com" }
```

```
"Principal": { "Federated": "www.amazon.com" }
```

```
"Principal": { "Federated": "graph.facebook.com" }
```

```
"Principal": { "Federated": "accounts.google.com" }
```

Federated SAML Users

```
"Principal": { "Federated": "arn:aws:iam::AWS-account-ID:saml-provider/provider-name" }
```

IAM Roles

```
"Principal": { "AWS": "arn:aws:iam::AWS-account-ID:role/role-name" }
```

Important

If your `Principal` element in a role trust policy contains an ARN that points to a specific IAM role, then that ARN is transformed to the role's unique principal ID when the policy is saved. This helps mitigate the risk of someone escalating their privileges by removing and recreating the role. You don't normally see this ID in the console, because there is also a reverse transformation back to the role's ARN when the trust policy is displayed. However, if you delete the role, then the relationship is broken. The policy no longer applies, even if you recreate the role because the new role has a new principal ID that does not match the ID stored in the trust policy. When this happens, the principal ID shows up in the console because AWS can no longer map it back to a valid ARN. The end result is that if you delete and recreate a role referenced in a trust policy's `Principal` element, you must edit the role to replace the now incorrect principal ID with the correct ARN. The ARN will once again be transformed into the role's new principal ID when you save the policy.

Specific Assumed-Role Sessions

```
"Principal": { "AWS": "arn:aws:sts::AWS-account-ID:assumed-role/role-name/role-session-name" }
```

When you specify an assumed-role session in a `Principal` element, you cannot use a wildcard (*) to mean "all sessions". Principals must always name a specific session.

AWS Services

IAM roles that can be assumed by an AWS service are called [service roles \(p. 176\)](#). Service roles must include a trust policy. *Trust policies* are resource-based policies that are attached to a role that define which principals can assume the role. Some service roles have predefined trust policies. However, in some cases, you must specify the service principal in the trust policy. A *service principal* is an identifier that is used to grant permissions to a service. The identifier includes the long version of a service name, and is usually in the following format:

`long_service-name.amazonaws.com`

The service principal is defined by the service. To learn the service principal for a service, see the documentation for that service. For some services, see [AWS Services That Work with IAM \(p. 586\)](#) and look for the services that have Yes in the **Service-Linked Role** column. Choose a Yes with a link to view the service-linked role documentation for that service. View the **Service-Linked Role Permissions** section for that service to view the service principal.

The following example shows a policy that can be attached to a service role. The policy enables two services, Amazon EMR and AWS Data Pipeline, to assume the role. The services can then perform any tasks granted by the permissions policy assigned to the role (not shown). To specify multiple service principals, you do not specify two `Service` elements; you can have only one. Instead, you use an array of multiple service principals as the value of a single `Service` element.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Principal": {  
                "Service": [  
                    "elasticmapreduce.amazonaws.com",  
                    "datapipeline.amazonaws.com"  
                ]  
            },  
            "Action": "sts:AssumeRole"  
        }  
    ]  
}
```

Anonymous Users (Public)

The following are equivalent:

```
"Principal": "*"
```

```
"Principal" : { "AWS" : "*" }
```

Important

In these examples, the asterisk (*) is used as a placeholder for Everyone/Anonymous. You cannot use it as a wildcard to match part of a name or an ARN. We also strongly recommend that you do not use a wildcard in the `Principal` element in a role's trust policy unless you otherwise restrict access through a `Condition` element in the policy. Otherwise, any IAM user in any account can access the role.

More Information

For more information, see the following:

- [Bucket Policy Examples](#) in the *Amazon Simple Storage Service Developer Guide*
- [Example Policies for Amazon SNS](#) in the *Amazon Simple Notification Service Developer Guide*
- [Amazon SQS Policy Examples](#) in the *Amazon Simple Queue Service Developer Guide*
- [Key Policies](#) in the *AWS Key Management Service Developer Guide*
- [Account Identifiers](#) in the *AWS General Reference*
- [About Web Identity Federation \(p. 185\)](#)

AWS JSON Policy Elements: NotPrincipal

Use the `NotPrincipal` element to specify the IAM user, federated user, IAM role, AWS account, AWS service, or other principal that is *not* allowed or denied access to a resource. The `NotPrincipal` element enables you to specify an exception to a list of principals. Use this element to deny access to all principals *except* the one named in the `NotPrincipal` element. The syntax for specifying `NotPrincipal` is the same as for specifying [AWS JSON Policy Elements: Principal \(p. 601\)](#).

You cannot use the `NotPrincipal` element in an IAM identity-based policy. You can use it in the trust policies for IAM roles and in resource-based policies. Resource-based policies are policies that you embed directly in an IAM resource.

Important

Very few scenarios require the use of `NotPrincipal`, and we recommend that you explore other authorization options before you decide to use `NotPrincipal`.

NotPrincipal with Allow

We strongly recommend that you do not use `NotPrincipal` in the same policy statement as `"Effect": "Allow"`. Doing so allows all principals *except* the one named in the `NotPrincipal` element. We do not recommend this because the permissions specified in the policy statement will be granted to *all* principals except for the ones specified. By doing this, you might grant access to anonymous (unauthenticated) users.

NotPrincipal with Deny

When you use `NotPrincipal` in the same policy statement as `"Effect": "Deny"`, the actions specified in the policy statement are explicitly denied to all principals *except* for the ones specified. You can use this method to implement a form of whitelisting. When you use `NotPrincipal` with `Deny`, you must also specify the account ARN of the not-denied principal. Otherwise, the policy might deny access to the entire account containing the principal. Depending on the service that you include in your policy, AWS might validate the account first and then the user. If an assumed-role user (someone who is using a role) is being evaluated, AWS might validate the account first, then the role, and then the assumed-role user. The assumed-role user is identified by the role session name that is specified when they assumed the role. Therefore, we strongly recommend that you explicitly include the ARN for a user's account, or include both the ARN for a role and the ARN for the account containing that role.

Note

As a best practice, you should include the ARNs for the account in your policy. Some services require the account ARN, although this is not required in all cases. Any existing policies without the required ARN will continue to work, but new policies that include these services must meet this requirement. IAM does not track these services, and therefore recommends that you always include the account ARN.

The following examples show how to use `NotPrincipal` and `"Effect": "Deny"` in the same policy statement effectively.

Example 1: An IAM user in the same or a different account

In the following example, all principals *except* the user named Bob in AWS account 444455556666 are explicitly denied access to a resource. Note that as a best practice, the `NotPrincipal` element contains the ARN of both the user Bob and the AWS account that Bob belongs to (`arn:aws:iam::444455556666:root`). If the `NotPrincipal` element contained only Bob's ARN, the effect of the policy might be to explicitly deny access to the AWS account that contains the user Bob. In some cases, a user cannot have more permissions than its parent account, so if Bob's account is explicitly denied access then Bob might be unable to access the resource.

This example works as intended when it is part of a policy statement in a resource-based policy that is attached to a resource in either the same or a different AWS account (not 444455556666). This example by itself does not grant access to Bob, it only omits Bob from the list of principals that are explicitly denied. To allow Bob access to the resource, another policy statement must explicitly allow access using `"Effect": "Allow"`.

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

```

    "Effect": "Deny",
    "NotPrincipal": {"AWS": [
        "arn:aws:iam::444455556666:user/Bob",
        "arn:aws:iam::444455556666:root"
    ]},
    "Action": "s3:*",
    "Resource": [
        "arn:aws:s3::::BUCKETNAME",
        "arn:aws:s3::::BUCKETNAME/*"
    ]
}
]
}

```

Example 2: An IAM role in the same or different account

In the following example, all principals *except* the assumed-role user named cross-account-audit-app in AWS account 444455556666 are explicitly denied access to a resource. As a best practice, the NotPrincipal element contains the ARN of the assumed-role user (cross-account-audit-app), the role (cross-account-read-only-role), and the AWS account that the role belongs to (444455556666). If the NotPrincipal element was missing the ARN of the role, the effect of the policy might be to explicitly deny access to the role. Similarly, if the NotPrincipal element was missing the ARN of the AWS account that the role belongs to, the effect of the policy might be to explicitly deny access to the AWS account and all entities in that account. In some cases, assumed-role users cannot have more permissions than their parent role, and roles cannot have more permissions than their parent AWS account, so when the role or the account is explicitly denied access, the assumed role user might be unable to access the resource.

This example works as intended when it is part of a policy statement in a resource-based policy that is attached to a resource in a different AWS account (not 444455556666). This example by itself does not allow access to the assumed-role user cross-account-audit-app, it only omits cross-account-audit-app from the list of principals that are explicitly denied. To give cross-account-audit-app access to the resource, another policy statement must explicitly allow access using "Effect": "Allow".

```

{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "NotPrincipal": {"AWS": [
                "arn:aws:sts::444455556666:assumed-role/cross-account-read-only-role/cross-
account-audit-app",
                "arn:aws:iam::444455556666:role/cross-account-read-only-role",
                "arn:aws:iam::444455556666:root"
            ]},
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3::::Bucket_AccountAudit",
                "arn:aws:s3::::Bucket_AccountAudit/*"
            ]
        }
    ]
}

```

When you specify an assumed-role session in a NotPrincipal element, you cannot use a wildcard (*) to mean "all sessions". Principals must always name a specific session.

IAM JSON Policy Elements: Action

The Action element describes the specific action or actions that will be allowed or denied. Statements must include either an Action or NotAction element. Each AWS service has its own set of actions that describe tasks that you can perform with that service. For example, the list of actions for Amazon S3 can

be found at [Specifying Permissions in a Policy](#) in the *Amazon Simple Storage Service Developer Guide*, the list of actions for Amazon EC2 can be found in the [Amazon EC2 API Reference](#), and the list of actions for AWS Identity and Access Management can be found in the [IAM API Reference](#). To find the list of actions for other services, consult the API reference [documentation](#) for the service.

You specify a value using a service namespace as an action prefix (`iam`, `ec2`, `sqs`, `sns`, `s3`, etc.) followed by the name of the action to allow or deny. The name must match an action that is supported by the service. The prefix and the action name are case insensitive. For example, `iam>ListAccessKeys` is the same as `IAM:listaccesskeys`. The following examples show Action elements for different services.

Amazon SQS action

```
"Action": "sqs:SendMessage"
```

Amazon EC2 action

```
"Action": "ec2:StartInstances"
```

IAM action

```
"Action": "iam:ChangePassword"
```

Amazon S3 action

```
"Action": "s3:GetObject"
```

You can specify multiple values for the Action element.

```
"Action": [ "sqs:SendMessage", "sqs:ReceiveMessage", "ec2:StartInstances",
  "iam:ChangePassword", "s3:GetObject" ]
```

You can use a wildcard (*) to give access to all the actions the specific AWS product offers. For example, the following Action element applies to all S3 actions.

```
"Action": "s3:*
```

You can also use wildcards (*) as part of the action name. For example, the following Action element applies to all IAM actions that include the string `AccessKey`, including `CreateAccessKey`, `DeleteAccessKey`, `ListAccessKeys`, and `UpdateAccessKey`.

```
"Action": "iam:*AccessKey*"
```

Some services let you limit the actions that are available. For example, Amazon SQS lets you make available just a subset of all the possible Amazon SQS actions. In that case, the * wildcard doesn't allow complete control of the queue; it allows only the subset of actions that you've shared. For more information, see [Understanding Permissions](#) in the *Amazon Simple Queue Service Developer Guide*.

IAM JSON Policy Elements: NotAction

NotAction is an advanced policy element that explicitly matches everything *except* the specified list of actions. Using NotAction can result in a shorter policy by listing only a few actions that should not

match, rather than including a long list of actions that will match. When using `NotAction`, you should keep in mind that actions specified in this element are the *only* actions in that are limited. This, in turn, means that all of the applicable actions or services that are not listed are allowed if you use the `Allow` effect. In addition, such unlisted actions or services are denied if you use the `Deny` effect. When you use `NotAction` with the `Resource` element, you provide scope for the policy. This is how AWS determines which actions or services are applicable. For more information, see the following example policy.

NotAction with Allow

You can use the `NotAction` element in a statement with `"Effect": "Allow"` to provide access to all of the actions in an AWS service, except for the actions specified in `NotAction`. You can use it with the `Resource` element to provide scope for the policy, limiting the allowed actions to the actions that can be performed on the specified resource.

The following example allows users to access all of the Amazon S3 actions that can be performed on any S3 resource *except* for deleting a bucket. This does not allow users to use the `ListAllMyBuckets` S3 API operation, because that action requires the `"*"` resource. This policy also does not allow actions in other services, because other service actions are not applicable to the S3 resources.

```
"Effect": "Allow",
"NotAction": "s3:DeleteBucket",
"Resource": "arn:aws:s3:::/*",
```

Sometimes, you might want to allow access to a large number of actions. By using the `NotAction` element you effectively reverse the statement, resulting in a shorter list of actions. For example, because AWS has so many services, you might want to create a policy that allows the user to do everything *except* access IAM actions.

The following example allows users to access every action in every AWS service *except* for IAM.

```
"Effect": "Allow",
"NotAction": "iam:*",
"Resource": "*"
```

Be careful using the `NotAction` element and `"Effect": "Allow"` in the same statement or in a different statement within a policy. `NotAction` matches all services and actions that are not explicitly listed or applicable to the specified resource, and could result in granting users more permissions than you intended.

NotAction with Deny

You can use the `NotAction` element in a statement with `"Effect": "Deny"` to deny access to all of the listed resources *except* for the actions specified in the `NotAction` element. This combination does not allow the listed items, but instead explicitly denies the actions not listed. You must still allow actions that you want to allow.

The following conditional example denies access to non-IAM actions if the user is not signed in using MFA. If the user is signed in with MFA, then the `"Condition"` test fails and the final `"Deny"` statement has no effect. Note, however, that this would not grant the user access to any actions; it would only explicitly deny all other actions *except* IAM actions.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DenyAllUsersNotUsingMFA",
            "Effect": "Deny",
            "NotAction": "iam:*",
```

```

        "Resource": "*",
        "Condition": {"BoolIfExists": {"aws:MultiFactorAuthPresent": "false"}}
    }]
}

```

The following example policy denies access to any operations outside of the `eu-central-1` and `eu-west-1` regions, except for actions in the listed services. The services listed in the `NotActions` element are some of the AWS global services with a single endpoint physically located in the `us-east-1` Region. Operations in these services would fail otherwise. This policy denies access and requires another policy to grant access. To view the example policy, see [AWS: Denies Access to AWS Based on the Requested Region \(p. 408\)](#).

IAM JSON Policy Elements: Resource

The `Resource` element specifies the object or objects that the statement covers. Statements must include either a `Resource` or a `NotResource` element. You specify a resource using an ARN. (For more information about the format of ARNs, see [Amazon Resource Names \(ARNs\) and AWS Service Namespaces](#).)

Each service has its own set of resources. Although you always use an ARN to specify a resource, the details of the ARN for a resource depend on the service and the resource. For information about how to specify a resource, refer to the documentation for the service whose resources you're writing a statement for.

Note

Some services do not let you specify actions for individual resources; instead, any actions that you list in the `Action` or `NotAction` element apply to all resources in that service. In these cases, you use the wildcard `*` in the `Resource` element.

The following example refers to a specific Amazon SQS queue.

```
"Resource": "arn:aws:sqs:us-east-2:account-ID-without-hyphens:queue1"
```

The following example refers to the IAM user named Bob in an AWS account.

```
"Resource": "arn:aws:iam::account-ID-without-hyphens:user/Bob"
```

You can use wildcards as part of the resource ARN. You can use wildcard characters (`*` and `?`) within any ARN segment (the parts separated by colons). An asterisk (`*`) represents any combination of characters and a question mark (`?`) represents any single character. You can use multiple `*` or `?` characters in each segment, but a wildcard cannot span segments. The following example refers to all IAM users whose path is `/accounting`.

```
"Resource": "arn:aws:iam::account-ID-without-hyphens:user/accounting/*"
```

The following example refers to all items within a specific Amazon S3 bucket.

```
"Resource": "arn:aws:s3:::my_corporate_bucket/*"
```

You can specify multiple resources. The following example refers to two DynamoDB tables.

```

"Resource": [
    "arn:aws:dynamodb:us-east-2:account-ID-without-hyphens:table/books_table",
    "arn:aws:dynamodb:us-east-2:account-ID-without-hyphens:table/magazines_table"
]

```

]

In the `Resource` element, you can use JSON [policy variables \(p. 628\)](#) in the part of the ARN that identifies the specific resource (that is, in the trailing part of the ARN). For example, you can use the key `{aws:username}` as part of a resource ARN to indicate that the current user's name should be included as part of the resource's name. The following example shows how you can use the `{aws:username}` key in a `Resource` element. The policy allows access to a Amazon DynamoDB table that matches the current user's name.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "dynamodb:*",
      "Resource": "arn:aws:dynamodb:us-east-2:ACCOUNT-ID-WITHOUT-HYPHENS:table/$aws:username"
    }
  ]
}
```

For more information about JSON policy variables, see [IAM Policy Elements: Variables and Tags \(p. 628\)](#).

IAM JSON Policy Elements: NotResource

`NotResource` is an advanced policy element that explicitly matches every resource except those specified. Using `NotResource` can result in a shorter policy by listing only a few resources that should not match, rather than including a long list of resources that will match. This is particularly useful for policies that apply within a single AWS service.

For example, imagine you have a group named `HRPayroll`. Members of `HRPayroll` should not be allowed to access any Amazon S3 resources except the `Payroll` folder in the `HRBucket` bucket. The following policy explicitly denies access to all Amazon S3 resources other than the listed resources. Note, however, that this policy does not grant the user access to any resources.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "s3:*",
      "NotResource": [
        "arn:aws:s3:::HRBucket/Payroll",
        "arn:aws:s3:::HRBucket/Payroll/*"
      ]
    }
  ]
}
```

Normally, to explicitly deny access to a resource you would write a policy that uses `"Effect": "Deny"` and that includes a `Resource` element that lists each folder individually. However, in that case, each time you add a folder to `HRBucket`, or add a resource to Amazon S3 that should not be accessed, you must add its name to the list in `Resource`. If you use a `NotResource` element instead, users are automatically denied access to new folders unless you add the folder names to the `NotResource` element.

When using `NotResource`, you should keep in mind that resources specified in this element are the *only* resources that are not limited. This, in turn, limits all of the resources that would apply to the action. In the example above, the policy affects only Amazon S3 actions, and therefore only Amazon S3 resources. If the action also included Amazon EC2 actions, then the policy would not deny access to any EC2 resources. Additionally, this policy does not deny access to S3 actions that can't be performed on a

specific resource, such as `s3>ListAllMyBuckets`. To learn which actions in a service allow specifying the ARN of a resource, see [Actions, Resources, and Condition Keys for AWS Services \(p. 686\)](#).

NotResource with Other Elements

You should **never** use the `"Effect": "Allow"`, `"Action": "*"`, and `"NotResource": "arn:aws:s3:::HRBucket"` elements together. This statement is very dangerous, because it allows all actions in AWS on all resources except the HRBucket S3 bucket. This would even allow the user to add a policy to themselves that allows them to access HRBucket. Do not do this.

Be careful using the `NotResource` element and `"Effect": "Allow"` in the same statement or in a different statement within a policy. `NotResource` allows all services and resources that are not explicitly listed, and could result in granting users more permissions than you intended. Using the `NotResource` element and `"Effect": "Deny"` in the same statement denies services and resources that are not explicitly listed.

IAM JSON Policy Elements: Condition

The `Condition` element (or *Condition block*) lets you specify conditions for when a policy is in effect. The `Condition` element is optional. In the `Condition` element, you build expressions in which you use [condition operators \(p. 614\)](#) (equal, less than, etc.) to match the condition keys and values in the policy against keys and values in the request context. To learn more about the request context, see [Request \(p. 5\)](#).

```
"Condition" : { "{condition-operator}" : { "{condition-key}" : "{condition-value}" }}
```

The condition key that you specify can be a [global condition key \(p. 665\)](#) or a service-specific condition key. Global condition keys have the `aws:` prefix. Service-specific condition keys have the service's prefix. For example, Amazon EC2 lets you write a condition using the `ec2:InstanceType` key, which is unique to that service. To view service-specific IAM condition keys with the `iam:` prefix, see [IAM and AWS STS Condition Context Keys \(p. 676\)](#).

Condition key *names* are not case-sensitive. For example, including the `aws:SourceIP` condition key is equivalent to testing for `AWS:SourceIp`. Case-sensitivity of condition key *values* depends on the [condition operator \(p. 614\)](#) that you use. For example, the following condition includes the `StringEquals` operator to ensure that only requests made by `johndoe` will match. Users named `JohnDoe` are denied access.

```
"Condition" : { "StringEquals" : { "aws:username" : "johndoe" }}
```

The following condition uses the `StringEqualsIgnoreCase (p. 614)` operator to match users named `johndoe` or `JohnDoe`.

```
"Condition" : { "StringEqualsIgnoreCase" : { "aws:username" : "johndoe" }}
```

Some condition keys support key-value pairs that allow you to specify part of the key name. Examples include the `aws:RequestTag/tag-key (p. 665)` global condition key, the AWS KMS `kms:EncryptionContext:encryption_context_key`, and the `ResourceTag/tag-key` condition key supported by multiple services. If you use the `ResourceTag/tag-key` condition key for a service such as [Amazon EC2](#), then you must specify a key name for the tag-key. **Key names are not case-sensitive**. This means that if you specify `"ec2:ResourceTag:TagKey1": "Value1"` in the condition element of your policy, then the condition matches a resource tag key named either `TagKey1` or `tagkey1`, but not both. AWS services that support these attributes might allow you to create multiple key names that differ only by case. An example is tagging an Amazon EC2 instance with `foo=bar1` and `Foo=bar2`. When you use a condition such as `"ec2:ResourceTag:Foo": "bar1"` to allow

access to that resource, the key name matches both tags, but only one value matches. This can result in unexpected condition failures.

Important

As a best practice, make sure that members of your account follow a consistent naming convention when naming key-value pair attributes. Examples include tags or AWS KMS encryption contexts. You can enforce this using the [aws:TagKeys \(p. 674\)](#) condition key for tagging, or the [kms:EncryptionContextKeys](#) for the AWS KMS encryption context.

- For a list of all of the condition operators and a description of how each one works, see [Condition Operators \(p. 614\)](#)
- Unless otherwise specified, all keys can have multiple values. For a description of how to handle condition keys that have multiple values, see [Creating a Condition with Multiple Keys or Values \(p. 621\)](#)
- For a list of all of the globally available condition keys, see [AWS Global Condition Context Keys \(p. 665\)](#).
- For conditions keys that are defined by each service, see [Actions, Resources, and Condition Keys for AWS Services \(p. 686\)](#).

The Request Context

When a [principal \(p. 5\)](#) makes a [request \(p. 5\)](#) to AWS, AWS gathers the request information into a request context. The information is used to evaluate and authorize the request. You can use the `Condition` element of a JSON policy to test specific conditions against the request context. For example, you can create a policy that uses the [aws:CurrentTime \(p. 665\)](#) condition key to [allow a user to perform specific actions only during business hours \(p. 397\)](#).

When a request is submitted, AWS evaluates each condition key in the policy returns a value of *true*, *false*, *not present*, and occasionally *null* (an empty data string). A key that is not present in the request is not considered a mismatch. For example, the following policy allows removing your own multifactor authentication (MFA) device, but only if you have signed in using MFA in the last hour (3,600 seconds).

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {"Sid": "AllowRemoveMfaOnlyIfRecentMfa",  
         "Effect": "Allow",  
         "Action": [  
             "iam:DeactivateMFADevice",  
             "iam:DeleteVirtualMFADevice"  
         ],  
         "Resource": "arn:aws:iam::*:user/${aws:username}",  
         "Condition": {  
             "NumericLessThanEquals": {"aws:MultiFactorAuthAge": "3600"}  
         }  
    }  
}
```

The request context can return the following values:

- **True** – If the requester signed in using MFA in the last one hour or less, then the condition returns *true*.
- **False** – If the requester signed in using MFA more than one hour ago, then the condition returns *false*.
- **Not present** – If the requester made a request using their IAM user access keys in the AWS CLI or AWS API, the key is not present. In this case, the key is not present, and it won't match.
- **Null** – For condition keys that are defined by the user, such as passing tags in a request, it is possible to include an empty string. In this case, the value in the request context is *null*. A null value might return true in some cases. For example, if you use the multivalued [ForAllValues \(p. 622\)](#) condition

operator with the [aws:TagKeys \(p. 674\)](#) condition key, you can experience unexpected results if the request context returns `null`. For more information, see [aws:TagKeys \(p. 674\)](#) and [Using Multiple Keys and Values \(p. 622\)](#).

The Condition Block

The following example shows the basic format of a Condition element:

```
"Condition": {  
    "DateGreaterThan" : {  
        "aws:CurrentTime" : "2013-12-15T12:00:00Z"  
    }  
}
```

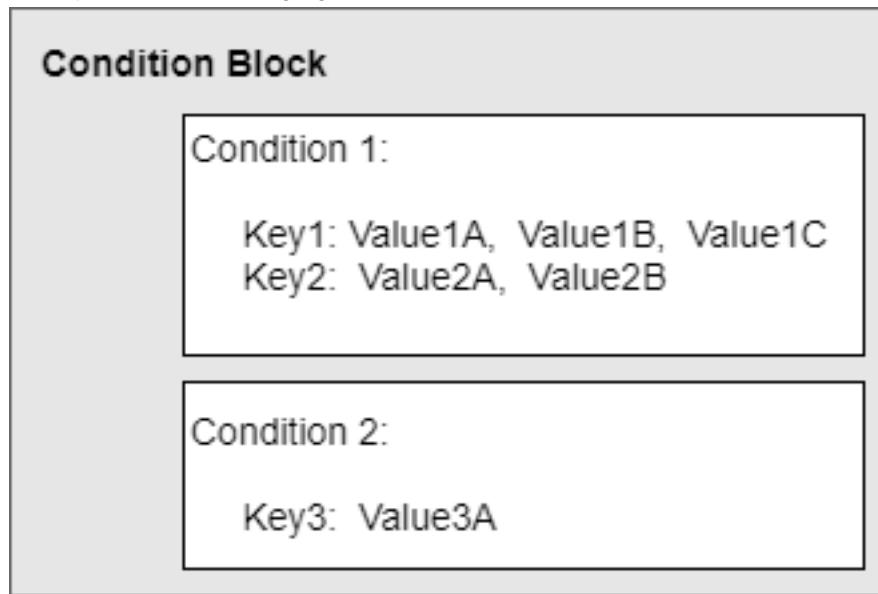
A value from the request is represented by a key, in this case `aws:CurrentTime`. The key value is compared to a value that you specify either as a literal value (`2013-08-16T12:00:00Z`) or as a policy variable, as explained later. The type of comparison to make is specified by the [condition operator \(p. 614\)](#) (here, `DateGreaterThan`). You can create conditions that compare strings, dates, numbers, and so on, using typical Boolean comparisons like equals, greater than, and less than.

Under some circumstances, keys can contain multiple values. For example, a request to Amazon DynamoDB might ask to return or update multiple attributes from a table. A policy for access to DynamoDB tables can include the `dynamodb:Attributes` key, which contains all the attributes listed in the request. You can test the multiple attributes in the request against a list of allowed attributes in a policy by using set operators in the Condition element. For more information, see [Creating a Condition with Multiple Keys or Values \(p. 621\)](#).

When the policy is evaluated during a request, AWS replaces the key with the corresponding value from the request. (In this example, AWS would use the date and time of the request.) The condition is evaluated to return true or false, which is then factored into whether the policy as a whole allows or denies the request.

Multiple Values in a Condition

A Condition element can contain multiple conditions, and each condition can contain multiple key-value pairs. The following figure illustrates this.



For more information, see [Creating a Condition with Multiple Keys or Values \(p. 621\)](#).

IAM JSON Policy Elements: Condition Operators

Use condition operators in the Condition element to match the condition key and value in the policy against values in the request context. For more information about the Condition element, see [IAM JSON Policy Elements: Condition \(p. 611\)](#).

The condition operator that you can use in a policy depends on the condition key you choose. You can choose a global condition key or a service-specific condition key. To learn which condition operator you can use for a global condition key, see [AWS Global Condition Context Keys \(p. 665\)](#). To learn which condition operator you can use for a service-specific condition key, see [Actions, Resources, and Condition Keys for AWS Services \(p. 686\)](#) and choose the service that you want to view.

Important

If the key that you specify in a policy condition is not present in the request context, the values do not match. This applies to all condition operators except [...IfExists \(p. 620\)](#) and [Null check \(p. 621\)](#). These operators test whether the key is present (exists) in the request context.

The condition operators can be grouped into the following categories:

- [String \(p. 614\)](#)
- [Numeric \(p. 616\)](#)
- [Date and time \(p. 616\)](#)
- [Boolean \(p. 617\)](#)
- [Binary \(p. 618\)](#)
- [IP address \(p. 618\)](#)
- [Amazon Resource Name \(ARN\) \(p. 619\)](#) (available for only some services.)
- [...IfExists \(p. 620\)](#) (checks if the key value exists as part of another check)
- [Null check \(p. 621\)](#) (checks if the key value exists as a standalone check)

String Condition Operators

String condition operators let you construct Condition elements that restrict access based on comparing a key to a string value.

Condition Operator	Description
StringEquals	Exact matching, case sensitive
StringNotEquals	Negated matching
StringEqualsIgnoreCase	Exact matching, ignoring case
StringNotEqualsIgnoreCase	Negated matching, ignoring case
StringLike	Case-sensitive matching. The values can include a multi-character match wildcard (*) or a single-character match wildcard (?) anywhere in the string. Note If a key contains multiple values, StringLike can be qualified with set operators — ForAllValues:StringLike and ForAnyValue:StringLike . For more information, see Creating a Condition with Multiple Keys or Values (p. 621) .

Condition Operator	Description
StringNotLike	Negated case-sensitive matching. The values can include a multi-character match wildcard (*) or a single-character match wildcard (?) anywhere in the string.

For example, the following statement contains a Condition element that uses the StringEquals condition operator with the aws:PrincipalTag key to specify that the principal making the request must be tagged with the iamuser-admin job category.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:*AccessKey*",
    "Resource": "arn:aws:iam::ACCOUNT-ID-WITHOUT-HYPHENS:user/*",
    "Condition": {"StringEquals": {"aws:PrincipalTag/job-category": "iamuser-admin"}}
  }
}
```

If the key that you specify in a policy condition is not present in the request context, the values do not match. In this example, the aws:PrincipalTag/job-category key is present in the request context if the principal is using an IAM user with attached tags. It is also included for a principal using an IAM role with attached tags or session tags. If a user without the tag attempts to view or edit an access key, the condition returns false and the request is implicitly denied by this statement.

The following example uses the StringLike condition operator to perform string matching with a [policy variable \(p. 628\)](#) to create a policy that lets an IAM user use the Amazon S3 console to manage his or her own "home directory" in an Amazon S3 bucket. The policy allows the specified actions on an S3 bucket as long as the s3:prefix matches any one of the specified patterns.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3>ListAllMyBuckets",
        "s3:GetBucketLocation"
      ],
      "Resource": "arn:aws:s3::::*"
    },
    {
      "Effect": "Allow",
      "Action": "s3>ListBucket",
      "Resource": "arn:aws:s3::::BUCKET-NAME",
      "Condition": {"StringLike": {"s3:prefix": [
          "",
          "home/",
          "home/${aws:username}/"
        ]}}
    },
    {
      "Effect": "Allow",
      "Action": "s3:*",
      "Resource": [
        "arn:aws:s3::::BUCKET-NAME/home/${aws:username}",
        "arn:aws:s3::::BUCKET-NAME/home/${aws:username}/*"
      ]
    }
  ]
}
```

```
    ]  
}
```

For an example of a policy that shows how to use the `Condition` element to restrict access to resources based on an application ID and a user ID for web identity federation, see [Amazon S3: Allows Amazon Cognito Users to Access Objects in Their Bucket \(p. 436\)](#).

Numeric Condition Operators

Numeric condition operators let you construct `Condition` elements that restrict access based on comparing a key to an integer or decimal value.

Condition Operator	Description
<code>NumericEquals</code>	Matching
<code>NumericNotEquals</code>	Negated matching
<code>NumericLessThan</code>	"Less than" matching
<code>NumericLessThanEquals</code>	"Less than or equals" matching
<code>NumericGreaterThan</code>	"Greater than" matching
<code>NumericGreaterThanEquals</code>	"Greater than or equals" matching

For example, the following statement contains a `Condition` element that uses the `NumericLessThanEquals` condition operator with the `s3:max-keys` key to specify that the requester can list *up to 10* objects in `example_bucket` at a time.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3>ListBucket",
    "Resource": "arn:aws:s3:::example_bucket",
    "Condition": {"NumericLessThanEquals": {"s3:max-keys": "10"}}
  }
}
```

If the key that you specify in a policy condition is not present in the request context, the values do not match. In this example, the `s3:max-keys` key is always present in the request when you perform the `ListBucket` operation. If this policy allowed all Amazon S3 operations, then only the operations that include the `max-keys` context key with a value of less than or equal to 10 would be allowed.

Date Condition Operators

Date condition operators let you construct `Condition` elements that restrict access based on comparing a key to a date/time value. You use these condition operators with the `aws:CurrentTime` key or `aws:EpochTime` keys. You must specify date/time values with one of the [W3C implementations of the ISO 8601 date formats](#) or in epoch (UNIX) time.

Note

Wildcards are not permitted for date condition operators.

Condition Operator	Description
<code>DateEquals</code>	Matching a specific date

Condition Operator	Description
DateNotEquals	Negated matching
DateLessThan	Matching before a specific date and time
DateLessThanEquals	Matching at or before a specific date and time
DateGreaterThanOrEqual	Matching after a specific date and time
DateGreaterThanOrEqual	Matching at or after a specific date and time

For example, the following statement contains a Condition element that uses the DateLessThan condition operator with the aws:TokenIssueTime key. This condition specifies that the temporary security credentials used to make the request were issued in 2020. This policy can be updated programmatically every day to ensure that account members use fresh credentials.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:*AccessKey*",
    "Resource": "arn:aws:iam::ACCOUNT-ID-WITHOUT-HYPHENS:user/*",
    "Condition": {"DateGreaterThanOrEqual": {"aws:TokenIssueTime": "2020-01-01T00:00:00Z"}}
  }
}
```

If the key that you specify in a policy condition is not present in the request context, the values do not match. The aws:TokenIssueTime key is present in the request context only when the principal uses temporary credentials to make the request. They key is not present in AWS CLI, AWS API, or AWS SDK requests that are made using access keys. In this example, if an IAM user attempts to view or edit an access key, the request is denied.

Boolean Condition Operators

Boolean conditions let you construct Condition elements that restrict access based on comparing a key to "true" or "false."

Condition Operator	Description
Bool	Boolean matching

For example, the following statement uses the Bool condition operator with the aws:SecureTransport key to specify that the request must use SSL.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:*AccessKey*",
    "Resource": "arn:aws:iam::ACCOUNT-ID-WITHOUT-HYPHENS:user/*",
    "Condition": {"Bool": {"aws:SecureTransport": "true"}}
  }
}
```

If the key that you specify in a policy condition is not present in the request context, the values do not match. The aws:SecureTransport key is always present in the request context.

Binary Condition Operators

The `BinaryEquals` condition operator let you construct Condition elements that test key values that are in binary format. It compares the value of the specified key byte for byte against a [base-64](#) encoded representation of the binary value in the policy.

```
"Condition" : {
    "BinaryEquals": {
        "key" : "QmluYXJ5VmFsdWVJbkJhc2U2NA=="
    }
}
```

If the key that you specify in a policy condition is not present in the request context, the values do not match.

IP Address Condition Operators

IP address condition operators let you construct Condition elements that restrict access based on comparing a key to an IPv4 or IPv6 address or range of IP addresses. You use these with the `aws:SourceIp` key. The value must be in the standard CIDR format (for example, `203.0.113.0/24` or `2001:DB8:1234:5678::/64`). If you specify an IP address without the associated routing prefix, IAM uses the default prefix value of `/32`.

Some AWS services support IPv6, using `::` to represent a range of 0s. To learn whether a service supports IPv6, see the documentation for that service.

Condition Operator	Description
<code>IpAddress</code>	The specified IP address or range
<code>NotIpAddress</code>	All IP addresses except the specified IP address or range

For example, the following statement uses the `IpAddress` condition operator with the `aws:SourceIp` key to specify that the request must come from the IP range `203.0.113.0` to `203.0.113.255`.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "iam:*AccessKey*",
        "Resource": "arn:aws:iam::ACCOUNT-ID-WITHOUT-HYPHENS:user/*",
        "Condition": {"IpAddress": {"aws:SourceIp": "203.0.113.0/24"}}
    }
}
```

The `aws:SourceIp` condition key resolves to the IP address that the request originates from. If the requests originates from an Amazon EC2 instance, `aws:SourceIp` evaluates to the instance's public IP address.

If the key that you specify in a policy condition is not present in the request context, the values do not match. The `aws:SourceIp` key is always present in the request context, except when the requester uses a VPC endpoint to make the request. In this case, the condition returns `false` and the request is implicitly denied by this statement.

The following example shows how to mix IPv4 and IPv6 addresses to cover all of your organization's valid IP addresses. We recommend that you augment your organization's policies with your IPv6 address

ranges in addition to IPv4 ranges you already have to ensure the policies continue to work as you make the transition to IPv6.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "someservice:*",
      "Resource": "*",
      "Condition": {
        "IpAddress": {
          "aws:SourceIp": [
            "203.0.113.0/24",
            "2001:DB8:1234:5678::/64"
          ]
        }
      }
    }
  ]
}
```

The `aws:SourceIp` condition key works only in a JSON policy if you are calling the tested API directly as a user. If you instead use a service to call the target service on your behalf, the target service sees the IP address of the calling service rather than the IP address of the originating user. This can happen, for example, if you use AWS CloudFormation to call Amazon EC2 to construct instances for you. There is currently no way to pass the originating IP address through a calling service to the target service for evaluation in a JSON policy. For these types of service API calls, do not use the `aws:SourceIp` condition key.

Amazon Resource Name (ARN) Condition Operators

Amazon Resource Name (ARN) condition operators let you construct `Condition` elements that restrict access based on comparing a key to an ARN. The ARN is considered a string. This value is available for only some services; not all services support request values that can be compared as ARNs.

Condition Operator	Description
<code>ArnEquals</code> , <code>ArnLike</code>	Case-sensitive matching of the ARN. Each of the six colon-delimited components of the ARN is checked separately and each can include a multi-character match wildcard (*) or a single-character match wildcard (?). These behave identically.
<code>ArnNotEquals</code> , <code>ArnNotLike</code>	Negated matching for ARN. These behave identically.

The following resource-based policy example shows a policy attached to an Amazon SQS queue to which you want to send SNS messages. It gives Amazon SNS permission to send messages to the queue (or queues) of your choice, but only if the service is sending the messages on behalf of a particular Amazon SNS topic (or topics). You specify the queue in the `Resource` field, and the Amazon SNS topic as the value for the `SourceArn` key.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {"AWS": "123456789012"},
      "Action": "SQS:SendMessage",
      "Resource": "arn:aws:sqs:REGION:123456789012:QUEUE-ID",
      "Condition": {"ArnEquals": {"aws:SourceArn": "arn:aws:sns:REGION:123456789012:TOPIC-ID"}}
    }
  ]
}
```

```
}
```

If the key that you specify in a policy condition is not present in the request context, the values do not match. The `aws:SourceArn` key is present in the request context only if a resource triggers a service to call another service on behalf of the resource owner. If an IAM user attempts to perform this operation directly, the condition returns `false` and the request is implicitly denied by this statement.

[...IfExists Condition Operators](#)

You can add `IfExists` to the end of any condition operator name except the `Null` condition—for example, `StringLikeIfExists`. You do this to say "If the policy key is present in the context of the request, process the key as specified in the policy. If the key is not present, evaluate the condition element as true." Other condition elements in the statement can still result in a nonmatch, but not a missing key when checked with `...IfExists`.

[Example using IfExists](#)

Many condition keys describe information about a certain type of resource and only exist when accessing that type of resource. These condition keys are not present on other types of resources. This doesn't cause an issue when the policy statement applies to only one type of resource. However, there are cases where a single statement can apply to multiple types of resources, such as when the policy statement references actions from multiple services or when a given action within a service accesses several different resource types within the same service. In such cases, including a condition key that applies to only one of the resources in the policy statement can cause the `Condition` element in the policy statement to fail such that the statement's `"Effect"` does not apply.

For example, consider the following policy example:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "THISPOLICYDOESNOTWORK",
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
      "Resource": "*",
      "Condition": {"StringLike": {"ec2:InstanceType": [
        "t1.*",
        "t2.*",
        "m3.*"
      ]}}
    }
  ]
}
```

The *intent* of the preceding policy is to enable the user to launch any instance that is type t1, t2 or m3. However, launching an instance actually requires accessing many resources in addition to the instance itself; for example, images, key pairs, security groups, etc. The entire statement is evaluated against every resource that is required to launch the instance. These additional resources do not have the `ec2:InstanceType` condition key, so the `StringLike` check fails, and the user is not granted the ability to launch *any* instance type. To address this, use the `StringLikeIfExists` condition operator instead. This way, the test only happens if the condition key exists. You could read the following as: "If the resource being checked has an `"ec2:InstanceType"` condition key, then allow the action only if the key value begins with `"t1.*"`, `"t2.*"`, or `"m3.*"`. If the resource being checked does not have that condition key, then don't worry about it." The `DescribeActions` statement includes the actions required to view the instance in the console.

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

```

    "Sid": "RunInstance",
    "Effect": "Allow",
    "Action": "ec2:RunInstances",
    "Resource": "*",
    "Condition": {
        "StringLikeIfExists": {
            "ec2:InstanceType": [
                "t1.*",
                "t2.*",
                "m3.*"
            ]
        }
    },
    {
        "Sid": "DescribeActions",
        "Effect": "Allow",
        "Action": [
            "ec2:DescribeImages",
            "ec2:DescribeInstances",
            "ec2:DescribeVpcs",
            "ec2:DescribeKeyPairs",
            "ec2:DescribeSubnets",
            "ec2:DescribeSecurityGroups"
        ],
        "Resource": "*"
    }
}

```

Condition Operator to Check Existence of Condition Keys

Use a `Null` condition operator to check if a condition key is present at the time of authorization. In the policy statement, use either `true` (the key doesn't exist — it is null) or `false` (the key exists and its value is not null).

For example, you can use this condition operator to determine whether a user is using their own credentials for the operation or temporary credentials. If the user is using temporary credentials, then the key `aws:TokenIssueTime` exists and has a value. The following example shows a condition that states that the user must not be using temporary credentials (the key must not exist) for the user to use the Amazon EC2 API.

```
{
    "Version": "2012-10-17",
    "Statement":{
        "Action": "ec2:*",
        "Effect": "Allow",
        "Resource": "*",
        "Condition": {"Null": {"aws:TokenIssueTime": "true"}}
    }
}
```

Creating a Condition with Multiple Keys or Values

You can use the `Condition` element of a policy to test multiple keys or multiple values for a single key in a request. When you make a request to AWS, either programmatically or through the AWS Management Console, your request includes information about your principal, operation, tags, and more. To learn about information and data included in a request, see [Request \(p. 5\)](#). You can use condition keys to test the values of the matching keys in the request. For example, you can use a condition key to control access to specific attributes of a DynamoDB table or to an Amazon EC2 instance based on tags.

A `Condition` element can contain multiple conditions, and each condition can contain multiple key-value pairs. Most condition keys support using multiple values. The following figure illustrates this. Unless otherwise specified, all keys can have multiple values.

Condition Block

Condition 1:

Key1: Value1A, Value1B, Value1C
Key2: Value2A, Value2B

Condition 2:

Key3: Value3A

Topics

- [Evaluation Logic for Conditions with Multiple Keys or Values \(p. 622\)](#)
- [Using Multiple Keys and Values \(p. 622\)](#)
- [Examples of Using Multiple Values with Condition Set Operators \(p. 624\)](#)
- [Evaluation Logic for Multiple Values with Condition Set Operators \(p. 626\)](#)

Evaluation Logic for Conditions with Multiple Keys or Values

If your policy has multiple condition operators or multiple keys attached to a single condition operator, the conditions are evaluated using a logical AND. If a single condition operator includes multiple values for one key, that condition operator is evaluated using a logical OR. All conditions must resolve to true to trigger the desired Allow or Deny effect.

Condition Block

AND

Condition 1:

Key1: Value1A **OR** Value1B **OR** Value1C
AND
Key2: Value2A **OR** Value2B

Condition 2:

Key3: Value3A

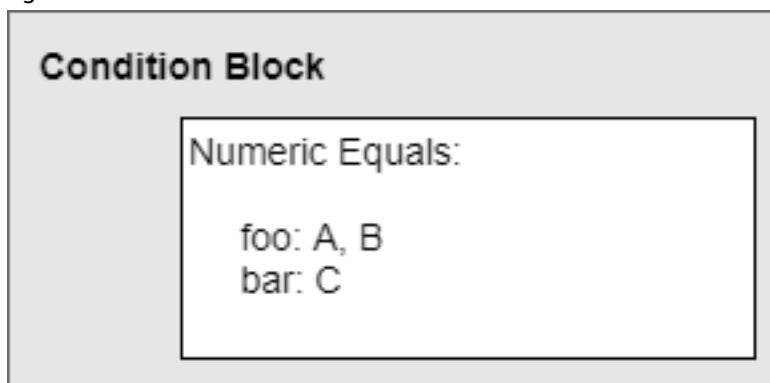
Using Multiple Keys and Values

For requests that include multiple values for a single key, you must enclose the conditions within brackets like an array ("Key2":["Value2A", "Value2B"]). You must also use the `ForAllValues` or `ForAnyValue` set operators with the `StringLike` condition operator (p. 614). These qualifiers add

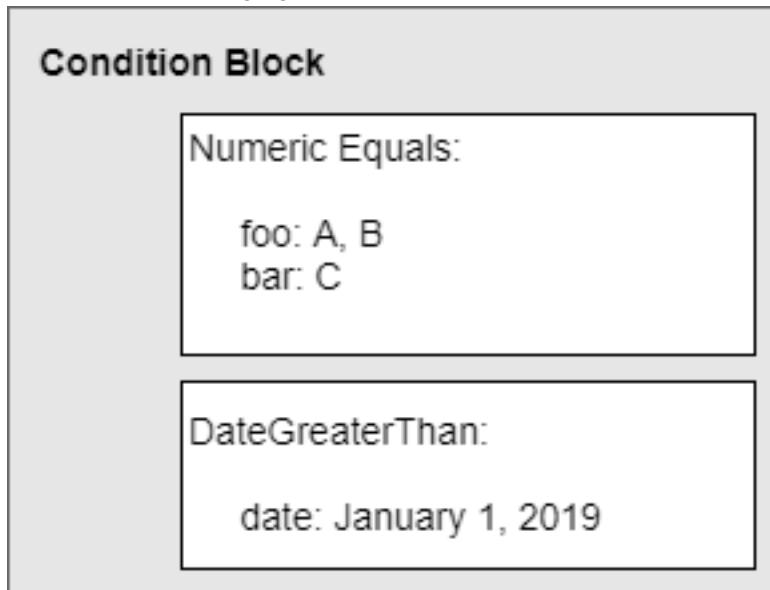
set-operation functionality to the condition operator so that you can test multiple request values against multiple condition values.

- **ForAllValues** – Tests whether the value of every member of the request set is a subset of the condition key set. The condition returns true if every key value in the request matches at least one value in the policy. It also returns true if there are no keys in the request, or if the key values resolve to a null data set, such as an empty string.
- **ForAnyValue** – Tests whether at least one member of the set of request values matches at least one member of the set of condition key values. The condition returns true if any one of the key values in the request matches any one of the condition values in the policy. For no matching key or a null dataset, the condition returns false.

Assume that you want to let John use a resource only if a numeric value *foo* equals either A or B, and another numeric value *bar* equals C. You would create a condition block that looks like the following figure.



Assume that you also want to restrict John's access to after January 1, 2019. You would add another condition, DateGreaterThan, with a date equal to January 1, 2019. The condition block would then look like the following figure.



AWS has predefined condition operators and keys (like `aws:currentTime`). Individual AWS services also define service-specific keys.

As an example, assume that you want to let user John access your Amazon SQS queue under the following conditions:

- The time is after 12:00 p.m. on 7/16/2019
- The time is before 3:00 p.m. on 7/16/2019
- The request comes from an IP address within the range 192.0.2.0 to 192.0.2.255 or 203.0.113.0 to 203.0.113.255.

Your condition block has three separate condition operators, and all three of them must be met for John to have access to your queue, topic, or resource.

The following shows what the condition block looks like in your policy. The two values for `aws:SourceIp` are evaluated using OR. The three separate condition operators are evaluated using AND.

```

"Condition" : {
    "DateGreaterThan" : {
        "aws:CurrentTime" : "2019-07-16T12:00:00Z"
    },
    "DateLessThan": {
        "aws:CurrentTime" : "2019-07-16T15:00:00Z"
    },
    "IpAddress" : {
        "aws:SourceIp" : ["192.0.2.0/24", "203.0.113.0/24"]
    }
}

```

Examples of Using Multiple Values with Condition Set Operators

You can create a policy to test multiple values in a request against one or more values that you specify in the policy. Assume that you have an Amazon DynamoDB table named `Thread` that is used to store information about threads in a technical support forum. The table has attributes named `ID`, `UserName`, `PostDateTime`, `Message`, and `Tags`.

```

{
    ID=101
    UserName=Bob
    PostDateTime=20130930T231548Z
    Message="A good resource for this question is docs.aws.amazon.com"
    Tags=[ "AWS", "Database", "Security" ]
}

```

For information about how set operators are used in DynamoDB to implement fine-grained access to individual data items and attributes, see [Fine-Grained Access Control for DynamoDB](#) in the *Amazon DynamoDB Developer Guide*.

You can create a policy that allows users to see only the `PostDateTime`, `Message`, and `Tags` attributes. If the user's request contains any of these attributes, it is allowed. But if the request contains any other attributes (for example, `ID`), the request is denied. Logically speaking, you want to create a list of allowed attributes (`PostDateTime`, `Message`, `Tags`). You also want to indicate in the policy that all of the user's requested attributes must be in that list of allowed attributes.

The following example policy shows how to use the `ForAllValues` qualifier with the `StringEquals` condition operator. The condition allows a user to request *only* the attributes `ID`, `Message`, or `Tags` from the DynamoDB table named `Thread`.

```

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

```

```

    "Effect": "Allow",
    "Action": "dynamodb:GetItem",
    "Resource": "arn:aws:dynamodb:*:*:table/Thread",
    "Condition": {
        "ForAllValues:StringEquals": {
            "dynamodb:Attributes": [
                "ID",
                "Message",
                "Tags"
            ]
        }
    }
}

```

Assume the user makes a request to DynamoDB to get the attributes `Message` and `Tags` from the `Thread` table. In that case, the request is allowed because the user's requested attributes all match values specified in the policy. The `GetItem` operation requires the user to pass the `ID` attribute as the database table key, which is also allowed in the policy. However, if the user's request includes the `UserName` attribute, the request fails. The reason is that `UserName` is not within the list of allowed attributes and the `ForAllValues` qualifier requires all requested values to be listed in the policy.

Important

If you use `dynamodb:Attributes`, you must specify the names of all of the primary key and index key attributes for the table. You must also specify any secondary indexes that are listed in the policy. Otherwise, DynamoDB can't use these key attributes to perform the requested action.

Alternatively, you might want to make sure that users are explicitly forbidden to include some attributes in a request, such as the `ID` and `UserName` attributes. For example, you might exclude attributes when the user is updating the DynamoDB table, because an update (PUT operation) should not change certain attributes. In that case, you create a list of forbidden attributes (`ID`, `UserName`). If any of the user's requested attributes match any of the forbidden attributes, the request is denied.

The following example shows how to use the `ForAnyValue` qualifier to deny access to the `ID` and `PostDateTime` attributes if the user tries to perform the `PutItem` action. That is, if the user tries to update either of those attributes in the `Thread` table. Notice that the `Effect` element is set to `Deny`.

```

{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Deny",
        "Action": "dynamodb:PutItem",
        "Resource": "arn:aws:dynamodb:*:*:table/Thread",
        "Condition": {
            "ForAnyValue:StringEquals": {
                "dynamodb:Attributes": [
                    "ID",
                    "PostDateTime"
                ]
            }
        }
    }
}

```

Assume that the user makes a request to update the `PostDateTime` and `Message` attributes of the `Thread` table. The `ForAnyValue` qualifier determines whether any of the requested attributes appear in the list in the policy. In this case, there is one match (`PostDateTime`), so the condition is true. Assuming the other values in the request also match (for example, the resource), the overall policy evaluation returns true. Because the policy's effect is `Deny`, the request is denied.

Imagine the user instead makes a request to perform `PutItem` with just the `UserName` attribute. None of the attributes in the request (just `UserName`) match any of attributes listed in the policy (`ID`, `PostDateTime`). The condition returns false, so the effect of the policy (Deny) is also false, and the request is not denied by this policy. (For the request to succeed, it must be explicitly allowed by a different policy. It is not explicitly denied by this policy, but all requests are implicitly denied.)

Warning

When you use the `ForAllValues` condition operator, it returns true if there are no keys in the request, or if the key values resolve to a null data set, such as an empty string. To require that the request includes at least one value, you must use another condition in the policy. For an example, see [Controlling Access During AWS Requests \(p. 393\)](#).

Evaluation Logic for Multiple Values with Condition Set Operators

This section discusses the specifics of the evaluation logic used with the `ForAllValues` and `ForAnyValue` operators. The following table illustrates possible keys that might be included in a request (`PostDateTime` and `UserName`) and a policy condition that includes the values `PostDateTime`, `Message`, and `Tags`.

Key (in the Request)	Condition Value (in the Policy)
<code>PostDateTime</code>	<code>PostDateTime</code>
<code>UserName</code>	<code>Message</code>
	<code>Tags</code>

The evaluation for the combination is this:

<code>PostDateTime</code> matches <code>PostDateTime</code> ?
<code>PostDateTime</code> matches <code>Message</code> ?
<code>PostDateTime</code> matches <code>Tags</code> ?
<code>UserName</code> matches <code>PostDateTime</code> ?
<code>UserName</code> matches <code>Message</code> ?
<code>UserName</code> matches <code>Tags</code> ?

The result of the condition operator depends on which modifier is used with the policy condition:

- **ForAllValues.** If every key in the request (`PostDateTime` or `UserName`) matches at least one condition value in the policy (`PostDateTime`, `Message`, `Tags`), the condition operator returns true. Stated another way, in order for the condition to be true, (`PostDateTime` must equal `PostDateTime`, `Message`, or `Tags`) *and* (`UserName` must equal `PostDateTime`, `Message`, or `Tags`).
- **ForAnyValue.** If any combination of request value and policy value (any one of the six in the example) returns true, the condition operator returns true.

The following policy includes a `ForAllValues` qualifier:

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Condition": {
            "ForAllValues": [
                {"PostDateTime": "2012-10-17T00:00:00Z"}]
```

```

    "Action": "dynamodb:GetItem",
    "Resource": "arn:aws:dynamodb:***:table/Thread",
    "Condition": {
        "ForAllValues:StringEquals": {
            "dynamodb:Attributes": [
                "PostDateTime",
                "Message",
                "Tags"
            ]
        }
    }
}

```

Suppose that the user makes a request to DynamoDB to get the attributes `PostDateTime` and `UserName`. The evaluation for the combination is this:

<code>PostDateTime</code> matches <code>PostDateTime</code> ?	True
<code>PostDateTime</code> matches <code>Message</code> ?	False
<code>PostDateTime</code> matches <code>Tags</code> ?	False
<code>UserName</code> matches <code>PostDateTime</code> ?	False
<code>UserName</code> matches <code>Message</code> ?	False
<code>UserName</code> matches <code>Tags</code> ?	False

The policy includes the `ForAllValues` condition operator modifier, meaning that there must be at least one match for `PostDateTime` and one match for `UserName`. There's no match for `UserName`, so the condition operator returns false, and the policy does not allow the request.

The following policy includes a `ForAnyValue` qualifier:

```

{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Deny",
        "Action": "dynamodb:PutItem",
        "Resource": "arn:aws:dynamodb:***:table/Thread",
        "Condition": {
            "ForAnyValue:StringEquals": {
                "dynamodb:Attributes": [
                    "ID",
                    "PostDateTime"
                ]
            }
        }
    }
}

```

Notice that the policy includes `"Effect": "Deny"` and the action is `PutItem`. Imagine that the user makes a `PutItem` request that includes the attributes `UserName`, `Message`, and `PostDateTime`. The evaluation is this:

<code>UserName</code> matches <code>ID</code> ?	False
<code>UserName</code> matches <code>PostDateTime</code> ?	False

Messages matches ID?	False
Message matches PostDateTime?	False
PostDateTime matches ID?	False
PostDateTime matches PostDateTime?	True

With the modifier `ForAnyValue`, if any one of these tests returns true, the condition returns true. The last test returns true, so the condition is true; because the `Effect` element is set to `Deny`, the request is denied.

Note

If the key values in the request resolve to an empty data set (for example, an empty string), a condition operator modified by `ForAllValues` returns true. In addition, a condition operator modified by `ForAnyValue` returns false.

IAM Policy Elements: Variables and Tags

Use AWS Identity and Access Management (IAM) policy variables as placeholders when you don't know the exact value of a resource or condition key when you write the policy.

Note

If AWS cannot resolve a variable, this might cause the entire statement to be invalid. For example, if you use the `aws:TokenIssueTime` variable, the variable resolves to a value only when the requester authenticated using temporary credentials (an IAM role). To prevent variables from causing invalid statements, use the [...IfExists condition operator](#). (p. 620)

Topics

- [Introduction \(p. 628\)](#)
- [Tags as Policy Variables \(p. 630\)](#)
- [Where You Can Use Policy Variables \(p. 630\)](#)
- [Request Information That You Can Use for Policy Variables \(p. 632\)](#)
- [For More Information \(p. 635\)](#)

Introduction

In IAM policies, many actions allow you to provide a name for the specific resources that you want to control access to. For example, the following policy allows the user to list, read, and write objects with a prefix `David` in the Amazon S3 bucket `mybucket`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": ["s3>ListBucket"],
      "Effect": "Allow",
      "Resource": ["arn:aws:s3:::mybucket"],
      "Condition": {"StringLike": {"s3:prefix": ["David/*"]}}
    },
    {
      "Action": [
        "s3.GetObject",
        "s3:PutObject"
      ],
      "Effect": "Allow",
      "Resource": ["arn:aws:s3:::mybucket/David/*"]
    }
  ]
}
```

```
[  
}]
```

In some cases, you might not know the exact name of the resource when you write the policy. You might want to generalize the policy so it works for many users without having to make a unique copy of the policy for each user. For example, consider writing a policy to allow each user to have access to his or her own objects in an Amazon S3 bucket, as in the previous example. But don't create a separate policy for each user that explicitly specifies the user's name as part of the resource. Instead, create a single group policy that works for any user in that group.

You can do this by using *policy variables*, a feature that lets you specify placeholders in a policy. When the policy is evaluated, the policy variables are replaced with values that come from the context of the request itself.

The following example shows a policy for an Amazon S3 bucket that uses a policy variable.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": ["s3>ListBucket"],
      "Effect": "Allow",
      "Resource": ["arn:aws:s3:::mybucket"],
      "Condition": {"StringLike": {"s3:prefix": ["${aws:username}/*"]}}
    },
    {
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Effect": "Allow",
      "Resource": ["arn:aws:s3:::mybucket/${aws:username}/*"]
    }
  ]
}
```

When this policy is evaluated, IAM replaces the variable \${aws:username} with the [friendly name \(p. 575\)](#) of the actual current user. This means that a single policy applied to a group of users can control access to a bucket by using the user name as part of the resource's name.

The variable is marked using a \$ prefix followed by a pair of curly braces ({}). Inside the \${ } characters, you can include the name of the value from the request that you want to use in the policy. The values you can use are discussed later on this page.

Note

In order to use policy variables, you must include the `version` element in a statement, and the `version` must be set to a version that supports policy variables. Variables were introduced in version 2012-10-17. Earlier versions of the policy language don't support policy variables. If you don't include the `Version` element and set it to an appropriate version date, variables like \${aws:username} are treated as literal strings in the policy.

A `Version` policy element is different from a policy version. The `Version` policy element is used within a policy and defines the version of the policy language. A policy version, on the other hand, is created when you change a customer managed policy in IAM. The changed policy doesn't overwrite the existing policy. Instead, IAM creates a new version of the managed policy. To learn more about the `Version` policy element see [the section called "Version" \(p. 598\)](#). To learn more about policy versions, see [the section called "Versioning IAM Policies" \(p. 465\)](#).

You can use policy variables in a similar way to allow each user to manage his or her own access keys. A policy that allows a user to programmatically change the access key for user David looks like this:

```
{
```

```

    "Version": "2012-10-17",
    "Statement": [{
        "Action": ["iam:*AccessKey*"],
        "Effect": "Allow",
        "Resource": ["arn:aws:iam::ACCOUNT-ID-WITHOUT-HYPHENS:user/David"]
    }]
}

```

If this policy is attached to user David, that user can change his own access key. As with the policies for accessing user-specific Amazon S3 objects, you would have to create a separate policy for each user that includes the user's name. You would then attach each policy to the individual users.

By using a policy variable, you can create a policy like this:

```

{
    "Version": "2012-10-17",
    "Statement": [{
        "Action": ["iam:*AccessKey*"],
        "Effect": "Allow",
        "Resource": ["arn:aws:iam::ACCOUNT-ID-WITHOUT-HYPHENS:user/${aws:username}"]
    }]
}

```

When you use a policy variable for the user name like this, you don't have to have a separate policy for each individual user. Instead, you can attach this new policy to an IAM group that includes everyone who should be allowed to manage their own access keys. When a user makes a request to modify his or her access key, IAM substitutes the user name from the current request for the \${aws:username} variable and evaluates the policy.

Tags as Policy Variables

In some AWS services you can attach your own custom attributes to resources that are created by those services. For example, you can apply tags to Amazon S3 buckets or to IAM users and roles. These tags are key-value pairs. You define the tag key name and the value associated with that key name. For example, you might create a tag with a **department** key and a **Human Resources** value. For more information about tagging IAM entities, see [Tagging IAM Users and Roles \(p. 295\)](#). For information about tagging resources created by other AWS services, see the documentation for that service. For information about using Tag Editor, see [Working with Tag Editor in the AWS Management Console User Guide](#).

You can tag IAM identities to simplify discovering, organizing, and tracking your IAM resources. You can also tag IAM identities to control access to resources or to tagging itself. To learn more about using tags to control access, see [Controlling Access to and for IAM Users and Roles Using IAM Resource Tags \(p. 389\)](#).

Where You Can Use Policy Variables

You can use policy variables in the **Resource** element and in string comparisons in the **Condition** element.

Resource Element

A policy variable can appear as the last part of the **ARN** that identifies a resource. The following policy might be attached to a group. It gives each of the users in the group full programmatic access to a user-specific object (their own "home directory") in Amazon S3.

```

{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": ["s3>ListBucket"]
        }
    ]
}

```

```

        "Effect": "Allow",
        "Resource": ["arn:aws:s3:::mybucket"],
        "Condition": {"StringLike": {"s3:prefix": ["${aws:username}/*"]}}
    },
    {
        "Action": [
            "s3:GetObject",
            "s3:PutObject"
        ],
        "Effect": "Allow",
        "Resource": ["arn:aws:s3:::mybucket/${aws:username}/*"]
    }
]
}

```

Note

This example uses the `aws:username` key, which returns the user's friendly name (like "Adele" or "David"). Under some circumstances, you might want to use the `aws:userid` key instead, which is a globally unique value. For more information, see [Unique Identifiers \(p. 579\)](#).

The following policy might be used for an IAM group. It gives users in that group the ability to create, use, and delete queues that have their names and that are in the us-east-2 Region.

```

{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ListForConsole",
            "Effect": "Allow",
            "Action": "sns>ListQueues",
            "Resource": "*"
        },
        {
            "Sid": "AllQueueActions",
            "Effect": "Allow",
            "Action": "sns:*",
            "Resource": "arn:aws:sqs:us-east-2:*$aws:username-queue"
        }
    ]
}

```

To replace part of an ARN with a tag value, surround the prefix and key name with `${}`. For example, the following `Resource` element refers to only a bucket that is named the same as the value in the requesting user's `department` tag.

```
"Resource": ["arn:aws:s3:::bucket/${aws:PrincipalTag/department}"]
```

Condition Element

A policy variable can also be used for `Condition` values in any condition that involves the string operators (`StringEquals`, `StringLike`, `StringNotLike`, etc.) or the ARN operators (`ArnEquals`, `ArnLike`, etc.). The following Amazon SNS topic policy gives users in AWS account 999999999999 the ability to manage (perform all actions for) the topic only if the URL matches their AWS user name.

```

{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Principal": {"AWS": "999999999999"},
            "Effect": "Allow",
            "Action": "sns:*",
            "Condition": {"StringLike": {"sns:endpoint": "https://example.com/${aws:username}/*"}}
        }
    ]
}

```

```
    }]
}
```

When referencing a tag in a Condition element expression, use the relevant prefix and key name as the condition key. Then use the value that you want to test in the condition value. For example, the following policy example allows full access to IAM resources, but only if the tag `costCenter` is attached to the resource. The tag must also have a value of either `12345` or `67890`. If the tag has no value, or has any other value, then the request fails.

```
{
  "Version": "2015-01-01",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "iam:*",
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "iam:ResourceTag/costCenter": [ "12345", "67890" ]
        }
      }
    }
  ]
}
```

Request Information That You Can Use for Policy Variables

The values that can be substituted for policy variables must come from the current [request context](#) (p. 636).

Information Available in All Requests

Policies contain keys whose values you can use as policy variables. (Under some circumstances, the keys do not contain a value—see the information that follows this list.)

- **`aws:CurrentTime`** This can be used for conditions that check the date and time.
- **`aws:EpochTime`** This is the date in epoch or Unix time, for use with date/time conditions.
- **`aws:TokenIssueTime`** This is the date and time that temporary security credentials were issued and can be used with date/time conditions. **Note:** This key is only available in requests that are signed using temporary security credentials. For more information about temporary security credentials, see [Temporary Security Credentials](#) (p. 307).
- **`aws:principalType`** This value indicates whether the principal is an account, user, federated, or assumed role—see the explanation that follows later.
- **`aws:SecureTransport`** This is a Boolean value that represents whether the request was sent using SSL.
- **`aws:SourceIp`** This is the requester's IP address, for use with IP address conditions. Refer to [IP Address Condition Operators](#) (p. 618) for information about when `SourceIp` is valid and when you should use a VPC-specific key instead.
- **`aws:UserAgent`** This value is a string that contains information about the requester's client application. This string is generated by the client and can be unreliable. You can only use this context key from the AWS CLI.
- **`aws:userId`** This value is the unique ID for the current user—see the chart that follows.
- **`aws:username`** This is a string containing the [friendly name](#) (p. 575) of the current user—see the chart that follows.
- **`ec2:SourceInstanceARN`** This is the Amazon Resource Name (ARN) of the Amazon EC2 instance from which the request is made. This key is present only when the request comes from an Amazon EC2 instance using an IAM role associated with an EC2 instance profile.

Important

Key names are case-insensitive. For example, `aws:CurrentTime` is equivalent to `AWS:currenttime`.

Principal Key Values

The values for `aws:username`, `aws:userid`, and `aws:principaltype` depend on what type of principal initiated the request. For example, the request could be made using the credentials of an IAM user, an IAM role, or the AWS account root user. The following list shows values for these keys for different types of principals.

- **AWS account root user**
 - `aws:username`: (not present)
 - `aws:userid`: AWS account ID
 - `aws:principaltype`: Account
- **IAM user**
 - `aws:username`: *IAM-user-name*
 - `aws:userid`: *unique ID (p. 579)*
 - `aws:principaltype`: User
- **Federated user**
 - `aws:username`: (not present)
 - `aws:userid`: *account:caller-specified-name*
 - `aws:principaltype`: FederatedUser
- **Web federated user and SAML federated user**

Note

For information about policy keys that are available when you use web identity federation, see [Identifying Users with Web Identity Federation \(p. 188\)](#).

- `aws:username`: (not present)
- `aws:userid`: (not present)
- `aws:principaltype`: AssumedRole
- **Assumed role**
 - `aws:username`: (not present)
 - `aws:userid`: *role-id:caller-specified-role-name*
 - `aws:principaltype`: Assumed role
- **Role assigned to Amazon EC2 instance**
 - `aws:username`: (not present)
 - `aws:userid`: *role-id:ec2-instance-id*
 - `aws:principaltype`: Assumed role
- **Anonymous caller** (Amazon SQS Amazon SNS and Amazon S3 only)
 - `aws:username`: (not present)
 - `aws:userid`: (not present)
 - `aws:principaltype`: Anonymous

For the items in this list, note the following:

- *not present* means that the value is not in the current request information, and any attempt to match it fails and causes the statement to be invalid.
- *role-id* is a unique identifier assigned to each role at creation. You can display the role ID with the AWS CLI command: `aws iam get-role --role-name rolename`

- *caller-specified-name* and *caller-specified-role-name* are names that are passed by the calling process (such as an application or service) when it makes a call to get temporary credentials.
- *ec2-instance-id* is a value assigned to the instance when it is launched and appears on the **Instances** page of the Amazon EC2 console. You can also display the instance ID by running the AWS CLI command: `aws ec2 describe-instances`

Information Available in Requests for Federated Users

Federated users are users who are authenticated using a system other than IAM. For example, a company might have an application for use in-house that makes calls to AWS. It might be impractical to give an IAM identity to every corporate user who uses the application. Instead, the company might use a proxy (middle-tier) application that has a single IAM identity, or the company might use a SAML identity provider (IdP). The proxy application or SAML IdP authenticates individual users using the corporate network. A proxy application can then use its IAM identity to get temporary security credentials for individual users. A SAML IdP can in effect exchange identity information for AWS temporary security credentials. The temporary credentials can then be used to access AWS resources.

Similarly, you might create an app for a mobile device in which the app needs to access AWS resources. In that case, you might use *web identity federation*, where the app authenticates the user using a well-known identity provider like Login with Amazon, Amazon Cognito, Facebook, or Google. The app can then use the user's authentication information from these providers to get temporary security credentials for accessing AWS resources.

The recommended way to use web identity federation is by taking advantage of Amazon Cognito and the AWS mobile SDKs. For more information, see the following:

- [Amazon Cognito Overview](#) in the *AWS Mobile SDK for Android Developer Guide*
- [Amazon Cognito Overview](#) in the *AWS Mobile SDK for iOS Developer Guide*
- [Common Scenarios for Temporary Credentials \(p. 308\)](#).

Service-Specific Information

Requests can also include service-specific keys and values in its request context. Examples include the following:

- `s3:prefix`
- `s3:max-keys`
- `s3:x-amz-acl`
- `sns:Endpoint`
- `sns:Protocol`

For information about service-specific keys that you can use to get values for policy variables, refer to the documentation for the individual services. For example, see the following topics:

- [Bucket Keys in Amazon S3 Policies](#) in the *Amazon Simple Storage Service Developer Guide*.
- [Amazon SNS Keys](#) in the *Amazon Simple Notification Service Developer Guide*.

Special Characters

There are a few special predefined policy variables that have fixed values that enable you to represent characters that otherwise have special meaning. If these special characters are part of the string, you are trying to match and you inserted them literally they would be misinterpreted. For example, inserting an *

asterisk in the string would be interpreted as a wildcard, matching any characters, instead of as a literal *. In these cases, you can use the following predefined policy variables:

- `${*}` - use where you need an * asterisk character.
- `${?}` - use where you need a ? question mark character.
- `${$}` - use where you need a \$ dollar sign character.

These predefined policy variables can be used in any string where you can use regular policy variables.

For More Information

For more information about policies, see the following:

- [Policies and Permissions \(p. 356\)](#)
- [Example IAM Identity-Based Policies \(p. 394\)](#)
- [IAM JSON Policy Elements Reference \(p. 598\)](#)
- [Policy Evaluation Logic \(p. 636\)](#)
- [About Web Identity Federation \(p. 185\)](#)

IAM JSON Policy Elements: Supported Data Types

This section lists the data types that are supported when you specify values in JSON policies. The policy language doesn't support all types for each policy element; for information about each element, see the preceding sections.

- Strings
- Numbers (Ints and Floats)
- Boolean
- Null
- Lists
- Maps
- Structs (which are just nested Maps)

The following table maps each data type to the serialization. Note that all policies must be in UTF-8. For information about the JSON data types, go to [RFC 4627](#).

Type	JSON
String	String
Integer	Number
Float	Number
Boolean	true false
Null	null
Date	String adhering to the W3C Profile of ISO 8601
IpAddress	String adhering to RFC 4632
List	Array

Type	JSON
Object	Object

Policy Evaluation Logic

When a principal tries to use the AWS Management Console, the AWS API, or the AWS CLI, that principal sends a *request* to AWS. When an AWS service receives the request, AWS completes several steps to determine whether to allow or deny the request.

1. **Authentication** – AWS first authenticates the principal that makes the request, if necessary. This step is not necessary for a few services, such as Amazon S3, that allow some requests from anonymous users.
2. **Processing the Request Context (p. 636)** – AWS processes the information gathered in the request to determine which policies apply to the request.
3. **Evaluating Policies Within a Single Account (p. 636)** – AWS evaluates all of the policy types, which affect the order in which the policies are evaluated.
4. **Determining Whether a Request Is Allowed or Denied Within an Account (p. 638)** – AWS then processes the policies against the request context to determine whether the request is allowed or denied.

Processing the Request Context

AWS processes the request to gather the following information into a *request context*:

- **Actions (or operations)** – The actions or operations that the principal wants to perform.
- **Resources** – The AWS resource object upon which the actions or operations are performed.
- **Principal** – The user, role, federated user, or application that sent the request. Information about the principal includes the policies that are associated with that principal.
- **Environment data** – Information about the IP address, user agent, SSL enabled status, or the time of day.
- **Resource data** – Data related to the resource that is being requested. This can include information such as a DynamoDB table name or a tag on an Amazon EC2 instance.

AWS then uses this information to find policies that apply to the request context.

Evaluating Policies Within a Single Account

How AWS evaluates policies depends on the types of policies that apply to the request context. The following policy types, listed in order of frequency, are available for use within a single AWS account. For more information about these policy types, see [Policies and Permissions \(p. 356\)](#). To learn how AWS evaluates policies for cross-account access, see [Cross-Account Policy Evaluation Logic \(p. 643\)](#).

1. **Identity-based policies** – Identity-based policies are attached to an IAM identity (user, group of users, or role) and grant permissions to IAM entities (users and roles). If only identity-based policies apply to a request, then AWS checks all of those policies for at least one `Allow`.
2. **Resource-based policies** – Resource-based policies grant permissions to the principal (account, user, role, or federated user) specified as the principal. The permissions define what the principal can do with the resource to which the policy is attached. If resource-based policies and identity-based policies both apply to a request, then AWS checks all the policies for at least one `Allow`.
3. **IAM permissions boundaries** – Permissions boundaries are an advanced feature that sets the maximum permissions that an identity-based policy can grant to an IAM entity (user or role). When

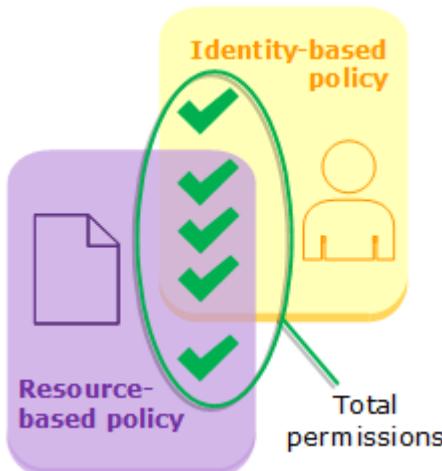
you set a permissions boundary for an entity, the entity can perform only the actions that are allowed by both its identity-based policies and its permissions boundaries. An implicit deny in a permissions boundary does not limit the permissions granted by a resource-based policy.

4. **AWS Organizations service control policies (SCPs)** – Organizations SCPs specify the maximum permissions for an organization or organizational unit (OU). The SCP maximum applies to entities in member accounts, including each AWS account root user. If an SCP is present, identity-based and resource-based policies grant permissions to entities only if those policies and the SCP allow the action. If both a permissions boundary and an SCP are present, then the boundary, the SCP, and the identity-based policy must all allow the action.
5. **Session policies** – Session policies are advanced policies that you pass as parameters when you programmatically create a temporary session for a role or federated user. To create a role session programmatically, use one of the `AssumeRole*` API operations. When you do this and pass session policies, the resulting session's permissions are the intersection of the IAM entity's identity-based policy and the session policies. To create a federated user session, you use an IAM user's access keys to programmatically call the `GetFederationToken` API operation. A resource-based policy has a different effect on the evaluation of session policy permissions. The difference depends on whether the user or role's ARN or the session's ARN is listed as the principal in the resource-based policy. For more information, see [Session Policies \(p. 358\)](#).

Remember, an explicit deny in any of these policies overrides the allow.

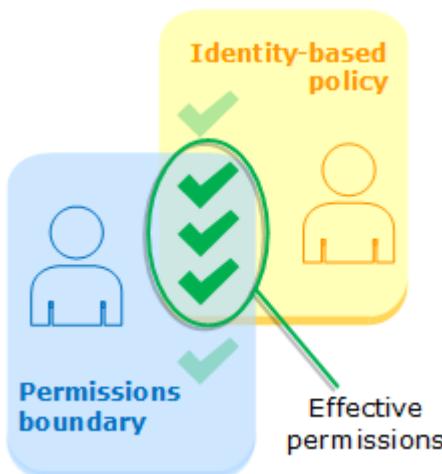
Evaluating Identity-Based Policies with Resource-Based Policies

Identity-based policies and resource-based policies grant permissions to the identities or resources to which they are attached. When an IAM entity (user or role) requests access to a resource within the same account, AWS evaluates all the permissions granted by the identity-based and resource-based policies. The resulting permissions are the total permissions of the two types. If an action is allowed by an identity-based policy, a resource-based policy, or both, then AWS allows the action. An explicit deny in either of these policies overrides the allow.



Evaluating Identity-Based Policies with Permissions Boundaries

When AWS evaluates the identity-based policies and permissions boundary for a user, the resulting permissions are the intersection of the two categories. That means that when you add a permissions boundary to a user with existing identity-based policies, you might reduce the actions that the user can perform. Alternatively, when you remove a permissions boundary from a user, you might increase the actions they can perform. An explicit deny in either of these policies overrides the allow. To view information about how other policy types are evaluated with permissions boundaries, see [Evaluating Effective Permissions with Boundaries \(p. 371\)](#).



Evaluating Identity-Based Policies with Organizations SCPs

When a user belongs to an account that is a member of an organization, the resulting permissions are the intersection of the user's policies and the SCP. This means that an action must be allowed by both the identity-based policy and the SCP. An explicit deny in either of these policies overrides the allow.



You can learn [whether your account is a member of an organization](#) in AWS Organizations. Organization members might be affected by an SCP. To view this data using the AWS CLI command or AWS API operation, you must have permissions for the `organizations:DescribeOrganization` action for your Organizations entity. You must have additional permissions to perform the operation in the Organizations console. To learn whether an SCP is denying access to a specific request, or to change your effective permissions, contact your AWS Organizations administrator.

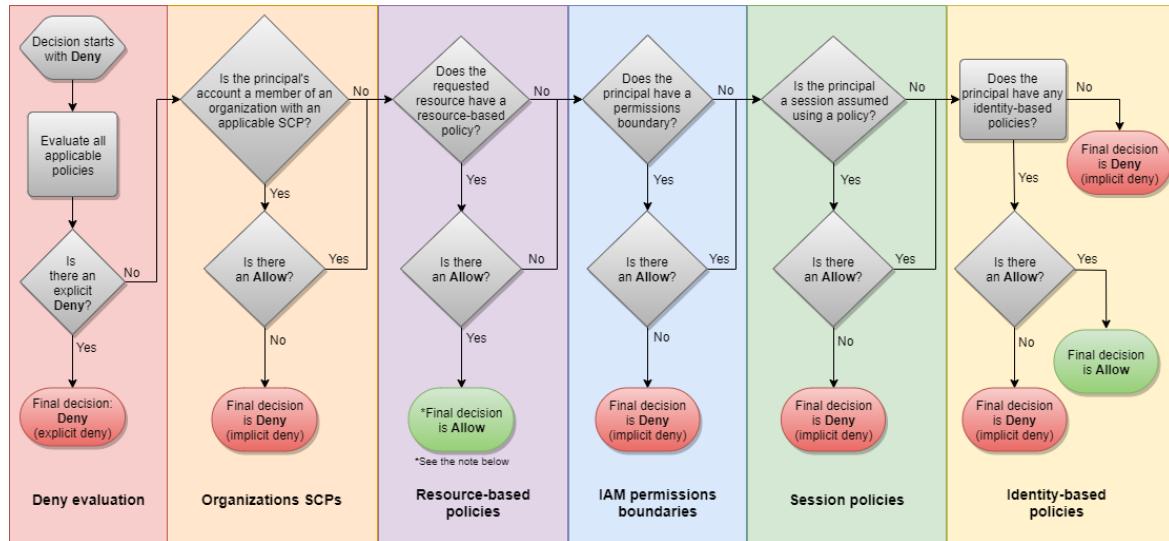
Determining Whether a Request Is Allowed or Denied Within an Account

Assume that a principal sends a request to AWS to access a resource in the same account as the principal's entity. The AWS enforcement code decides whether the request should be allowed or denied. AWS gathers all of the policies that apply to the request context. The following is a high-level summary of the AWS evaluation logic on those policies within a single account.

- By default, all requests are implicitly denied. (Alternatively, by default, the AWS account root user has full access.)

- An explicit allow in an identity-based or resource-based policy overrides this default.
- If a permissions boundary, Organizations SCP, or session policy is present, it might override the allow with an implicit deny.
- An explicit deny in any policy overrides any allows.

The following flow chart provides details about how the decision is made.



1. **Deny evaluation** – By default, all requests are denied. This is called an [implicit deny \(p. 642\)](#). The AWS enforcement code evaluates all policies within the account that apply to the request. These include AWS Organizations SCPs, resource-based policies, IAM permissions boundaries, role session policies, and identity-based policies. In all those policies, the enforcement code looks for a Deny statement that applies to the request. This is called an [explicit deny \(p. 642\)](#). If the code finds even one explicit deny that applies, the code returns a final decision of **Deny**. If there is no explicit deny, the code continues.
2. **Organizations SCPs** – Then the code evaluates AWS Organizations service control policies (SCPs) that apply to the request. SCPs apply if the request is made in an account to which the SCP is attached. If the enforcement code does not find any applicable Allow statements in the SCPs, then the request is implicitly denied. The code returns a final decision of **Deny**. If there is no SCP, or if the SCP allows the requested action, the code continues.
3. **Resource-based policies** – If the requested resource has a resource-based policy that allows the principal to perform the requested action, then the code returns a final decision of **Allow**. If there is no resource-based policy, or if the policy does not include an Allow statement, then the code continues.

Note

This logic can behave differently if you specify the ARN of an IAM role or user as the principal of the resource-based policy. Someone can use session policies to create a temporary credential session for that role or federated user. In that case, the effective permissions for the session might not exceed those allowed by the identity-based policy of the user or role. For more information, see [Session Policies](#).

4. **IAM permissions boundaries** – The enforcement code then checks whether the IAM entity that is used by the principal has a permissions boundary. If the policy that is used to set the permissions boundary does not allow the requested action, then the request is implicitly denied. The code returns a final decision of **Deny**. If there is no permissions boundary, or if the permissions boundary allows the requested action, the code continues.
5. **Session policies** – The code then checks whether the principal is using a session that was assumed by passing a session policy. You can pass a session policy while using the AWS CLI or AWS API to get

temporary credentials for a role or federated user. If the session policy is present and does not allow the requested action, then the request is implicitly denied. The code returns a final decision of **Deny**. If there is no session policy, or if the policy allows the requested action, the code continues.

6. **Identity-based policies** – The code then checks the identity-based policies for the principal. For an IAM user, these include user policies and policies from groups to which the user belongs. If any statement in any applicable identity-based policies allows the requested action, then the enforcement code returns a final decision of **Allow**. If there are no statements that allow the requested action, then the request is implicitly denied, and the code returns a final decision of **Deny**.
7. **Errors** – If the AWS enforcement code encounters an error at any point during the evaluation, then it generates an exception and closes.

Example Identity-Based and Resource-Based Policy Evaluation

The most common types of policies are identity-based policies and resource-based policies.

Assume that Carlos has the user name `carlossalazar` and he tries to save a file to the `carlossalazar-logs` Amazon S3 bucket.

Also assume that the following policy is attached to the `carlossalazar` IAM user.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowS3ListRead",
            "Effect": "Allow",
            "Action": [
                "s3>ListAllMyBuckets",
                "s3:HeadBucket"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowS3Self",
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3::::carlossalazar/*",
                "arn:aws:s3::::carlossalazar"
            ]
        },
        {
            "Sid": "DenyS3Logs",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3::::log*",
                "arn:aws:s3::::log*/**"
            ]
        }
    ]
}
```

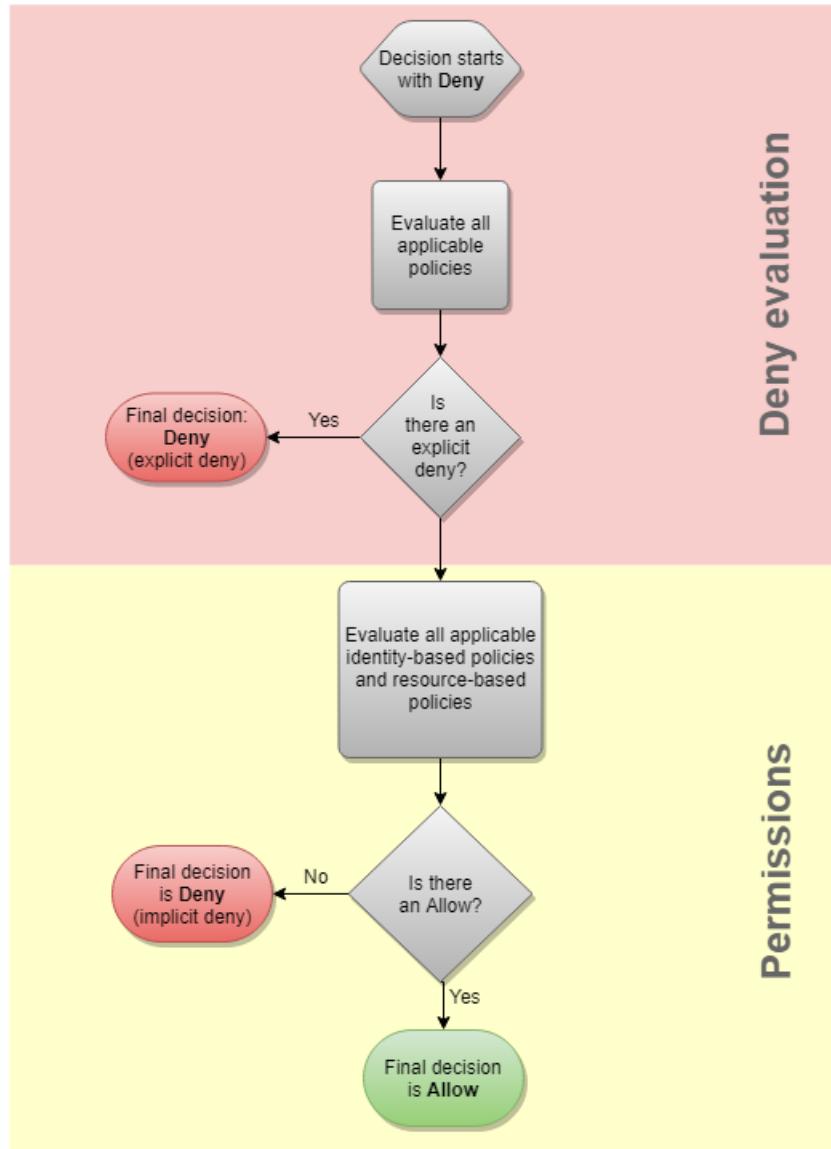
The `AllowS3ListRead` statement in this policy allows Carlos to view a list of all of the buckets in the account. The `AllowS3Self` statement allows Carlos full access to the bucket with the same name as his user name. The `DenyS3Logs` statement denies Carlos access to any S3 bucket with `log` in its name.

Additionally, the following resource-based policy (called a bucket policy) is attached to the `carlossalazar` bucket.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:*",
            "Principal": { "AWS": "arn:aws:iam::111122223333:user/carlossalazar" },
            "Resource": "*"
        }
    ]
}
```

This policy specifies that only the `carlossalazar` user can access the `carlossalazar` bucket.

When Carlos makes his request to save a file to the `carlossalazar-logs` bucket, AWS determines what policies apply to the request. In this case, only the identity-based policy and the resource-based policy apply. These are both permissions policies. Because no permissions boundaries apply, the evaluation logic is reduced to the following logic.



AWS first checks for a Deny statement that applies to the context of the request. It finds one, because the identity-based policy explicitly denies Carlos access to any S3 buckets used for logging. Carlos is denied access.

Assume that he then realizes his mistake and tries to save the file to the `carlossalazar` bucket. AWS checks for a Deny statement and does not find one. It then checks the permissions policies. Both the identity-based policy and the resource-based policy allow the request. Therefore, AWS allows the request. If either of them explicitly denied the statement, the request would have been denied. If one of the policy types allows the request and the other doesn't, the request is still allowed.

The Difference Between Explicit and Implicit Denies

A request results in an explicit deny if an applicable policy includes a Deny statement. If policies that apply to a request include an Allow statement and a Deny statement, the Deny statement trumps the Allow statement. The request is explicitly denied.

An implicit denial occurs when there is no applicable Deny statement but also no applicable Allow statement. Because an IAM user, role, or federated user is denied access by default, they must be explicitly allowed to perform an action. Otherwise, they are implicitly denied access.

When you design your authorization strategy, you must create policies with Allow statements to allow your principals to successfully make requests. However, you can choose any combination of explicit and implicit denies. For example, you can create the following policy to allow an administrator full access to all resources in AWS, but explicitly deny access to billing. If someone adds another policy to this administrator granting them access to billing, it is still denied because of this explicit deny.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "*",
            "Resource": "*"
        },
        {
            "Effect": "Deny",
            "Action": "aws-portal:*",
            "Resource": "*"
        }
    ]
}
```

Alternatively, you can create the following policy to allow a user to manage users, but not groups or any other resources in IAM. Those actions are implicitly denied, as are actions in other services. However, if someone adds a policy to the user that allows them to perform these other actions, then they are allowed.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:AttachUserPolicy",
            "iam>CreateUser",
            "iam>DeleteUser",
            "iam>DeleteUserPolicy",
            "iam:DetachUserPolicy",
            "iam:GetUser",
            "iam:GetUserPolicy",
            "iam>ListAttachedUserPolicies",
            "iam:ListGroups",
            "iam:ListUsers"
        ]
    }
}
```

```
    "iam>ListUserPolicies",
    "iam>ListUsers",
    "iam>PutUserPolicy",
    "iam>UpdateUser"
],
"Resource": "*"
}
```

Cross-Account Policy Evaluation Logic

You can allow a principal in one account to access resources in a second account. This is called cross-account access. When you allow cross-account access, the account where the principal exists is called the *trusted* account. The account where the resource exists is the *trusting* account.

To allow cross-account access, you attach a resource-based policy to the resource that you want to share. You must also attach an identity-based policy to the identity that acts the principal in the request. The resource-based policy in the trusting account must specify the principal of the trusted account that will have access the resource. You can specify the entire account or its IAM users, federated users, IAM roles, or assumed-role sessions. You can also specify an AWS service as a principal. For more information, see [Specifying a Principal \(p. 601\)](#).

The principal's identity-based policy must allow the requested access to the resource in the trusting service. You can do this by specifying the ARN of the resource or by allowing access to all resources (*).

In IAM, you can attach a resource-based policy to an IAM role to allow principals in other accounts to assume that role. The role's resource-based policy is called a role trust policy. After assuming that role, the allowed principals can use the resulting temporary credentials to access multiple resources in your account. This access is defined in the role's identity-based permissions policy. For more information, see

[Here are some basic terms to help you get started with roles.](#)

Role

An IAM identity that you can create in your account that has specific permissions. An IAM role has some similarities to an IAM user. Roles and users are both AWS identities with permissions policies that determine what the identity can and cannot do in AWS. However, instead of being uniquely associated with one person, a role is intended to be assumable by anyone who needs it. Also, a role does not have standard long-term credentials such as a password or access keys associated with it. Instead, when you assume a role, it provides you with temporary security credentials for your role session.

Roles can be used by the following:

- An IAM user in the same AWS account as the role
- An IAM user in a different AWS account than the role
- A web service offered by AWS such as Amazon Elastic Compute Cloud (Amazon EC2)
- An external user authenticated by an external identity provider (IdP) service that is compatible with SAML 2.0 or OpenID Connect, or a custom-built identity broker.

AWS service role

A role that a service assumes to perform actions in your account on your behalf. When you set up some AWS service environments, you must define a role for the service to assume. This service role must include all the permissions required for the service to access the AWS resources that it needs. Service roles vary from service to service, but many allow you to choose your permissions, as long as you meet the documented requirements for that service. Service roles provide access only within your account and cannot be used to grant access to services in other accounts. You can create, modify, and delete a service role from within IAM.

AWS service role for an EC2 instance

A special type of service role that an application running on an Amazon EC2 instance can assume to perform actions in your account. This role is assigned to the EC2 instance when it is launched. Applications running on that instance can retrieve temporary security credentials and perform actions that the role allows. For details about using a service role for an EC2 instance, see [Using an IAM Role to Grant Permissions to Applications Running on Amazon EC2 Instances](#) (p. 269).

AWS service-linked role

A unique type of service role that is linked directly to an AWS service. Service-linked roles are predefined by the service and include all the permissions that the service requires to call other AWS services on your behalf. The linked service also defines how you create, modify, and delete a service-linked role. A service might automatically create or delete the role. It might allow you to create, modify, or delete the role as part of a wizard or process in the service. Or it might require that you use IAM to create or delete the role. Regardless of the method, service-linked roles make setting up a service easier because you don't have to manually add the necessary permissions.

Note

If you are already using a service when it begins supporting service-linked roles, you might receive an email announcing a new role in your account. In this case, the service automatically created the service-linked role in your account. You don't need to take any action to support this role, and you should not manually delete it. For more information, see [A New Role Appeared in My AWS Account](#) (p. 564).

For information about which services support using service-linked roles, see [AWS Services That Work with IAM](#) (p. 586) and look for the services that have **Yes** in the **Service-Linked Role** column. Choose a **Yes** with a link to view the service-linked role documentation for that service. If the service does not include documentation for creating, modifying, or deleting the service-linked role, then you can use the IAM console, AWS CLI, or API. For more information, see [Using Service-Linked Roles](#) (p. 221).

Role chaining

Role chaining occurs when you use a role to assume a second role through the AWS CLI or API. For example, assume that `User1` has permission to assume `RoleA` and `RoleB`. Additionally, `RoleA` has permission to assume `RoleB`. You can assume `RoleA` by using `User1`'s long-term user credentials in the `AssumeRole` API operation. This operation returns `RoleA` short-term credentials. To engage in role chaining, you can use `RoleA`'s short-term credentials to assume `RoleB`.

When you assume a role, you can pass a session tag and set the tag as transitive. Transitive session tags are passed to all subsequent sessions in a role chain. To learn more about session tags, see [Passing Session Tags in AWS STS](#) (p. 299).

Role chaining limits your AWS CLI or AWS API role session to a maximum of one hour. When you use the `AssumeRole` API operation to assume a role, you can specify the duration of your role session with the `DurationSeconds` parameter. You can specify a parameter value of up to 43200 seconds (12 hours), depending on the maximum session duration setting (p. 255) for your role. However, if you assume a role using role chaining and provide a `DurationSeconds` parameter value greater than one hour, the operation fails.

Delegation

The granting of permissions to someone to allow access to resources that you control. Delegation involves setting up a trust between two accounts. The first is the account that owns the resource (the trusting account). The second is the account that contains the

users that need to access the resource (the trusted account). The trusted and trusting accounts can be any of the following:

- The same account.
- Separate accounts that are both under your organization's control.
- Two accounts owned by different organizations.

To delegate permission to access a resource, you create an IAM role (p. 229) in the trusting account that has two policies (p. 178) attached. The *permissions policy* grants the user of the role the needed permissions to carry out the intended tasks on the resource. The *trust policy* specifies which trusted account members are allowed to assume the role.

When you create a trust policy, you cannot specify a wildcard (*) as a principal. The trust policy is attached to the role in the trusting account, and is one-half of the permissions. The other half is a permissions policy attached to the user in the trusted account that allows that user to switch to, or assume the role (p. 255). A user who assumes a role temporarily gives up his or her own permissions and instead takes on the permissions of the role. When the user exits, or stops using the role, the original user permissions are restored. An additional parameter called external ID (p. 233) helps ensure secure use of roles between accounts that are not controlled by the same organization.

Federation

The creation of a trust relationship between an external identity provider and AWS. Users can sign in to a web identity provider, such as [Login with Amazon](#), [Facebook](#), [Google](#), or any IdP that is compatible with [OpenID Connect](#) (OIDC). Users can also sign in to an enterprise identity system that is compatible with Security Assertion Markup Language (SAML) 2.0, such as Microsoft Active Directory Federation Services. When you use OIDC and SAML 2.0 to configure a trust relationship between these external identity providers and AWS, the user is assigned to an IAM role. The user also receives temporary credentials that allow the user to access your AWS resources.

Federated user

Instead of creating an IAM user, you can use existing identities from AWS Directory Service, your enterprise user directory, or a web identity provider. These are known as *federated users*. AWS assigns a role to a federated user when access is requested through an identity provider (p. 184). For more information about federated users, see [Federated Users and Roles](#) (p. 12) in the *IAM User Guide*.

Trust policy

A JSON policy document (p. 651) in which you define the principals that you *trust* to assume the role. A role trust policy is a required resource-based policy (p. 357) that is attached to a role in IAM. The principals (p. 601) that you can specify in the trust policy include users, roles, accounts, and services.

Permissions policy

A permissions document in JSON format in which you define what actions and resources the role can use. The document is written according to the rules of the IAM policy language (p. 597).

Permissions boundary

An advanced feature in which you use policies to limit the maximum permissions that an identity-based policy can grant to a role. You cannot apply a permissions boundary to a service-linked role. For more information, see [Permissions Boundaries for IAM Entities](#) (p. 370).

Principal

An entity in AWS that can perform actions and access resources. A principal can be an AWS account root user, an IAM user, or a role. You can grant permissions to access a resource in one of two ways:

- You can attach a permissions policy to a user (directly, or indirectly through a group) or to a role.
- For those services that support resource-based policies (p. 12), you can identify the principal in the `Principal` element of a policy attached to the resource.

If you reference an AWS account as principal, it generally means any principal defined within that account.

Note

You cannot use a wildcard (*) in the `Principal` element in a role's trust policy.

Role for cross-account access

A role that grants access to resources in one account to a trusted principal in a different account. Roles are the primary way to grant cross-account access. However, some AWS services allow you to attach a policy directly to a resource (instead of using a role as a proxy). These are called resource-based policies, and you can use them to grant principals in another AWS account access to the resource. Some of these resources include Amazon Simple Storage Service (S3) buckets, S3 Glacier vaults, Amazon Simple Notification Service (SNS) topics, and Amazon Simple Queue Service (SQS) queues. To learn which services support resource-based policies, see [AWS Services That Work with IAM \(p. 586\)](#). For more information about resource-based policies, see [How IAM Roles Differ from Resource-based Policies \(p. 292\)](#).

(p. 178). To learn how allowing cross-account access using roles is different from allowing cross-account access using other resource-based policies, see [How IAM Roles Differ from Resource-based Policies \(p. 292\)](#).

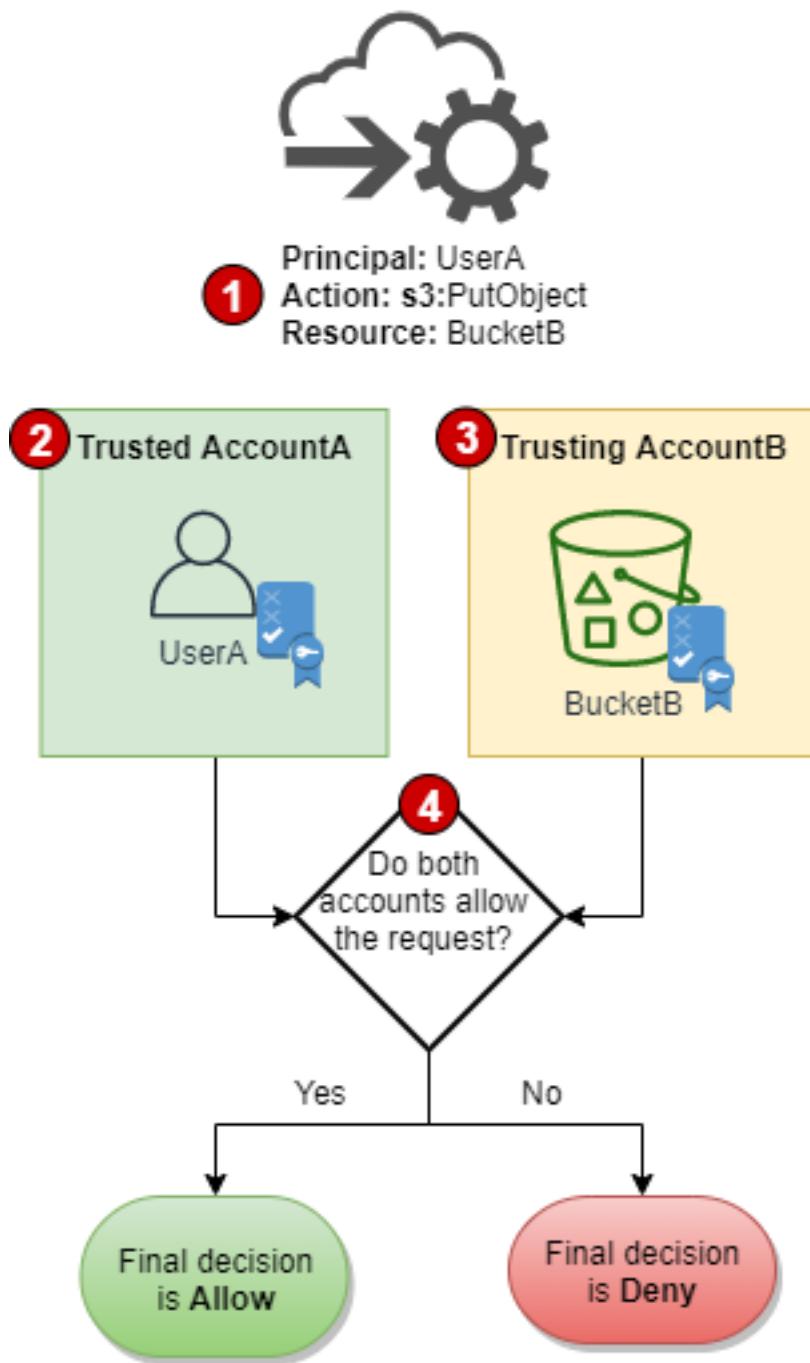
Important

Other services can affect the policy evaluation logic. For example, AWS Organizations supports [service control policies](#) that can be applied to one or more accounts. AWS Resource Access Manager supports [policy fragments](#) that control which actions that principals are allowed to perform on resources that are shared with them.

Determining Whether a Cross-Account Request Is Allowed

For cross-account requests, the requester in the trusted AccountA must have an identity-based policy. That policy must allow them to make a request to the resource in the trusting AccountB. Additionally, the resource-based policy in AccountB must allow the requester in AccountA to access the resource.

When you make a cross-account request, AWS performs two evaluations. AWS evaluates the request in the trusting account and the trusted account. For more information about how a request is evaluated within a single account, see [Determining Whether a Request Is Allowed or Denied Within an Account \(p. 638\)](#). The request is allowed only if both evaluations return a decision of `Allow`.



1. When a principal in one account makes a request to access a resource in another account, this is a cross-account request.
2. The requesting principal exists in the trusted account (AccountA). When AWS evaluates this account, it checks the identity-based policy and any policies that can limit an identity-based policy. For more information, see [Evaluating Policies Within a Single Account \(p. 636\)](#).
3. The requested resource exists in the trusting account (AccountB). When AWS evaluates this account, it checks the resource-based policy that is attached to the requested resource and any policies that can limit a resource-based policy. For more information, see [Evaluating Policies Within a Single Account \(p. 636\)](#).

4. AWS allows the request only if both account policy evaluations allow the request.

Example Cross-Account Policy Evaluation

The following example demonstrates a scenario where a user in one account is granted permissions by a resource-based policy in a second account.

Assume that Carlos is a developer with an IAM user named `carlossalazar` in account 111111111111. He wants to save a file to the `Production-logs` Amazon S3 bucket in account 222222222222.

Also assume that the following policy is attached to the `carlossalazar` IAM user.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowS3ListRead",
            "Effect": "Allow",
            "Action": "s3>ListAllMyBuckets",
            "Resource": "*"
        },
        {
            "Sid": "AllowS3ProductionObjectActions",
            "Effect": "Allow",
            "Action": "s3:*Object*",
            "Resource": "arn:aws:s3:::Production/*"
        },
        {
            "Sid": "DenyS3Logs",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::*log*",
                "arn:aws:s3:::*log*/"
            ]
        }
    ]
}
```

The `AllowS3ListRead` statement in this policy allows Carlos to view a list of all of the buckets in Amazon S3. The `AllowS3ProductionObjectActions` statement allows Carlos full access to objects in the `Production` bucket. The `DenyS3Logs` statement denies Carlos access to any S3 bucket with `log` in its name. It also denies access to all objects in those buckets.

Additionally, the following resource-based policy (called a bucket policy) is attached to the `Production` bucket in account 222222222222.

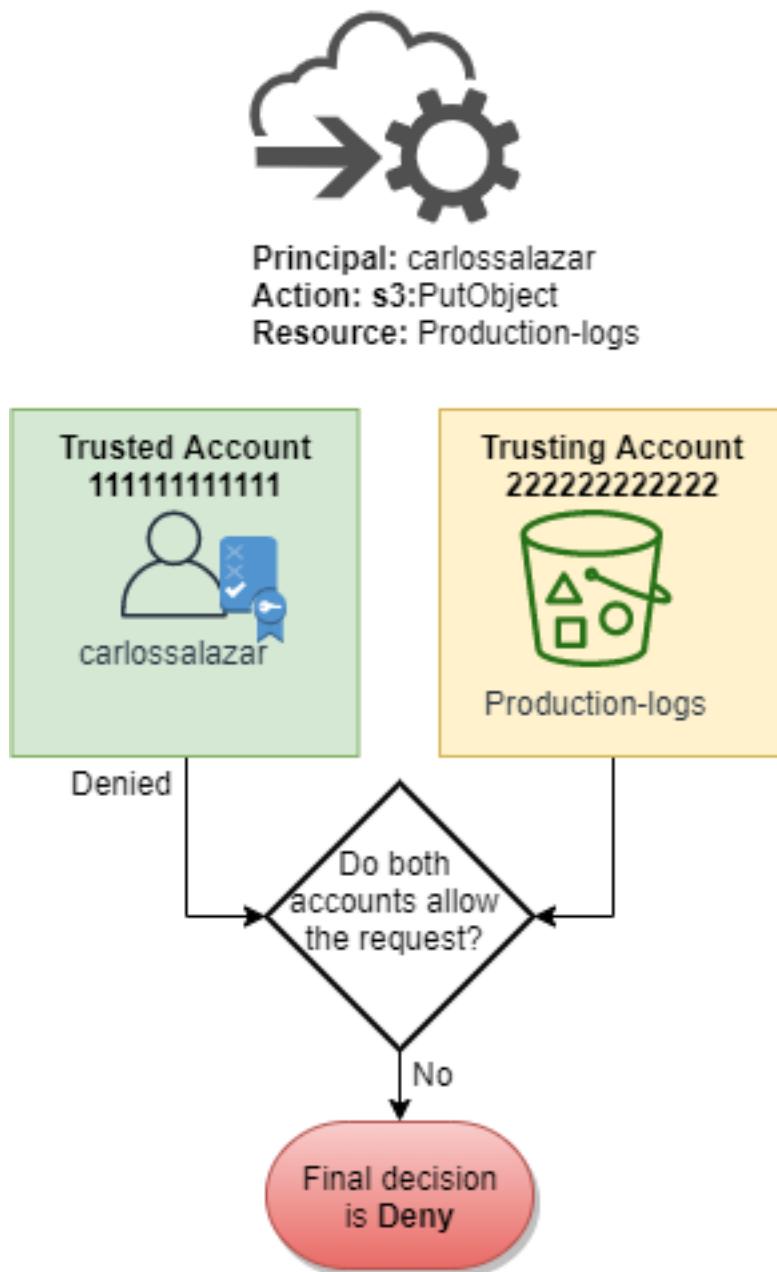
```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject*",
                "s3:PutObject*",
                "s3:ReplicateObject",
                "s3:RestoreObject"
            ],
            "Principal": { "AWS": "arn:aws:iam::111111111111:user/carlossalazar" },
            "Resource": "arn:aws:s3:::Production/*"
        }
    ]
}
```

```
    ]  
}
```

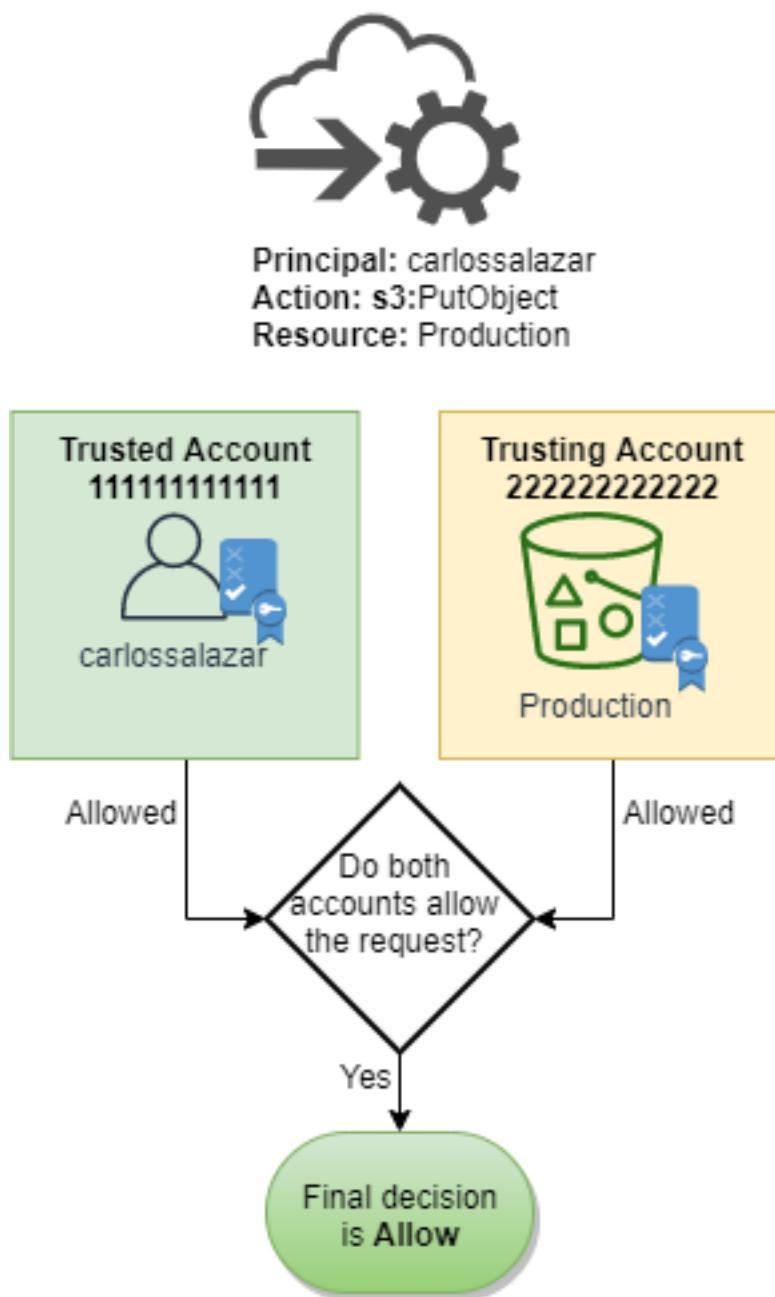
This policy allows the `carlossalazar` user to access objects in the `Production` bucket. He can create and edit, but not delete the objects in the bucket. He can't manage the bucket itself.

When Carlos makes his request to save a file to the `Production-logs` bucket, AWS determines what policies apply to the request. In this case, the identity-based policy attached to the `carlossalazar` user is the only policy that applies in account `111111111111`. In account `222222222222`, there is no resource-based policy attached to the `Production-logs` bucket. When AWS evaluates account `111111111111`, it returns a decision of `Deny`. This is because the `DenyS3Logs` statement in the identity-based policy explicitly denies access to any log buckets. For more information about how a request is evaluated within a single account, see [Determining Whether a Request Is Allowed or Denied Within an Account \(p. 638\)](#).

Because the request is explicitly denied within one of the accounts, the final decision is to deny the request.



Assume that Carlos then realizes his mistake and tries to save the file to the Production bucket. AWS first checks account 111111111111 to determine if the request is allowed. Only the identity-based policy applies, and it allows the request. AWS then checks account 222222222222. Only the resource-based policy attached to the Production bucket applies, and it allows the request. Because both accounts allow the request, the final decision is to allow the request.



Grammar of the IAM JSON Policy Language

This page presents a formal grammar for the language used to create JSON policies in IAM. We present this grammar so that you can understand how to construct and validate policies.

For examples of policies, see the following topics:

- [Policies and Permissions \(p. 356\)](#)
- [Example IAM Identity-Based Policies \(p. 394\)](#)
- [Example Policies for Working in the Amazon EC2 Console](#) and [Example Policies for Working With the AWS CLI, the Amazon EC2 CLI, or an AWS SDK](#) in the [Amazon EC2 User Guide for Linux Instances](#).

- [Bucket Policy Examples](#) and [User Policy Examples](#) in the *Amazon Simple Storage Service Developer Guide*.

For examples of policies used in other AWS services, go to the documentation for those services.

Topics

- [The Policy Language and JSON \(p. 652\)](#)
- [Conventions Used in This Grammar \(p. 652\)](#)
- [Grammar \(p. 653\)](#)
- [Policy Grammar Notes \(p. 654\)](#)

The Policy Language and JSON

Policies are expressed in JSON. When a policy is submitted to IAM, it is first validated to make sure that the JSON syntax is correct. In this document, we do not provide a complete description of what constitutes valid JSON. However, here are some basic JSON rules:

- White space between individual entities is allowed.
- Values are enclosed in quotation marks. Quotation marks are optional for numeric and Boolean values.
- Many elements (for example, `action_string_list` and `resource_string_list`) can take a JSON array as a value. Arrays can take one or more values. If more than one value is included, the array is in square brackets ([and]) and comma-delimited, as in the following example:

```
"Action" : [ "ec2:Describe*", "ec2>List*" ]
```

- Basic JSON data types (Boolean, number, and string) are defined in [RFC 7159](#).

You can use a JSON validator to check the syntax of a policy. You can find a validator online, and many code editors and XML-editing tools include JSON validation features.

Conventions Used in This Grammar

The following conventions are used in this grammar:

- The following characters are JSON tokens and *are* included in policies:
`{ } [] " , :`
- The following characters are special characters in the grammar and are *not* included in policies:
`= < > () |`
- If an element allows multiple values, it is indicated using repeated values, a comma delimiter, and an ellipsis (...). Examples:

```
[<action_string>, <action_string>, ...]
```

```
<principal_map> = { <principal_map_entry>, <principal_map_entry>, ... }
```

If multiple values are allowed, it is also valid to include only one value. For only one value, the trailing comma must be omitted. If the element takes an array (marked with [and]) but only one value is included, the brackets are optional. Examples:

```
"Action": [<action_string>]
```

```
"Action": <action_string>
```

- A question mark (?) following an element indicates that the element is optional. Example:

```
<version_block?>
```

However, be sure to refer to the notes that follow the grammar listing for details about optional elements.

- A vertical line (|) between elements indicates alternatives. In the grammar, parentheses define the scope of the alternatives. Example:

```
("Principal" | "NotPrincipal")
```

- Elements that must be literal strings are enclosed in double quotation marks ("). Example:

```
<version_block> = "Version" : ("2008-10-17" | "2012-10-17")
```

For additional notes, see [Policy Grammar Notes \(p. 654\)](#) following the grammar listing.

Grammar

The following listing describes the policy language grammar. For conventions used in the listing, see the preceding section. For additional information, see the notes that follow.

Note

This grammar describes policies marked with a version of 2008-10-17 and 2012-10-17. A Version policy element is different from a policy version. The Version policy element is used within a policy and defines the version of the policy language. A policy version, on the other hand, is created when you make changes to a customer managed policy in IAM. The changed policy doesn't overwrite the existing policy. Instead, IAM creates a new version of the managed policy. To learn more about the Version policy element see [IAM JSON Policy Elements: Version \(p. 598\)](#). To learn more about policy versions, see the section called "Versioning IAM Policies" (p. 465).

```
policy  = {
    <version_block?>
    <id_block?>
    <statement_block>
}

<version_block> = "Version" : ("2008-10-17" | "2012-10-17")

<id_block> = "Id" : <policy_id_string>

<statement_block> = "Statement" : [ <statement>, <statement>, ... ]

<statement> = {
    <sid_block?>,
    <principal_block?>,
    <effect_block>,
    <action_block>,
    <resource_block>,
    <condition_block?>
}

<sid_block> = "Sid" : <sid_string>

<effect_block> = "Effect" : ("Allow" | "Deny")

<principal_block> = ("Principal" | "NotPrincipal") : ("*" | <principal_map>)

<principal_map> = { <principal_map_entry>, <principal_map_entry>, ... }

<principal_map_entry> = ("AWS" | "Federated" | "Service" | "CanonicalUser") :
    [<principal_id_string>, <principal_id_string>, ...]
```

```

<action_block> = ("Action" | "NotAction") :
    ("*" | [<action_string>, <action_string>, ...])

<resource_block> = ("Resource" | "NotResource") :
    ("*" | [<resource_string>, <resource_string>, ...])

<condition_block> = "Condition" : { <condition_map> }
<condition_map> = {
    <condition_type_string> : { <condition_key_string> : <condition_value_list> },
    <condition_type_string> : { <condition_key_string> : <condition_value_list> }, ...
}
<condition_value_list> = [<condition_value>, <condition_value>, ...]
<condition_value> = ("string" | "number" | "Boolean")

```

Policy Grammar Notes

- A single policy can contain an array of statements.
- Policies have a maximum size between 2048 characters and 10,240 characters, depending on what entity the policy is attached to. For more information, see [IAM and STS Limits \(p. 581\)](#). Policy size calculations do not include white space characters.
- Individual elements must not contain multiple instances of the same key. For example, you cannot include the `Effect` block twice in the same statement.
- Blocks can appear in any order. For example, `version_block` can follow `id_block` in a policy. Similarly, `effect_block`, `principal_block`, `action_block` can appear in any order within a statement.
- The `id_block` is optional in resource-based policies. It must *not* be included in identity-based policies.
- The `principal_block` element is required in resource-based policies (for example, in Amazon S3 bucket policies) and in trust policies for IAM roles. It must *not* be included in identity-based policies.
- The `principal_map` element in Amazon S3 bucket policies can include the CanonicalUser ID. Most resource-based policies do not support this mapping. To learn more about using the canonical user ID in a bucket policy, see [Specifying a Principal in a Policy](#) in the *Amazon Simple Storage Service Developer Guide*.
- Each string value (`policy_id_string`, `sid_string`, `principal_id_string`, `action_string`, `resource_string`, `condition_type_string`, `condition_key_string`, and the string version of `condition_value`) can have its own minimum and maximum length restrictions, specific allowed values, or required internal format.

Notes About String Values

This section provides additional information about string values that are used in different elements in a policy.

`action_string`

Consists of a service namespace, a colon, and the name of an action. Action names can include wildcards. Examples:

```

"Action": "ec2:StartInstances"

"Action": [
    "ec2:StartInstances",
    "ec2:StopInstances"
]

"Action": "cloudformation:*
```

```
"Action": "*"  
  
"Action": [  
    "s3:Get*",  
    "s3>List*"  
]
```

policy_id_string

Provides a way to include information about the policy as a whole. Some services, such as Amazon SQS and Amazon SNS, use the `Id` element in reserved ways. Unless otherwise restricted by an individual service, `policy_id_string` can include spaces. Some services require this value to be unique within an AWS account.

Note

The `id_block` is allowed in resource-based policies, but not in identity-based policies.

There is no limit to the length, although this string contributes to the overall length of the policy, which is limited.

```
"Id": "Admin_Policy"  
  
"Id": "cd3ad3d9-2776-4ef1-a904-4c229d1642ee"
```

sid_string

Provides a way to include information about an individual statement. For IAM policies, basic alphanumeric characters (A-Z,a-z,0-9) are the only allowed characters in the `Sid` value. Other AWS services that support resource policies may have other requirements for the `Sid` value. For example, some services require this value to be unique within an AWS account, and some services allow additional characters such as spaces in the `Sid` value.

```
"Sid": "1"  
  
"Sid": "ThisStatementProvidesPermissionsForConsoleAccess"
```

principal_id_string

Provides a way to specify a principal using the [Amazon Resource Name \(ARN\)](#) (p. 576) of the AWS account, IAM user, IAM role, federated user, or assumed-role user. For an AWS account, you can also use the short form `AWS:accountnumber` instead of the full ARN. For all of the options including AWS services, assumed roles, and so on, see [Specifying a Principal \(p. 601\)](#).

Note that you can use `*` only to specify "everyone/anonymous." You cannot use it to specify part of a name or ARN.

resource_string

In most cases, consists of an [Amazon Resource Name \(ARN\)](#).

```
"Resource": "arn:aws:iam::123456789012:user/Bob"  
  
"Resource": "arn:aws:s3:::examplebucket/*"
```

condition_type_string

Identifies the type of condition being tested, such as `StringEquals`, `StringLike`, `NumericLessThan`, `DateGreaterThanOrEqual`, `Bool`, `BinaryEquals`, `IpAddress`,

`ArnEquals`, etc. For a complete list of condition types, see [IAM JSON Policy Elements: Condition Operators \(p. 614\)](#).

```
"Condition": {  
    "NumericLessThanEquals": {  
        "s3:max-keys": "10"  
    }  
}  
  
"Condition": {  
    "Bool": {  
        "aws:SecureTransport": "true"  
    }  
}  
  
"Condition": {  
    "StringEquals": {  
        "s3:x-amz-server-side-encryption": "AES256"  
    }  
}
```

`condition_key_string`

Identifies the condition key whose value will be tested to determine whether the condition is met. AWS defines a set of condition keys that are available in all AWS services, including `aws:principalType`, `aws:SecureTransport`, and `aws:userId`.

For a list of AWS condition keys, see [AWS Global Condition Context Keys \(p. 665\)](#). For condition keys that are specific to a service, see the documentation for that service such as the following:

- [Specifying Conditions in a Policy](#) in the *Amazon Simple Storage Service Developer Guide*
- [IAM Policies for Amazon EC2](#) in the *Amazon EC2 User Guide for Linux Instances*.

```
"Condition":{  
    "Bool": {  
        "aws:SecureTransport": "true"  
    }  
}  
  
"Condition": {  
    "StringNotEquals": {  
        "s3:x-amz-server-side-encryption": "AES256"  
    }  
}  
  
"Condition": {  
    "StringEquals": {  
        "ec2:ResourceTag/purpose": "test"  
    }  
}
```

AWS Managed Policies for Job Functions

AWS managed policies for job functions are designed to closely align to common job functions in the IT industry. You can use these policies to easily grant the permissions needed to carry out the tasks expected of someone in a specific job function. These policies consolidate permissions for many services into a single policy that's easier to work with than having permissions scattered across many policies.

You can attach these policies for job functions to any group, user, or role.

Use Roles to Combine Services

Some of the policies use IAM service roles to help you take advantage of features found in other AWS services. These policies grant access to `iam:passrole`, which allows a user with the policy to pass a role to an AWS service. This role delegates IAM permissions to the AWS service to carry out actions on your behalf.

You must create the roles according to your needs. For example, the Network Administrator policy allows a user with the policy to pass a role named "flow-logs-vpc" to the Amazon CloudWatch service. CloudWatch uses that role to log and capture IP traffic for VPCs created by the user.

To follow security best practices, the policies for job functions include filters that limit the names of valid roles that can be passed. This helps avoid granting unnecessary permissions. If your users do require the optional service roles, you must create a role that follows the naming convention specified in the policy. You then grant permissions to the role. Once that is done, the user can configure the service to use the role, granting it whatever permissions the role provides.

Keep Up to Date

These policies are all maintained by AWS and are kept up to date to include support for new services and new capabilities as they are added by AWS. These policies cannot be modified by customers. You can make a copy of the policy and then modify the copy, but that copy is not automatically updated as AWS introduces new services and API operations.

Job Functions

Names of policies

- [Administrator \(p. 657\)](#)
- [Billing \(p. 658\)](#)
- [Database Administrator \(p. 658\)](#)
- [Data Scientist \(p. 659\)](#)
- [Developer Power User \(p. 660\)](#)
- [Network Administrator \(p. 660\)](#)
- [Security Auditor \(p. 660\)](#)
- [Support User \(p. 661\)](#)
- [System Administrator \(p. 661\)](#)
- [View-Only User \(p. 661\)](#)

In the following sections, each policy's name is a link to the policy details page in the AWS Management Console. There you can see the policy document and review the permissions it grants.

Administrator

AWS managed policy name: [AdministratorAccess](#)

Use case: This user has full access and can delegate permissions to every service and resource in AWS.

Policy description: This policy grants all actions for all AWS services and for all resources in the account.

Note

Before an IAM user or role can access the AWS Billing and Cost Management console with the permissions in this policy, you must first activate IAM user and role access. To do this, follow the instructions in [Step 1 of the tutorial about delegating access to the billing console \(p. 27\)](#).

Billing

AWS managed policy name: [Billing](#)

Use case: This user needs to view billing information, set up payments, and authorize payments. The user can monitor the costs accumulated for the entire AWS service.

Policy description: This policy grants full permissions for managing billing, costs, payment methods, budgets, and reports.

Note

Before an IAM user or role can access the AWS Billing and Cost Management console with the permissions in this policy, you must first activate IAM user and role access. To do this, follow the instructions in [Step 1 of the tutorial about delegating access to the billing console \(p. 27\)](#).

Database Administrator

AWS managed policy name: [DatabaseAdministrator](#)

Use case: This user sets up, configures, and maintains databases in the AWS Cloud.

Policy description: This policy grants permissions to create, configure, and maintain databases. It includes access to AWS database services, such as Amazon DynamoDB, Amazon Relational Database Service (RDS), and Amazon Redshift. View the policy for the full list of database services that this policy supports.

This job function policy supports the ability to pass roles to AWS services. The policy allows the `iam:PassRole` action for only those roles named in the following table. For more information, see [Creating Roles and Attaching Policies \(Console\) \(p. 662\)](#) later in this topic.

Optional IAM service roles for the Database Administrator job function

Use case	Role name (* is a wildcard)	Service role type to select	Select this AWS managed policy
Allow the user to monitor RDS databases	rds-monitoring-role	Amazon RDS Role for Enhanced Monitoring	AmazonRDSEnhancedMonitoringRole
Allow AWS Lambda to monitor your database and access external databases	rdbms-lambda-access	Amazon EC2	AWSLambdaFullAccess
Allow Lambda to upload files to Amazon S3 and to Amazon Redshift clusters with DynamoDB	lambda_exec_role	AWS Lambda	Create a new managed policy as defined in the AWS Big Data Blog
Allow Lambda functions to act as triggers for your DynamoDB tables	lambda-dynamodb-*	AWS Lambda	AWSLambdaDynamoDBExecutionRole
Allow Lambda functions to access Amazon RDS in a VPC	lambda-vpc-execution-role	Create a role with a trust policy as defined in the AWS Lambda Developer Guide	AWSLambdaVPCAccessExecutionRole
Allow AWS Data Pipeline to access your AWS resources	DataPipelineDefaultRole	Create a role with a trust policy as defined in the	AWSDataPipelineRole

Use case	Role name (* is a wildcard)	Service role type to select	Select this AWS managed policy
		AWS Data Pipeline Developer Guide	
Allow your applications running on Amazon EC2 instances to access your AWS resources	DataPipelineDefaultRole	Create a role with a trust policy as defined in the AWS Data Pipeline Developer Guide	AmazonEC2RoleforDataPipelineRole

Data Scientist

AWS managed policy name: [DataScientist](#)

Use case: This user runs Hadoop jobs and queries. The user also accesses and analyzes information for data analytics and business intelligence.

Policy description: This policy grants permissions to create, manage, and run queries on an Amazon EMR cluster and perform data analytics with tools such as Amazon QuickSight. The policy includes access to additional data scientist services, such as AWS Data Pipeline, Amazon EC2, Amazon Kinesis, Amazon Machine Learning, and Amazon SageMaker. View the policy for the full list of data scientist services that this policy supports.

This job function policy supports the ability to pass roles to AWS services. One statement allows passing any role to Amazon SageMaker. Another statement allows the `iam:PassRole` action for only those roles named in the following table. For more information, see [Creating Roles and Attaching Policies \(Console\) \(p. 662\)](#) later in this topic.

Optional IAM service roles for the Data Scientist job function

Use case	Role name (* is a wildcard)	Service role type to select	AWS managed policy to select
Allow Amazon EC2 instances access to services and resources suitable for clusters	EMR-EC2_DefaultRole	Amazon EMR for EC2	AmazonElasticMapReduceforEC2Role
Allow Amazon EMR access to access the Amazon EC2 service and resources for clusters	EMR_DefaultRole	Amazon EMR	AmazonElasticMapReduceRole
Allow Kinesis Data Analytics to access streaming data sources	kinesis-*	Create a role with a trust policy as defined in the AWS Big Data Blog .	See the AWS Big Data Blog , which outlines four possible options depending on your use case
Allow AWS Data Pipeline to access your AWS resources	DataPipelineDefaultRole	Create a role with a trust policy as defined in the AWS Data Pipeline Developer Guide	AWSDataPipelineRole
Allow your applications running on Amazon EC2 instances to access your AWS resources	DataPipelineDefaultRole	Create a role with a trust policy as defined in the AWS Data Pipeline Developer Guide	AmazonEC2RoleforDataPipelineRole

Use case	Role name (* is a wildcard)	Service role type to select	AWS managed policy to select
		AWS Data Pipeline Developer Guide	

Developer Power User

AWS managed policy name: [PowerUserAccess](#)

Use case: This user performs application development tasks and can create and configure resources and services that support AWS aware application development.

Policy description: The first statement of this policy uses the [NotAction \(p. 607\)](#) element to allow all actions for all AWS services and for all resources except AWS Identity and Access Management and AWS Organizations. The second statement grants IAM permissions to create a service-linked role. This is required by some services that must access resources in another service, such as an Amazon S3 bucket. It also grants Organizations permissions to view information about the user's organization, including the master account email and organization limitations.

Network Administrator

AWS managed policy name: [NetworkAdministrator](#)

Use case: This user is tasked with setting up and maintaining AWS network resources.

Policy description: This policy grants permissions to create and maintain network resources in Auto Scaling, Amazon EC2, AWS Direct Connect, Route 53, Amazon CloudFront, Elastic Load Balancing, AWS Elastic Beanstalk, Amazon SNS, CloudWatch, CloudWatch Logs, Amazon S3, IAM, and Amazon Virtual Private Cloud.

This job function requires the ability to pass roles to AWS services. The policy grants `iam:GetRole` and `iam:PassRole` for only those roles named in the following table. For more information, see [Creating Roles and Attaching Policies \(Console\) \(p. 662\)](#) later in this topic.

Optional IAM service roles for the Network Administrator job function

Use case	Role name (* is a wildcard)	Service role type to select	AWS managed policy to select
Allows Amazon VPC to create and manage logs in CloudWatch Logs on the user's behalf to monitor IP traffic going in and out of your VPC	flow-logs-*	Create a role with a trust policy as defined in the Amazon VPC User Guide	This use case does not have an existing AWS managed policy, but the documentation lists the required permissions. See Amazon VPC User Guide .

Security Auditor

AWS managed policy name: [SecurityAudit](#)

Use case: This user monitors accounts for compliance with security requirements. This user can access logs and events to investigate potential security breaches or potential malicious activity.

Policy description: This policy grants permissions to view configuration data for many AWS services and to review their logs.

Support User

AWS managed policy name: [SupportUser](#)

Use case: This user contacts AWS Support, creates support cases, and views the status of existing cases.

Policy description: This policy grants permissions to create and update AWS Support cases.

System Administrator

AWS managed policy name: [SystemAdministrator](#)

Use case: This user sets up and maintains resources for development operations.

Policy description: This policy grants permissions to create and maintain resources across a large variety of AWS services, including AWS CloudTrail, Amazon CloudWatch, AWS CodeCommit, AWS CodeDeploy, AWS Config, AWS Directory Service, Amazon EC2, AWS Identity and Access Management, AWS Key Management Service, AWS Lambda, Amazon RDS, Route 53, Amazon S3, Amazon SES, Amazon SQS, AWS Trusted Advisor, and Amazon VPC.

This job function requires the ability to pass roles to AWS services. The policy grants `iam:GetRole` and `iam:PassRole` for only those roles named in the following table. For more information, see [Creating Roles and Attaching Policies \(Console\) \(p. 662\)](#) later in this topic.

Optional IAM service roles for the System Administrator job function

Use case	Role name (* is a wildcard)	Service role type to select	AWS managed policy to select
Allow apps running in EC2 instances in an Amazon ECS cluster to access Amazon ECS	ecr-sysadmin-*	Amazon EC2 Role for EC2 Container Service	AmazonEC2ContainerServiceforEC2Role
Allow a user to monitor databases	rds-monitoring-role	Amazon RDS Role for Enhanced Monitoring	AmazonRDSEnhancedMonitoringRole
Allow apps running in EC2 instances to access AWS resources.	ec2-sysadmin-*	Amazon EC2	Sample policy for role that grants access to an S3 bucket as shown in the Amazon EC2 User Guide for Linux Instances ; customize as needed
Allow Lambda to read DynamoDB streams and write to CloudWatch Logs	lambda-sysadmin-*	AWS Lambda	AWSLambdaDynamoDBExecutionRole

View-Only User

AWS managed policy name: [ViewOnlyAccess](#)

Use case: This user can view a list of AWS resources and basic metadata in the account across all services. The user cannot read resource content or metadata that goes beyond the quota and list information for resources.

Policy description: This policy grants `List*`, `Describe*`, `Get*`, `View*`, and `Lookup*` access to resources for most AWS services. To see what actions this policy includes for each service, see [ViewOnlyAccess](#).

Creating Roles and Attaching Policies (Console)

Several of the previously listed policies grant the ability to configure AWS services with roles that enable those services to perform operations on your behalf. The job function policies either specify exact role names that you must use or at least include a prefix that specifies the first part of the name that can be used. To create one of these roles, perform the steps in the following procedure.

To create a role for an AWS service (IAM console)

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane of the IAM console, choose **Roles**, and then choose **Create role**.
3. Choose the **AWS service** role type, and then choose the service that you want to allow to assume this role.
4. Choose the use case for your service. If the specified service has only one use case, it is selected for you. Use cases are defined by the service to include the trust policy that the service requires. Then choose **Next: Permissions**.
5. If possible, select the policy to use for the permissions policy or choose **Create policy** to open a new browser tab and create a new policy from scratch. For more information, see step 4 in the procedure [Creating IAM policies](#) in the *IAM User Guide*. After you create the policy, close that tab and return to your original tab. Select the check box next to the permissions policies that you want the service to have.

Depending on the use case that you selected, the service might allow you to do any of the following:

- Nothing, because the service defines the permissions for the role
 - Allow you to choose from a limited set of permissions
 - Allow you to choose from any permissions
 - Allow you to select no policies at this time, create the policies later, and then attach them to the role
6. (Optional) Set a [permissions boundary](#). This is an advanced feature that is available for service roles, but not service-linked roles.

Expand the **Set permissions boundary** section and choose **Use a permissions boundary to control the maximum role permissions**. IAM includes a list of the AWS managed and customer managed policies in your account. Select the policy to use for the permissions boundary or choose **Create policy** to open a new browser tab and create a new policy from scratch. For more information, see step 4 in the procedure [Creating IAM policies](#) in the *IAM User Guide*. After you create the policy, close that tab and return to your original tab to select the policy to use for the permissions boundary.

7. Choose **Next: Tags**.
8. (Optional) Add metadata to the user by attaching tags as key-value pairs. For more information about using tags in IAM, see [Tagging IAM Entities](#) in the *IAM User Guide*.
9. Choose **Next: Review**.
10. For **Role name**, the degree of role name customization is defined by the service. If the service defines the role's name, this option is not editable. In other cases, the service might define a prefix for the role and allow you to enter an optional suffix. Some services allow you to specify the entire name of your role.

If possible, enter a role name or role name suffix to help you identify the purpose of this role. Role names must be unique within your AWS account. They are not distinguished by case. For example,

you cannot create roles named both **PRODROLE** and **prodrole**. Because various entities might reference the role, you cannot edit the name of the role after it has been created.

11. (Optional) For **Role description**, enter a description for the new role.
12. Review the role and then choose **Create role**.

Example 1: Configuring a User as a Database Administrator (Console)

This example shows the steps required to configure Alice, an IAM user, as a [Database Administrator \(p. 658\)](#). You use the information in first row of the table in that section and allow the user to enable Amazon RDS monitoring. You attach the [DatabaseAdministrator](#) policy to Alice's IAM user so that she can manage the Amazon database services. That policy also enables Alice to pass a role called `rds-monitoring-role` to the Amazon RDS service that allows the service to monitor the RDS databases on her behalf.

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. Choose **Policies** and then type **database** in the search box.
3. Select the check box for the **DatabaseAdministrator** policy, choose **Policy actions**, and then choose **Attach**.
4. In the list of users, select **Alice** and then choose **Attach policy**. Alice now can administer AWS databases. However, to allow Alice to monitor those databases, you must configure the service role.
5. In the navigation pane of the IAM console, choose **Roles**, and then choose **Create role**.
6. Choose the **AWS Service** role type, and then choose **Amazon RDS**.
7. Choose the **Amazon RDS Role for Enhanced Monitoring** use case.
8. Amazon RDS defines the permissions for your role. Choose **Next: Review** to continue.
9. The role name must be one of those specified by the **DatabaseAdministrator** policy that Alice now has. One of those is `rds-monitoring-role`. Type that for the **Role name**.
10. (Optional) For **Role description**, type a description for the new role.
11. After you review the details, choose **Create role**.
12. Alice can now enable **RDS Enhanced Monitoring** in the **Monitoring** section of the Amazon RDS console. For example, she might do this when she creates a DB instance, creates a read replica, or modifies a DB instance. She must type the role name she created (`rds-monitoring-role`) in the **Monitoring Role** box when she sets **Enable Enhanced Monitoring** to **Yes**.

Example 2: Configuring a User as a Network Administrator (Console)

This example shows the steps required to configure Juan, an IAM user, as a [Network Administrator \(p. 660\)](#). It uses the information in the table in that section to allow Juan to monitor IP traffic going to and from a VPC. It also allows Juan to capture that information in the logs in CloudWatch Logs. You attach the [NetworkAdministrator](#) policy to Juan's IAM user so that he can configure AWS network resources. That policy also enables Juan to pass a role whose name begins with `f1ow-logs*` to Amazon EC2 when you create a flow log. In this scenario, unlike Example 1, there isn't a predefined service role type, so you must perform a few steps differently.

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane, choose **Policies** and then type **network** in the search box.

3. Select the check box next to **NetworkAdministrator** policy, choose **Policy actions**, and then choose **Attach**.
4. In the list of users, select the check box next to **Juan** and then choose **Attach policy**. Juan now can administer AWS network resources. However, to enable monitoring of IP traffic in your VPC, you must configure the service role.
5. Because the service role you need to create doesn't have a predefined managed policy, you must first create it. In the navigation pane, choose **Policies**, then choose **Create policy**.
6. Choose the **JSON** tab and copy the text from the following JSON policy document. Paste this text into the **JSON** text box.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Action": [  
                "logs:CreateLogGroup",  
                "logs:CreateLogStream",  
                "logs:PutLogEvents",  
                "logs:DescribeLogGroups",  
                "logs:DescribeLogStreams"  
            ],  
            "Effect": "Allow",  
            "Resource": "*"  
        }  
    ]  
}
```

7. When you are finished, choose **Review policy**. The [Policy Validator \(p. 448\)](#) reports any syntax errors.

Note

You can switch between the **Visual editor** and **JSON** tabs any time. However, if you make changes or choose **Review policy** in the **Visual editor** tab, IAM might restructure your policy to optimize it for the visual editor. For more information, see [Policy Restructuring \(p. 548\)](#).

8. On the **Review** page, type **vpc-flow-logs-policy-for-service-role** for the policy name. Review the policy **Summary** to see the permissions granted by your policy, and then choose **Create policy** to save your work.

The new policy appears in the list of managed policies and is ready to attach.

9. In the navigation pane of the IAM console, choose **Roles**, and then choose **Create role**.
10. Choose the **AWS Service** role type, and then choose **Amazon EC2**.
11. Choose the **Amazon EC2** use case.
12. On the **Attach permissions policies** page, choose the policy you created earlier, **vpc-flow-logs-policy-for-service-role**, and then choose **Next: Review**.
13. The role name must be permitted by the NetworkAdministrator policy that Juan now has. Any name that begins with **flow-logs-** is allowed. For this example, type **flow-logs-for-juan** for the **Role name**.
14. (Optional) For **Role description**, type a description for the new role.
15. After you review the details, choose **Create role**.
16. Now you can configure the trust policy required for this scenario. On the **Roles** page, choose the **flow-logs-for-juan** role (the name, not the check box). On the details page for your new role, choose the **Trust relationships** tab, and then choose **Edit trust relationship**.
17. Change the "Service" line to read as follows, replacing the entry for `ec2.amazonaws.com`:

```
"Service": "vpc-flow-logs.amazonaws.com"
```

18. Juan can now create flow logs for a VPC or subnet in the Amazon EC2 console. When you create the flow log, specify the **flow-logs-for-juan** role. That role has the permissions to create the log and write data to it.

AWS Global Condition Context Keys

When a [principal \(p. 5\)](#) makes a [request \(p. 5\)](#) to AWS, AWS gathers the request information into a [request context \(p. 5\)](#). You can use the Condition element of a JSON policy to compare the request context with values that you specify in your policy. To learn more about the circumstances under which a global key is included in the request context, see the **Availability** information for each global condition key. For information about how to use the Condition element in a JSON policy, see [IAM JSON Policy Elements: Condition \(p. 611\)](#).

Note

If you use condition keys that are available only in some circumstances, you can use the [IfExists \(p. 620\)](#) versions of the condition operators. If the condition keys are missing from a request context, the policy can fail the evaluation. For example, use the following condition block with `...IfExists` operators to match when a request comes from a specific IP range or from a specific VPC. If either or both keys are not included in the request context, the condition still returns `true`. The values are only checked if the specified key is included in the request context.

```
"Condition": {  
    "IpAddressIfExists": {"aws:SourceIp" : ["xxx"] },  
    "StringEqualsIfExists" : {"aws:SourceVpc" : ["yyy"]} } }
```

Global condition keys are condition keys with an `aws:` prefix. AWS services can provide service-specific keys that include the service prefix. For example, IAM condition keys include the `iam:` prefix. For more information, see [Actions, Resources, and Condition Keys for AWS Services \(p. 686\)](#) and choose the service whose keys you want to view.

aws:CurrentTime

Works with [date operators \(p. 616\)](#).

Use this key to compare the date and time of the request with the date and time that you specify in the policy.

- **Availability** – This key is always included in the request context.

aws:EpochTime

Works with [date operators \(p. 616\)](#) or [numeric operators \(p. 616\)](#).

Use this key to compare the date and time of the request in epoch or Unix time with the value that you specify in the policy. This key also accepts the number of seconds since January 1, 1970.

- **Availability** – This key is always included in the request context.

aws:MultiFactorAuthAge

Works with [numeric operators \(p. 616\)](#).

Use this key to compare the number of seconds since the requesting principal was authorized using MFA with the number that you specify in the policy. For more information about MFA, see [Using Multi-Factor Authentication \(MFA\) in AWS \(p. 119\)](#).

- **Availability** – This key is included in the request context only if the principal was authenticated using MFA. If MFA was not used, this key is not present.

aws:MultiFactorAuthPresent

Works with [Boolean operators \(p. 617\)](#).

Use this key to check whether multi-factor authentication (MFA) was used to validate the temporary security credentials that made the request.

- **Availability** – This key is included in the request context only when the principal uses temporary credentials to make the request. The key is not present in AWS CLI, AWS API, or AWS SDK requests that are made using long-term credentials.

Temporary credentials are used to authenticate IAM roles, federated users, IAM users with temporary tokens from `sts:GetSessionToken`, and users of the AWS Management Console. IAM users in the AWS Management Console unknowingly use temporary credentials. Users sign into the console using their user name and password, which are long-term credentials. However, in the background, the console generates temporary credentials on behalf of the user. To learn which services support using temporary credentials, see [AWS Services That Work with IAM \(p. 586\)](#).

The `aws:MultiFactorAuthPresent` key is not present when an API or CLI command is called with long-term credentials, such as user access key pairs. Therefore we recommend that when you check for this key that you use the [`...IfExists \(p. 620\)`](#) versions of the condition operators.

It is important to understand that the following Condition element is **not** a reliable way to check whether a request is authenticated using MFA.

```
##### WARNING: NOT RECOMMENDED #####
"Effect" : "Deny",
"Condition" : { "Bool" : { "aws:MultiFactorAuthPresent" : false } }
```

This combination of the Deny effect, `Bool` element, and `false` value denies requests that can be authenticated using MFA, but were not. This applies only to temporary credentials that support using MFA. This statement does not deny access to requests that are made using long-term credentials, or to requests that are authenticated using MFA. Use this example with caution because its logic is complicated and it does not test whether MFA-authentication was actually used.

Also do not use the combination of the Deny effect, `Null` element, and `true` because it behaves the same way and the logic is even more complicated.

Recommended Combination

We recommend that you use the [`BoolIfExists \(p. 620\)`](#) operator to check whether a request is authenticated using MFA.

```
"Effect" : "Deny",
"Condition" : { "BoolIfExists" : { "aws:MultiFactorAuthPresent" : false } }
```

This combination of Deny, `BoolIfExists`, and `false` denies requests that are not authenticated using MFA. Specifically, it denies requests from temporary credentials that do not include MFA. It also denies requests that are made using long-term credentials, such as AWS CLI or AWS API operations made using

access keys. The `*IfExists` operator checks for the presence of the `aws:MultiFactorAuthPresent` key and whether or not it could be present, as indicated by its existence. Use this when you want to deny any request that is not authenticated using MFA. This is more secure, but can break any code or scripts that use access keys to access the AWS CLI or AWS API.

Alternative Combinations

You can also use the [BoolIfExists \(p. 620\)](#) operator to allow MFA-authenticated requests and AWS CLI or AWS API requests that are made using long-term credentials.

```
"Effect" : "Allow",
"Condition" : { "BoolIfExists" : { "aws:MultiFactorAuthPresent" : true } }
```

This condition matches either if the key exists and is present **or** if the key does not exist. This combination of `Allow`, `BoolIfExists`, and `true` allows requests that are authenticated using MFA, or requests that cannot be authenticated using MFA. This means that AWS CLI, AWS API, and AWS SDK operations are allowed when the requester uses their long-term access keys. This combination does not allow requests from temporary credentials that could, but do not include MFA.

When you create a policy using the IAM console visual editor and choose **MFA required**, this combination is applied. This setting requires MFA for console access, but allows programmatic access with no MFA.

Alternatively, you can use the `Bool` operator to allow programmatic and console requests only when authenticated using MFA.

```
"Effect" : "Allow",
"Condition" : { "Bool" : { "aws:MultiFactorAuthPresent" : true } }
```

This combination of the `Allow`, `Bool`, and `true` allows only MFA-authenticated requests. This applies only to temporary credentials that support using MFA. This statement does not allow access to requests that were made using long-term access keys, or to requests made using temporary credentials without MFA.

Do not use a policy construct similar to the following to check whether the MFA key is present:

```
##### WARNING: USE WITH CAUTION #####
"Effect" : "Allow",
"Condition" : { "Null" : { "aws:MultiFactorAuthPresent" : false } }
```

This combination of the `Allow` effect, `Null` element, and `false` value allows only requests that can be authenticated using MFA, regardless of whether the request is actually authenticated. This allows all requests that are made using temporary credentials, and denies access for long-term credentials. Use this example with caution because it does not test whether MFA-authentication was actually used.

aws:PrincipalAccount

Works with [string operators \(p. 614\)](#).

Use this key to compare the account to which the requesting principal belongs with the account identifier that you specify in the policy.

- **Availability** – This key is always included in the request context.

aws:PrincipalArn

Works with [ARN operators \(p. 619\)](#).

Use this key to compare the [Amazon Resource Name \(ARN\)](#) of the principal that made the request with the ARN that you specify in the policy. For IAM roles, the request context returns the ARN of the role, not the ARN of the user that assumed the role. To learn which types of principals you can specify in this condition key, see [Specifying a Principal \(p. 601\)](#).

- **Availability** – This key is always included in the request context.

aws:PrincipalOrgID

Works with [string operators \(p. 614\)](#).

Use this key to compare the identifier of the organization in AWS Organizations to which the requesting principal belongs with the identifier specified in the policy.

- **Availability** – This key is included in the request context only if the principal is a member of an organization.

This global key provides an alternative to listing all the account IDs for all AWS accounts in an organization. You can use this condition key to simplify specifying the `Principal` element in a [resource-based policy \(p. 379\)](#). You can specify the [organization ID](#) in the condition element. When you add and remove accounts, policies that include the `aws:PrincipalOrgID` key automatically include the correct accounts and don't require manual updating.

For example, the following Amazon S3 bucket policy allows members of any account in the `o-xxxxxxxxxx` organization to add an object into the `policy-ninja-dev` bucket.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "AllowPutObject",  
            "Effect": "Allow",  
            "Principal": "*",  
            "Action": "s3:PutObject",  
            "Resource": "arn:aws:s3:::policy-ninja-dev/*",  
            "Condition": {"StringEquals":  
                {"aws:PrincipalOrgID": ["o-xxxxxxxxxx"]}}  
        }  
    ]  
}
```

Note

This global condition also applies to the master account of an AWS organization.

For more information about AWS Organizations, see [What Is AWS Organizations?](#) in the [AWS Organizations User Guide](#).

aws:PrincipalOrgPaths

Works with [string operators \(p. 614\)](#).

Use this key to compare the AWS Organizations path for the principal who is making the request to the path in the policy. That principal can be an IAM user, IAM role, federated user, or AWS account root user. In a policy, this condition key ensures that the requester is an account member within the specified organization root or organizational units (OUs) in AWS Organizations. An AWS Organizations path is a text representation of the structure of an Organizations entity. For more information about using and understanding paths, see [Understand the AWS Organizations Entity Path \(p. 482\)](#).

- **Availability** – This key is included in the request context only if the principal is a member of an organization.

Note

Organization IDs are globally unique but OU IDs and root IDs are unique only within an organization. This means that no two organizations share the same organization ID. However, another organization might have an OU or root with the same ID as yours. We recommend that you always include the organization ID when you specify an OU or root.

For example, the following condition returns `true` for principals in accounts that are attached directly to the `ou-jk10-awsdddd` OU, but not in its child OUs.

```
"Condition" : { "ForAnyValue:StringEquals" : {
    "aws:PrincipalOrgPaths": ["o-a1b2c3d4e5/r-f6g7h8i9j0example/ou-ghi0-awscccccc/ou-jk10-
awsdddddd/*"]
}}
```

The following condition returns `true` for principals in an account that is attached directly to the OU or any of its child OUs. When you include a wildcard, you must use the `StringLike` condition operator.

```
"Condition" : { "ForAnyValue:StringLike" : {
    "aws:PrincipalOrgPaths": ["o-a1b2c3d4e5/r-f6g7h8i9j0example/ou-ghi0-awscccccc/ou-jk10-
awsdddddd*"]
}}
```

The following condition returns `true` for principals in an account that is attached directly to the OU or any of its child OUs.

```
"Condition" : { "ForAnyValue:StringLike" : {
    "aws:PrincipalOrgPaths": ["o-a1b2c3d4e5/r-f6g7h8i9j0example/ou-ghi0-awscccccc/ou-jk10-
awsdddddd/*"]
}}
```

The following condition allows access for every principal in the `o-a1b2c3d4e5` organization, regardless of their parent OU.

```
"Condition" : { "ForAnyValue:StringLike" : {
    "aws:PrincipalOrgPaths": ["o-a1b2c3d4e5/*"]
}}
```

`aws:PrincipalOrgPaths` is a multivalued condition key. Multivalued keys include one or more values in a list format. The result is a logical OR. When you use multiple values with the `ForAnyValue` condition operator, the principal's path must match one of the paths listed in the policy. For policies that include multiple values for a single key, you must enclose the conditions within brackets like an array (`"Key":["Value1", "Value2"]`). You should also include these brackets when there is a single value. For more information about multivalued condition keys, see [Creating a Condition with Multiple Keys or Values \(p. 621\)](#).

```
"Condition": {
    "ForAnyValue:StringLike": {
        "aws:PrincipalOrgPaths": [
            "o-a1b2c3d4e5/r-f6g7h8i9j0example/ou-def0-awsbbbbbb/*",
            "o-a1b2c3d4e5/r-f6g7h8i9j0example/ou-jkl0-awsdddddd/*"
        ]
    }
}
```

aws:PrincipalTag

Works with string operators (p. 614).

Use this key to compare the tag attached to the principal making the request with the tag that you specify in the policy. If the principal has more than one tag attached, the request context includes one aws:PrincipalTag key for each attached tag key.

- **Availability** – This key is included in the request context if the principal is using an IAM user with attached tags. It is included for a principal using an IAM role with attached tags or session tags (p. 299).

You can add custom attributes to a user or role in the form of a key-value pair. For more information about IAM tags, see Tagging IAM Users and Roles (p. 295). You can use aws:PrincipalTag to control access (p. 390) for AWS principals.

This example shows how you might create a policy that allows users with the `tagManager=true` tag to manage IAM users, groups, or roles. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": "iam:*",  
            "Resource": "*",  
            "Condition": {"StringEquals": {"aws:PrincipalTag/tagManager": "true"}}  
        }  
    ]  
}
```

aws:PrincipalType

Works with string operators (p. 614).

Use this key to compare the type of principal making the request with the principal type that you specify in the policy. For details about how the information appears in the request context for different principals, see Specifying a Principal (p. 601).

- **Availability** – This key is always included in the request context.

aws:Referer

Works with string operators (p. 614).

Use this key to compare who referred the request in the client browser with the referer that you specify in the policy. The aws:referer request context value is provided by the caller in an HTTP header.

- **Availability** – This key is included in the request context only if the request was invoked using a URL in the browser.

For example, you can call Amazon S3 API operations directly using a web browser. This means that you can view S3 objects, such as images and documents, directly through a web browser. The aws:referer condition allows you to restrict access to specific values in the HTTP or HTTPS request based on the value of the referrer header.

Warning

This key should be used carefully. It is dangerous to include a publicly known referer header value. Unauthorized parties can use modified or custom browsers to provide any aws:referer value that they choose. As a result, aws:referer should not be used to prevent unauthorized parties from making direct AWS requests. It is offered only to allow customers to protect their digital content, such as content stored in Amazon S3, from being referenced on unauthorized third-party sites.

aws:RequestedRegion

Works with string operators (p. 614).

Use this key to compare the AWS Region that was called in the request with the Region that you specify in the policy. You can use this global condition key to control which Regions can be requested. To view the AWS Regions for each service, see [AWS Regions and Endpoints](#) in the *Amazon Web Services General Reference*.

- **Availability** – This key is always included in the request context.

Some global services, such as IAM, have a single endpoint. Because this endpoint is physically located in the US East (N. Virginia) Region, IAM calls are always made to the us-east-1 Region. For example, if you create a policy that denies access to all services if the requested Region is not us-west-2, then IAM calls always fail. To view an example of how to work around this, see [NotAction with Deny \(p. 607\)](#).

Note

The `aws:RequestedRegion` condition key allows you to control which endpoint of a service is invoked but does not control the impact of the operation. Some services have cross-Region impacts. For example, Amazon S3 has API operations that control cross-Region replication. You can invoke `s3:PutBucketReplication` in one Region (which is affected by the `aws:RequestedRegion` condition key), but other Regions are affected based on the replications configuration settings.

You can use this context key to limit access to AWS services within a given set of Regions. For example, the following policy allows a user to view all of the Amazon EC2 instances in the AWS Management Console. However it only allows them to make changes to instances in Ireland (eu-west-1), London (eu-west-2), or Paris (eu-west-3).

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "InstanceConsoleReadOnly",
            "Effect": "Allow",
            "Action": [
                "ec2:Describe*",
                "ec2:Export*",
                "ec2:Get*",
                "ec2:Search*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "InstanceWriteRegionRestricted",
            "Effect": "Allow",
            "Action": [
                "ec2:Associate*",
                "ec2:Import*",
                "ec2:Modify*",
                "ec2:Monitor*",
                "ec2:Reset*"
            ]
        }
    ]
}
```

```
        "ec2:Run*",
        "ec2:Start*",
        "ec2:Stop*",
        "ec2:Terminate*"
    ],
    "Resource": "*",
    "Condition": {
        "StringEquals": {
            "aws:RequestedRegion": [
                "eu-west-1",
                "eu-west-2",
                "eu-west-3"
            ]
        }
    }
}
```

aws:RequestTag/*tag-key*

Works with [string operators \(p. 614\)](#).

Use this key to compare the tag key-value pair that was passed in the request with the tag pair that you specify in the policy. For example, you could check whether the request includes the tag key "Dept" and that it has the value "Accounting". For more information, see [Controlling Access During AWS Requests \(p. 393\)](#).

- **Availability** – This key is included in the request context when tags are passed in the request. When multiple tags are passed in the request, there is one context key for each tag key-value pair.

This context key is formatted "aws:RequestTag/*tag-key*":"*tag-value*" where *tag-key* and *tag-value* are a tag key and value pair.

Because you can include multiple tag key-value pairs in a request, the request content could be a multivalued request. In this case, you should consider using the `ForAllValues` or `ForAnyValue` set operators. For more information, see [Using Multiple Keys and Values \(p. 622\)](#).

aws:ResourceTag/*tag-key*

Works with [string operators \(p. 614\)](#).

Use this key to compare the tag key-value pair that you specify in the policy with the key-value pair that is attached to the resource. For example, you could require that access to a resource is allowed only if the resource has the attached tag key "Dept" with the value "Marketing". For more information, see [Controlling Access to AWS Resources \(p. 392\)](#).

- **Availability** – This key is included in the request context when the requested resource already has attached tags. This key is returned only for resources that [support authorization based on tags \(p. 586\)](#). There is one context key for each tag key-value pair.

This context key is formatted "aws:ResourceTag/*tag-key*":"*tag-value*" where *tag-key* and *tag-value* are a tag key and value pair.

aws:SecureTransport

Works with [Boolean operators \(p. 617\)](#).

Use this key to check whether the request was sent using SSL. The request context returns `true` or `false`. In a policy, you can allow specific actions only if the request is sent using SSL.

- **Availability** – This key is always included in the request context.

aws:SourceAccount

Works with [string operators \(p. 614\)](#).

Use this key to compare the account ID of the resource making a service-to-service request with the account ID that you specify in the policy.

- **Availability** – This key is included in the request context only if accessing a resource triggers an AWS service to call another service on behalf of the resource owner. The calling service must pass the resource ARN of the source to the called service. This ARN includes the source account ID.

You can use this condition key to check that Amazon S3 is not being used as a [confused deputy \(p. 234\)](#). For example, when an Amazon S3 bucket update triggers an Amazon SNS topic post, the Amazon S3 service invokes the `sns:Publish` API operation. The bucket is considered the source of the SNS request and the value of the key is the account ID from the bucket's ARN.

aws:SourceArn

Works with [ARN operators \(p. 619\)](#).

Use this key to compare the [Amazon Resource Name \(ARN\) \(p. 576\)](#) of the resource making a service-to-service request with the ARN that you specify in the policy.

This key does not work with the ARN of the principal making the request. Instead, use [aws:PrincipalArn \(p. 667\)](#). The source's ARN includes the account ID, so it is not necessary to use `aws:SourceAccount` with `aws:SourceArn`.

- **Availability** – This key is included in the request context only if accessing a resource triggers an AWS service to call another service on behalf of the resource owner. The calling service must pass the ARN of the original resource to the called service.

You can use this condition key to check that Amazon S3 is not being used as a [confused deputy \(p. 234\)](#). For example, when an Amazon S3 bucket update triggers an Amazon SNS topic post, the Amazon S3 service invokes the `sns:Publish` API operation. The bucket is considered the source of the SNS request and the value of the key is the bucket's ARN.

aws:SourceIp

Works with [IP address operators \(p. 618\)](#).

Use this key to compare the requester's IP address with the IP address that you specify in the policy. To learn about the condition operators that you can use with this key, see [IP Address Condition Operators \(p. 618\)](#).

- **Availability** – This key is included in the request context, except when the requester uses a VPC endpoint to make the request.

The `aws:SourceIp` condition key can be used in a policy to allow principals to make requests only from within a specified IP range. However, this policy would deny access to an AWS service that makes calls on your behalf. For example, assume that AWS CloudFormation uses a [service role \(p. 176\)](#) to call Amazon

EC2 to stop an instance. In this case, the request is denied because the target service (Amazon EC2) sees the IP address of the calling service (AWS CloudFormation). The request context does not include the IP address of the originating user. There is no way to pass the originating IP address through a calling service to the target service for evaluation in a JSON policy.

If the request comes from a host that uses an Amazon VPC endpoint, then the `aws:SourceIp` key is not available. You should instead use a VPC-specific key such as [aws:VpcSourceIp \(p. 673\)](#). For more information about using VPC endpoints, see [VPC Endpoints - Controlling the Use of Endpoints](#) in the [Amazon VPC User Guide](#).

aws:SourceVpc

Works with [string operators \(p. 614\)](#).

Use this key to check whether the request comes from the VPC that you specify in the policy. In a policy, you can use this key to allow access to only a specific VPC. For more information, see [Restricting Access to a Specific VPC](#) in the [Amazon Simple Storage Service Developer Guide](#).

- **Availability** – This key is included in the request context only if the requester uses a VPC endpoint to make the request.

aws:SourceVpce

Works with [string operators \(p. 614\)](#).

Use this key to compare the VPC endpoint identifier of the request with the endpoint ID that you specify in the policy. In a policy, you can use this key to restrict access to a specific VPC endpoint. For more information, see [Restricting Access to a Specific VPC Endpoint](#) in the [Amazon Simple Storage Service Developer Guide](#).

- **Availability** – This key is included in the request context only if the requester uses a VPC endpoint to make the request.

aws:TagKeys

Works with [string operators \(p. 614\)](#).

Use this key to compare the tag keys in a request with the keys that you specify in the policy. As a best practice when you use policies to control access using tags, use the `aws:TagKeys` condition key to define what tag keys are allowed. For example policies and more information, see [the section called "Controlling Access Based on Tag Keys" \(p. 394\)](#).

- **Availability** – This key is included in the request context only if the operation supports attaching tags to resources.

This context key is formatted "`aws:TagKeys`": "`tag-key`" where `tag-key` is a list of tag keys without values (for example, `["Dept", "Cost-Center"]`).

Because you can include multiple tag key-value pairs in a request, the request content could be a multivalued request. In this case, you should consider using the `ForAllValues` or `ForAnyValue` set operators. For more information, see [Using Multiple Keys and Values \(p. 622\)](#).

Some services support tagging with resource operations, such as creating, modifying, or deleting a resource. To allow tagging and operations as a single call, you must create a policy that includes both the tagging action and the resource-modifying action. You can then use the `aws:TagKeys`

condition key to enforce using specific tag keys in the request. For example, to limit tags when someone creates an Amazon EC2 snapshot, you must include the `ec2:CreateSnapshot` creation action **and** the `ec2:CreateTags` tagging action in the policy. To view a policy for this scenario that uses `aws:TagKeys`, see [Creating a Snapshot with Tags](#) in the *Amazon EC2 User Guide for Linux Instances*.

aws:TokenIssueTime

Works with [date operators \(p. 616\)](#).

Use this key to compare the date and time that temporary security credentials were issued with the date and time that you specify in the policy.

- **Availability** – This key is included in the request context only when the principal uses temporary credentials to make the request. They key is not present in AWS CLI, AWS API, or AWS SDK requests that are made using access keys.

To learn which services support using temporary credentials, see [AWS Services That Work with IAM \(p. 586\)](#).

aws:UserAgent

Works with [string operators \(p. 614\)](#).

Use this key to compare the requester's client application with the application that you specify in the policy.

- **Availability** – This key is always included in the request context.

Warning

This key should be used carefully. Since the `aws:UserAgent` value is provided by the caller in an HTTP header, unauthorized parties can use modified or custom browsers to provide any `aws:UserAgent` value that they choose. As a result, `aws:UserAgent` should not be used to prevent unauthorized parties from making direct AWS requests. You can use it to allow only specific client applications, and only after testing your policy.

aws:userid

Works with [string operators \(p. 614\)](#).

Use this key to compare the requester's principal identifier with the ID that you specify in the policy. For IAM users, the request context value is the user ID. For IAM roles, this value format can vary. For details about how the information appears for different principals, see [Specifying a Principal \(p. 601\)](#).

- **Availability** – This key is included in the request context for all signed requests. Anonymous requests do not include this key.

aws:username

Works with [string operators \(p. 614\)](#).

Use this key to compare the requester's user name with the user name that you specify in the policy. For details about how the information appears for different principals, see [Specifying a Principal \(p. 601\)](#).

- **Availability** – This key is always included in the request context for IAM users. Anonymous requests and requests that are made using the AWS account root user or IAM roles do not include this key.

aws:VpcSourceIp

Works with IP address operators (p. 618).

Use this key to compare the IP address from which a request was made with the IP address that you specify in the policy. In a policy, the key matches only if the request originates from the specified IP address and it goes through a VPC endpoint.

- **Availability** – This key is included in the request context only if the request is made using a VPC endpoint.

For more information, see [Controlling Access to Services with VPC Endpoints](#) in the *Amazon VPC User Guide*.

IAM and AWS STS Condition Context Keys

You can use the `Condition` element in a JSON policy to test the value of keys that are included in the request context of all AWS requests. These keys provide information about the request itself or the resources that the request references. You can check that keys have specified values before allowing the action requested by the user. This gives you granular control over when your JSON policy statements match or don't match an incoming request. For information about how to use the `Condition` element in a JSON policy, see [IAM JSON Policy Elements: Condition \(p. 611\)](#).

This topic describes the keys defined and provided by the IAM service (with an `iam:` prefix) and the AWS Security Token Service (AWS STS) service (with an `sts:` prefix). Several other AWS services also provide service-specific keys that are relevant to the actions and resources defined by that service. For more information, see [Actions, Resources, and Condition Keys for AWS Services \(p. 686\)](#). The documentation for a service that supports condition keys often has additional information. For example, for information about keys that you can use in policies for Amazon S3 resources, see [Amazon S3 Policy Keys](#) in the *Amazon Simple Storage Service Developer Guide*.

Topics

- [Available Keys for IAM \(p. 676\)](#)
- [Available Keys for AWS Web Identity Federation \(p. 678\)](#)
- [Available Keys for SAML-Based AWS STS Federation \(p. 681\)](#)
- [Available Keys for AWS STS \(p. 685\)](#)

Available Keys for IAM

You can use the following condition keys in policies that control access to IAM resources:

iam:AssociatedResourceArn

Works with [ARN operators \(p. 619\)](#).

Specifies the ARN of the resource to which this role will be associated at the destination service. The resource usually belongs to the service to which the principal is passing the role. Sometimes, the resource might belong to a third service. For example, you might pass a role to Amazon EC2 Auto Scaling that they use on an Amazon EC2 instance. In this case, the condition would match the ARN of the Amazon EC2 instance.

This condition key applies to only the [PassRole \(p. 258\)](#) action in a policy. It can't be used to limit any other action.

Use this condition key in a policy to allow an entity to pass a role, but only if that role is associated with the specified resource. You can use wildcards (*) to allow operations performed on a specific

type of resource without restricting the region or resource ID. For example, you can allow an IAM user or role to pass any role to the Amazon EC2 service to be used with instances in the region "us-east-1" or "us-west-1". The IAM user or role would not be allowed to pass roles to other services, and it doesn't allow Amazon EC2 to use the role with instances in other Regions.

```
{  
    "Effect": "Allow",  
    "Action": "iam:PassRole",  
    "Resource": "*",  
    "Condition": {  
        "StringEquals": {"iam:PassedToService": "ec2.amazonaws.com"},  
        "StringLike": {  
            "iam:AssociatedResourceARN": [  
                "arn:aws:ec2:us-east-1:111122223333:instance/*",  
                "arn:aws:ec2:us-west-1:111122223333:instance/*"  
            ]  
        }  
    }  
}
```

Note

AWS services that support [iam:PassedToService \(p. 677\)](#) also support this condition key.

iam:AWSServiceName

Works with [string operators \(p. 614\)](#).

Specifies the AWS service to which this role is attached.

iam:OrganizationsPolicyId

Works with [string operators \(p. 614\)](#).

Checks that the policy with the specified AWS Organizations ID matches the policy used in the request. To view an example IAM policy that uses this condition key, see [IAM: View Service Last Accessed Data for an Organizations Policy \(p. 432\)](#).

iam:PassedToService

Works with [string operators \(p. 614\)](#).

Specifies the service principal of the service to which a role can be passed. This condition key applies to only the [PassRole \(p. 258\)](#) action in a policy. It can't be used to limit any other action.

When you use this condition key in a policy, specify the service using a service principal. A service principal is the name of a service that can be specified in the `Principal` element of a policy. This is the usual format: `SERVICE_NAME_URL.amazonaws.com`.

You can use `iam:PassedToService` to restrict your users so that they can pass roles only to specific services. For example, a user might create a [service role \(p. 176\)](#) that trusts CloudWatch to write log data to an Amazon S3 bucket on their behalf. Then the user must attach a permissions policy and a trust policy to the new service role. In this case, the trust policy must specify `cloudwatch.amazonaws.com` in the `Principal` element. To view a policy that allows the user to pass the role to CloudWatch, see [IAM: Pass an IAM Role to a Specific AWS Service \(p. 426\)](#).

By using this condition key, you can ensure that users create service roles only for the services that you specify. For example, if a user with the preceding policy attempts to create a service role for Amazon EC2, the operation will fail. The failure occurs because the user does not have permission to pass the role to Amazon EC2.

Note

Some services, such as AWS CodeBuild and AWS CodeCommit do not support this condition key.

iam:PermissionsBoundary

Works with [string operators \(p. 614\)](#).

Checks that the specified policy is attached as permissions boundary on the IAM principal resource. For more information, see [Permissions Boundaries for IAM Entities \(p. 370\)](#)

iam:PolicyARN

Works with [ARN operators \(p. 619\)](#).

Checks the Amazon Resource Name (ARN) of a managed policy in requests that involve a managed policy. For more information, see [Controlling Access to Policies \(p. 385\)](#).

iam:ResourceTag/*key-name*

Works with [string operators \(p. 614\)](#).

Checks that the tag attached to the identity resource (user or role) matches the specified key name and value.

Note

IAM does not support using the [aws:ResourceTag \(p. 672\)](#) global condition key. AWS STS supports both the IAM key and the global key.

You can add custom attributes to a user or role in the form of a key-value pair. For more information about IAM tags, see [the section called "Tagging Users and Roles" \(p. 295\)](#). You can use [iam:ResourceTag](#) to [control access \(p. 390\)](#) to IAM users and roles. However, because IAM does not support tags for groups, you cannot use tags to control access to groups.

This example shows how you might create a policy that allows deleting users with the **status=terminated** tag. To use this policy, replace the *italicized placeholder text* in the example policy with your own information.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {"Effect": "Allow",  
         "Action": "iam:DeleteUser",  
         "Resource": "*",
         "Condition": {"StringLike": {"iam:ResourceTag/status": "terminated"}}  
    ]  
}
```

Available Keys for AWS Web Identity Federation

You can use web identity federation to give temporary security credentials to users who have been authenticated through an identity provider (IdP). Examples of such providers include Login with Amazon, Amazon Cognito, Google, or Facebook. In that case, additional condition keys are available when the temporary security credentials are used to make a request. You can use these keys to write policies that limit the access of federated users to resources that are associated with a specific provider, app, or user. These keys are typically used in the trust policy for a role.

aws:FederatedProvider

Works with [string operators \(p. 614\)](#).

The **FederatedProvider** key identifies which of the IdPs was used to authenticate the user. For example, if the user was authenticated through Amazon Cognito, the key would contain `cognito-identity.amazonaws.com`. Similarly, if the user was authenticated through Login with Amazon,

the key would contain the value `www.amazon.com`. You might use the key in a resource policy like the following, which uses the `aws:FederatedProvider` key as a policy variable in the ARN of a resource. The policy allows any user who has been authenticated using an IdP to get objects out of a folder in an Amazon S3 bucket. However, the bucket must be specific to the provider that the user authenticates with.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Principal": "*",
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::BUCKET-NAME/${aws:FederatedProvider}/*"
        }
    ]
}
```

amr

Works with [string operators \(p. 614\)](#).

Example: `cognito-identity.amazonaws.com:com:amr`

If you are using Amazon Cognito for web identity federation, the `cognito-identity.amazonaws.com:amr` key (Authentication Methods Reference) includes login information about the user. The key is multivalued, meaning that you test it in a policy using [condition set operators \(p. 621\)](#). The key can contain the following values:

- If the user is unauthenticated, the key contains only `unauthenticated`.
- If the user is authenticated, the key contains the value `authenticated` and the name of the login provider used in the call (`graph.facebook.com`, `accounts.google.com`, or `www.amazon.com`).

As an example, the following condition in the trust policy for an Amazon Cognito role tests whether the user is unauthenticated:

```
"Condition": {  
    "StringEquals":  
        { "cognito-identity.amazonaws.com:aud": "us-east-2:identity-pool-id" },  
    "ForAnyValue:StringLike":  
        { "cognito-identity.amazonaws.com:amr": "unauthenticated" }  
}
```

aud

Works with [string operators \(p. 614\)](#).

Use the `aud` condition key to verify that the Google client ID or Amazon Cognito identity pool ID matches the one that you specify in the policy. You can use the `aud` key with the `sub` key for the same identity provider.

Examples:

- `accounts.google.com:aud`
- `cognito-identity.amazonaws.com:aud`

The `accounts.google.com:aud` condition key matches the following Google ID Token fields.

- `aud` for OAuth 2.0 Google client IDs of your application, when the `azp` field is not set. When the `azp` field is set, the `aud` field matches the [accounts.google.com:oauth \(p. 681\)](#) condition key.

- `azp` when the `azp` field is set. This can happen for hybrid apps where a web application and Android app have a different OAuth 2.0 Google client ID but share the same Google APIs project.

For more information about Google `aud` and `azp` fields, see the [Google Identity Platform OpenID Connect Guide](#).

When you write a policy using the `accounts.google.com:aud` condition key, you must know whether the app is a hybrid app that sets the `azp` field.

azp Field Not Set

The following example policy works for non-hybrid apps that do not set the `azp` field. In this case the Google ID Token `aud` field value matches both the `accounts.google.com:aud` and the `accounts.google.com:oaud` condition key values.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Principal": {"Federated": "accounts.google.com"},  
            "Action": "sts:AssumeRoleWithWebIdentity",  
            "Condition": {  
                "StringEquals": {  
                    "accounts.google.com:aud": "aud-value",  
                    "accounts.google.com:oaud": "aud-value",  
                    "accounts.google.com:sub": "sub-value"  
                }  
            }  
        }  
    ]  
}
```

azp Field Set

The following example policy works for hybrid apps that do set the `azp` field. In this case the Google ID Token `aud` field value matches only the `accounts.google.com:oaud` condition key value. The `azp` field value matches the `accounts.google.com:aud` condition key value.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Principal": {"Federated": "accounts.google.com"},  
            "Action": "sts:AssumeRoleWithWebIdentity",  
            "Condition": {  
                "StringEquals": {  
                    "accounts.google.com:aud": "azp-value",  
                    "accounts.google.com:oaud": "aud-value",  
                    "accounts.google.com:sub": "sub-value"  
                }  
            }  
        }  
    ]  
}
```

id

Works with [string operators](#) (p. 614).

Examples:

- graph.facebook.com:app_id
- graph.facebook.com:id
- www.amazon.com:app_id
- www.amazon.com:user_id

Use these keys to verify that the application (or site) ID or user ID matches the one that you specify in the policy. This works for Facebook or Login with Amazon. You can use the app_id key with the id key for the same identity provider.

oaud

Works with [string operators \(p. 614\)](#).

Example: accounts.google.com:oaud

If you use Google for web identity federation, this key specifies the Google audience (aud) that this ID token is intended for. It must be one of the OAuth 2.0 client IDs of your application.

sub

Works with [string operators \(p. 614\)](#).

Examples:

- accounts.google.com:sub
- cognito-identity.amazonaws.com:sub

Use these keys to verify that the user ID matches the one that you specify in the policy. You can use the sub key with the aud key for the same identity provider.

More Information About Web Identity Federation

For more information about web identity federation, see the following:

- [Amazon Cognito Overview](#) in the *AWS Mobile SDK for Android Developer Guide* guide
- [Amazon Cognito Overview](#) in the *AWS Mobile SDK for iOS Developer Guide* guide
- [About Web Identity Federation \(p. 185\)](#)

Available Keys for SAML-Based AWS STS Federation

If you are working with [SAML-based federation](#) using AWS Security Token Service (AWS STS), you can include additional condition keys in the policy.

SAML Role Trust Policies

In the trust policy of a role, you can include the following keys, which help you establish whether the caller is allowed to assume the role. Except for saml:doc, all the values are derived from the SAML assertion. All items in the list are available in the IAM console visual editor when you create or edit a policy with conditions. Items marked with  can have a value that is a list of the specified type.

saml:aud

Works with [string operators \(p. 614\)](#).

An endpoint URL to which SAML assertions are presented. The value for this key comes from the SAML Recipient field in the assertion, *not* the Audience field.

saml:commonName[]

Works with [string operators \(p. 614\)](#).

This is a `commonName` attribute.

saml:cn[]

Works with [string operators \(p. 614\)](#).

This is an `eduOrg` attribute.

saml:doc

Works with [string operators \(p. 614\)](#).

This represents the principal that was used to assume the role. The format is `account-ID/provider-friendly-name`, such as `123456789012/SAMLProviderName`. The `account-ID` value refers to the account that owns the [SAML provider \(p. 201\)](#).

saml:edupersonaffiliation[]

Works with [string operators \(p. 614\)](#).

This is an `eduPerson` attribute.

saml:edupersonassurance[]

Works with [string operators \(p. 614\)](#).

This is an `eduPerson` attribute.

saml:edupersonentitlement[]

Works with [string operators \(p. 614\)](#).

This is an `eduPerson` attribute.

saml:edupersonnickname[]

Works with [string operators \(p. 614\)](#).

This is an `eduPerson` attribute.

saml:edupersonorgdn

Works with [string operators \(p. 614\)](#).

This is an `eduPerson` attribute.

saml:edupersonorgunitdn[]

Works with [string operators \(p. 614\)](#).

This is an `eduPerson` attribute.

saml:edupersonprimaryaffiliation

Works with [string operators \(p. 614\)](#).

This is an `eduPerson` attribute.

saml:edupersonprimaryorgunitdn

Works with [string operators \(p. 614\)](#).

This is an `eduPerson` attribute.

saml:edupersonprincipalname

Works with [string operators \(p. 614\)](#).

This is an eduPerson attribute.

saml:edupersonscopedaffiliation[]

Works with [string operators \(p. 614\)](#).

This is an eduPerson attribute.

saml:edupersontargetedid[]

Works with [string operators \(p. 614\)](#).

This is an eduPerson attribute.

saml:eduorghomepageuri[]

Works with [string operators \(p. 614\)](#).

This is an eduOrg attribute.

saml:eduorgidentityauthnpolicyuri[]

Works with [string operators \(p. 614\)](#).

This is an eduOrg attribute.

saml:eduorglegalname[]

Works with [string operators \(p. 614\)](#).

This is an eduOrg attribute.

saml:eduorgsuperioruri[]

Works with [string operators \(p. 614\)](#).

This is an eduOrg attribute.

saml:eduorgwhitepagesuri[]

Works with [string operators \(p. 614\)](#).

This is an eduOrg attribute.

saml:givenName[]

Works with [string operators \(p. 614\)](#).

This is a givenName attribute.

saml:iss

Works with [string operators \(p. 614\)](#).

The issuer, which is represented by a URN.

saml:mail[]

Works with [string operators \(p. 614\)](#).

This is a mail attribute.

saml:name[]

Works with [string operators \(p. 614\)](#).

This is a `name` attribute.

saml:nameQualifier

Works with [string operators \(p. 614\)](#).

A hash value based on the friendly name of the SAML provider. The value is the concatenation of the following values, in order and separated by a '/' character:

1. The `Issuer` response value (`saml:iss`)
2. The AWS account ID
3. The friendly name (the last part of the ARN) of the SAML provider in IAM

The concatenation of the account ID and friendly name of the SAML provider is available to IAM policies as the key `saml:doc`. For more information, see [Uniquely Identifying Users in SAML-Based Federation \(p. 193\)](#).

saml:organizationStatus[]

Works with [string operators \(p. 614\)](#).

This is an `organizationStatus` attribute.

saml:primaryGroupSID[]

Works with [string operators \(p. 614\)](#).

This is a `primaryGroupSID` attribute.

saml:sub

Works with [string operators \(p. 614\)](#).

This is the subject of the claim, which includes a value that uniquely identifies an individual user within an organization (for example, `_cbb88bf52c2510eabe00c1642d4643f41430fe25e3`).

saml:sub_type

Works with [string operators \(p. 614\)](#).

This key can have the value `persistent`, `transient`, or consist of the full `Format` URI from the `Subject` and `NameID` elements used in your SAML assertion. A value of `persistent` indicates that the value in `saml:sub` is the same for a user between sessions. If the value is `transient`, the user has a different `saml:sub` value for each session. For information about the `NameID` element's `Format` attribute, see [Configuring SAML Assertions for the Authentication Response \(p. 206\)](#).

saml:surname[]

Works with [string operators \(p. 614\)](#).

This is a `surnameuid` attribute.

saml:uid[]

Works with [string operators \(p. 614\)](#).

This is a `uid` attribute.

saml:x500UniqueIdentifier[]

Works with [string operators \(p. 614\)](#).

This is an `x500UniqueIdentifier` attribute.

For general information about eduPerson and eduOrg attributes, see the [Internet2 website](#). For a list of eduPerson attributes, see [eduPerson Object Class Specification \(201203\)](#).

Condition keys whose type is a list can include multiple values. To create conditions in the policy for list values, you can use [set operators \(p. 621\)](#) (ForAllValues, ForAnyValue). For example, to allow any user whose affiliation is "faculty" or "staff" (but not "student"), you might use a condition like the following:

```
"Condition": {  
    "ForAllValues:StringLike": {  
        "saml:edupersonaffiliation": [ "faculty", "staff" ]  
    }  
}
```

SAML Role Permissions Policies

In the permissions policy of a role for SAML federation that defines what users are allowed to access in AWS, you can include the following keys:

saml:namequalifier

Works with [string operators \(p. 614\)](#).

This contains a hash value that represents the combination of the saml:doc and saml:iss values. It is used as a namespace qualifier; the combination of saml:namequalifier and saml:sub uniquely identifies a user.

saml:sub

Works with [string operators \(p. 614\)](#).

This is the subject of the claim, which includes a value that uniquely identifies an individual user within an organization (for example, _cbb88bf52c2510eabe00c1642d4643f41430fe25e3).

saml:sub_type

Works with [string operators \(p. 614\)](#).

This key can have the value persistent, transient, or consist of the full Format URI from the Subject and NameID elements used in your SAML assertion. A value of persistent indicates that the value in saml:sub is the same for a user between sessions. If the value is transient, the user has a different saml:sub value for each session. For information about the NameID element's Format attribute, see [Configuring SAML Assertions for the Authentication Response \(p. 206\)](#).

For more information about using these keys, see [About SAML 2.0-based Federation \(p. 190\)](#).

Available Keys for AWS STS

You can use the following condition keys in IAM role trust policies for roles that are assumed using AWS Security Token Service (AWS STS) operations.

sts:ExternalId

Works with [string operators \(p. 614\)](#).

A unique identifier that might be required when you assume a role in another account. If the administrator of the account to which the role belongs provided you with an external ID, then provide that value in the ExternalId parameter. This value can be any string, such as a passphrase

or account number. The primary function of the external ID is to address and prevent the confused deputy problem. For more information about the external ID and the confused deputy problem, see [How to Use an External ID When Granting Access to Your AWS Resources to a Third Party \(p. 233\)](#).

The `ExternalId` value must have a minimum of 2 characters and a maximum of 1,224 characters. The value must be alphanumeric without white space. It can also include the following symbols: plus (+), equal (=), comma (,), period (.), at (@), colon (:), forward slash (/), and hyphen (-).

sts:TransitiveTagKeys

Works with [string operators \(p. 614\)](#).

Use this key to compare the transitive session tag keys in the request with those specified in the policy. When you make a request using temporary security credentials, the [request context \(p. 612\)](#) includes the [aws:PrincipalTag \(p. 670\)](#) context key. This key includes a list of [session tags \(p. 299\)](#), [transitive session tags \(p. 305\)](#), and role tags. Transitive session tags are tags that persist into all subsequent sessions when you use the session credentials to assume another role. Assuming one role from another is called [role chaining \(p. 177\)](#).

You can use this condition key in a policy to require setting specific session tags as transitive when assuming a role or federating a user.

Actions, Resources, and Condition Keys for AWS Services

Each AWS service can define actions, resources, and condition context keys for use in IAM policies. This topic describes how the elements provided for each service are documented.

Each topic consists of tables that provide the list of available actions, resources, and condition keys.

The Actions Table

The **Actions** table lists all the actions that you can use in an IAM policy statement's `Action` element. Not all API operations that are defined by a service can be used as an action in an IAM policy. In addition, a service might define some actions that don't directly correspond to an API operation. Use this list to determine which actions you can use in an IAM policy. For more information about the `Action`, `Resource`, or `Condition` elements, see [IAM Policy Element Reference](#). The **Actions** and **Description** table columns are self-descriptive.

- The **Access Level** column describes how the action is classified (List, Read, Write, Permissions management, or Tagging). This classification can help you understand the level of access that an action grants when you use it in a policy. For more information about access levels, see [Understanding Access Level Summaries Within Policy Summaries](#).
- The **Resource Types** column indicates whether the action supports resource-level permissions. If the column is empty, then the action does not support resource-level permissions and you must specify all resources ("*") in your policy. If the column includes a resource type, then you can specify the resource ARN in the `Resource` element of your policy. For more information about that resource, refer to that row in the **Resource Types** table. All actions and resources that are included in one statement must be compatible with each other. If you specify a resource that is not valid for the action, any request to use that action fails, and the statement's `Effect` does not apply.

Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

- The **Condition Keys** column includes keys that you can specify in a policy statement's Condition element. Condition keys might be supported with an action, or with an action and a specific resource. Pay close attention to whether the key is in the same row as a specific resource type. This table does not include global condition keys that are available for any action or under unrelated circumstances. For more information about global condition keys, see [AWS Global Condition Context Keys](#).
- The **Dependent Actions** column includes any additional permissions that you must have, in addition to the permission for the action itself, to successfully call the action. This can be required if the action accesses more than one resource.

The Resource Types Table

The **Resource Types** table lists all the resource types that you can specify as an ARN in the Resource policy element. Not every resource type can be specified with every action. Some resource types work with only certain actions. If you specify a resource type in a statement with an action that does not support that resource type, then the statement doesn't allow access. For more information about the Resource element, see [IAM JSON Policy Elements: Resource](#).

- The **ARN** column specifies the Amazon Resource Name (ARN) format that you must use to reference resources of this type. The portions that are preceded by a \$ must be replaced by the actual values for your scenario. For example, if you see \$user-name in an ARN, you must replace that string with either the actual IAM user's name or a [policy variable](#) that contains an IAM user's name. For more information about ARNs, see [IAM ARNs](#).
- The **Condition Keys** column specifies condition context keys that you can include in an IAM policy statement only when both this resource and a supporting action from the table above are included in the statement.

The Condition Keys Table

The **Condition Keys** table lists all of the condition context keys that you can use in an IAM policy statement's Condition element. Not every key can be specified with every action or resource. Certain keys only work with certain types of actions and resources. For more information about the Condition element, see [IAM JSON Policy Elements: Condition](#).

- The **Type** column specifies the data type of the condition key. This data type determines which [condition operators](#) you can use to compare values in the request with the values in the policy statement. You must use an operator that is appropriate for the data type. If you use an incorrect operator, then the match always fails and the policy statement never applies.

If the **Type** column specifies a "List of ..." one of the simple types, then you can use [multiple keys and values](#) in your policies. Do this using condition set prefixes with your operators. Use the `ForAllValues` prefix to specify that **all** values in the request must match a value in the policy statement. Use the `ForAnyValue` prefix to specify that **at least one** value in the request matches one of the values in the policy statement.

Topics

- [Actions, Resources, and Condition Keys for AWS Accounts \(p. 693\)](#)
- [Actions, Resources, and Condition Keys for Alexa for Business \(p. 694\)](#)
- [Actions, Resources, and Condition Keys for AWS Amplify \(p. 701\)](#)
- [Actions, Resources, and Condition Keys for Amazon API Gateway \(p. 706\)](#)
- [Actions, Resources, and Condition Keys for AWS App Mesh \(p. 707\)](#)
- [Actions, Resources, and Condition Keys for AWS App Mesh Preview \(p. 712\)](#)
- [Actions, Resources, and Condition Keys for Application Auto Scaling \(p. 716\)](#)

- [Actions, Resources, and Condition Keys for Application Discovery \(p. 717\)](#)
- [Actions, Resources, and Condition Keys for Application Discovery Arsenal \(p. 721\)](#)
- [Actions, Resources, and Condition Keys for Amazon AppStream 2.0 \(p. 722\)](#)
- [Actions, Resources, and Condition Keys for AWS AppSync \(p. 731\)](#)
- [Actions, Resources, and Condition Keys for AWS Artifact \(p. 735\)](#)
- [Actions, Resources, and Condition Keys for Amazon Athena \(p. 737\)](#)
- [Actions, Resources, and Condition Keys for AWS Auto Scaling \(p. 741\)](#)
- [Actions, Resources, and Condition Keys for AWS Backup \(p. 742\)](#)
- [Actions, Resources, and Condition Keys for AWS Backup storage \(p. 747\)](#)
- [Actions, Resources, and Condition Keys for AWS Batch \(p. 748\)](#)
- [Actions, Resources, and Condition Keys for AWS Billing \(p. 750\)](#)
- [Actions, Resources, and Condition Keys for AWS Budget Service \(p. 752\)](#)
- [Actions, Resources, and Condition Keys for AWS Certificate Manager \(p. 753\)](#)
- [Actions, Resources, and Condition Keys for AWS Certificate Manager Private Certificate Authority \(p. 756\)](#)
- [Actions, Resources, and Condition Keys for AWS Chatbot \(p. 759\)](#)
- [Actions, Resources, and Condition Keys for Amazon Chime \(p. 761\)](#)
- [Actions, Resources, and Condition Keys for Amazon Cloud Directory \(p. 774\)](#)
- [Actions, Resources, and Condition Keys for AWS Cloud Map \(p. 780\)](#)
- [Actions, Resources, and Condition Keys for AWS Cloud9 \(p. 783\)](#)
- [Actions, Resources, and Condition Keys for AWS CloudFormation \(p. 786\)](#)
- [Actions, Resources, and Condition Keys for Amazon CloudFront \(p. 793\)](#)
- [Actions, Resources, and Condition Keys for AWS CloudHSM \(p. 800\)](#)
- [Actions, Resources, and Condition Keys for Amazon CloudSearch \(p. 804\)](#)
- [Actions, Resources, and Condition Keys for AWS CloudTrail \(p. 807\)](#)
- [Actions, Resources, and Condition Keys for Amazon CloudWatch \(p. 809\)](#)
- [Actions, Resources, and Condition Keys for CloudWatch Application Insights \(p. 813\)](#)
- [Actions, Resources, and Condition Keys for Amazon CloudWatch Logs \(p. 815\)](#)
- [Actions, Resources, and Condition Keys for Amazon CloudWatch Synthetics \(p. 820\)](#)
- [Actions, Resources, and Condition Keys for AWS Code Signing for Amazon FreeRTOS \(p. 822\)](#)
- [Actions, Resources, and Condition Keys for AWS CodeBuild \(p. 824\)](#)
- [Actions, Resources, and Condition Keys for AWS CodeCommit \(p. 830\)](#)
- [Actions, Resources, and Condition Keys for AWS CodeDeploy \(p. 840\)](#)
- [Actions, Resources, and Condition Keys for Amazon CodeGuru Profiler \(p. 845\)](#)
- [Actions, Resources, and Condition Keys for Amazon CodeGuru Reviewer \(p. 847\)](#)
- [Actions, Resources, and Condition Keys for AWS CodePipeline \(p. 849\)](#)
- [Actions, Resources, and Condition Keys for AWS CodeStar \(p. 854\)](#)
- [Actions, Resources, and Condition Keys for AWS CodeStar Notifications \(p. 858\)](#)
- [Actions, Resources, and Condition Keys for Amazon Cognito Identity \(p. 863\)](#)
- [Actions, Resources, and Condition Keys for Amazon Cognito Sync \(p. 866\)](#)
- [Actions, Resources, and Condition Keys for Amazon Cognito User Pools \(p. 869\)](#)
- [Actions, Resources, and Condition Keys for Amazon Comprehend \(p. 877\)](#)

- [Actions, Resources, and Condition Keys for Comprehend Medical \(p. 883\)](#)
- [Actions, Resources, and Condition Keys for Compute Optimizer \(p. 884\)](#)
- [Actions, Resources, and Condition Keys for AWS Config \(p. 886\)](#)
- [Actions, Resources, and Condition Keys for Amazon Connect \(p. 894\)](#)
- [Actions, Resources, and Condition Keys for AWS Cost and Usage Report \(p. 901\)](#)
- [Actions, Resources, and Condition Keys for AWS Cost Explorer Service \(p. 902\)](#)
- [Actions, Resources, and Condition Keys for AWS Data Exchange \(p. 905\)](#)
- [Actions, Resources, and Condition Keys for Amazon Data Lifecycle Manager \(p. 908\)](#)
- [Actions, Resources, and Condition Keys for Data Pipeline \(p. 910\)](#)
- [Actions, Resources, and Condition Keys for AWS Database Migration Service \(p. 913\)](#)
- [Actions, Resources, and Condition Keys for Database Query Metadata Service \(p. 921\)](#)
- [Actions, Resources, and Condition Keys for DataSync \(p. 922\)](#)
- [Actions, Resources, and Condition Keys for AWS DeepLens \(p. 927\)](#)
- [Actions, Resources, and Condition Keys for AWS DeepRacer \(p. 929\)](#)
- [Actions, Resources, and Condition Keys for Amazon Detective \(p. 933\)](#)
- [Actions, Resources, and Condition Keys for AWS Device Farm \(p. 935\)](#)
- [Actions, Resources, and Condition Keys for AWS Direct Connect \(p. 944\)](#)
- [Actions, Resources, and Condition Keys for AWS Directory Service \(p. 952\)](#)
- [Actions, Resources, and Condition Keys for Amazon DynamoDB \(p. 959\)](#)
- [Actions, Resources, and Condition Keys for Amazon DynamoDB Accelerator \(DAX\) \(p. 966\)](#)
- [Actions, Resources, and Condition Keys for Amazon EC2 \(p. 970\)](#)
- [Actions, Resources, and Condition Keys for Amazon EC2 Auto Scaling \(p. 1059\)](#)
- [Actions, Resources, and Condition Keys for Amazon EC2 Image Builder \(p. 1069\)](#)
- [Actions, Resources, and Condition Keys for Amazon EC2 Instance Connect \(p. 1075\)](#)
- [Actions, Resources, and Condition Keys for AWS Elastic Beanstalk \(p. 1077\)](#)
- [Actions, Resources, and Condition Keys for Amazon Elastic Block Store \(p. 1085\)](#)
- [Actions, Resources, and Condition Keys for Amazon Elastic Container Registry \(p. 1087\)](#)
- [Actions, Resources, and Condition Keys for Amazon Elastic Container Service \(p. 1091\)](#)
- [Actions, Resources, and Condition Keys for Amazon Elastic Container Service for Kubernetes \(p. 1099\)](#)
- [Actions, Resources, and Condition Keys for Amazon Elastic File System \(p. 1103\)](#)
- [Actions, Resources, and Condition Keys for Amazon Elastic Inference \(p. 1107\)](#)
- [Actions, Resources, and Condition Keys for Elastic Load Balancing \(p. 1108\)](#)
- [Actions, Resources, and Condition Keys for Elastic Load Balancing V2 \(p. 1112\)](#)
- [Actions, Resources, and Condition Keys for Amazon Elastic MapReduce \(p. 1118\)](#)
- [Actions, Resources, and Condition Keys for Amazon Elastic Transcoder \(p. 1123\)](#)
- [Actions, Resources, and Condition Keys for Amazon ElastiCache \(p. 1126\)](#)
- [Actions, Resources, and Condition Keys for Amazon Elasticsearch Service \(p. 1131\)](#)
- [Actions, Resources, and Condition Keys for AWS Elemental MediaConnect \(p. 1135\)](#)
- [Actions, Resources, and Condition Keys for AWS Elemental MediaConvert \(p. 1137\)](#)
- [Actions, Resources, and Condition Keys for AWS Elemental MediaLive \(p. 1141\)](#)
- [Actions, Resources, and Condition Keys for AWS Elemental MediaPackage \(p. 1147\)](#)

- [Actions, Resources, and Condition Keys for AWS Elemental MediaPackage VOD \(p. 1150\)](#)
- [Actions, Resources, and Condition Keys for AWS Elemental MediaStore \(p. 1152\)](#)
- [Actions, Resources, and Condition Keys for AWS Elemental MediaTailor \(p. 1154\)](#)
- [Actions, Resources, and Condition Keys for Amazon EventBridge \(p. 1157\)](#)
- [Actions, Resources, and Condition Keys for Amazon EventBridge Schemas \(p. 1162\)](#)
- [Actions, Resources, and Condition Keys for AWS Firewall Manager \(p. 1166\)](#)
- [Actions, Resources, and Condition Keys for Amazon Forecast \(p. 1169\)](#)
- [Actions, Resources, and Condition Keys for Amazon Fraud Detector \(p. 1172\)](#)
- [Actions, Resources, and Condition Keys for Amazon FreeRTOS \(p. 1176\)](#)
- [Actions, Resources, and Condition Keys for Amazon FSx \(p. 1178\)](#)
- [Actions, Resources, and Condition Keys for Amazon GameLift \(p. 1182\)](#)
- [Actions, Resources, and Condition Keys for Amazon Glacier \(p. 1190\)](#)
- [Actions, Resources, and Condition Keys for AWS Global Accelerator \(p. 1193\)](#)
- [Actions, Resources, and Condition Keys for AWS Glue \(p. 1196\)](#)
- [Actions, Resources, and Condition Keys for AWS Ground Station \(p. 1206\)](#)
- [Actions, Resources, and Condition Keys for Amazon GroundTruth Labeling \(p. 1210\)](#)
- [Actions, Resources, and Condition Keys for Amazon GuardDuty \(p. 1211\)](#)
- [Actions, Resources, and Condition Keys for AWS Health APIs and Notifications \(p. 1218\)](#)
- [Actions, Resources, and Condition Keys for IAM Access Analyzer \(p. 1221\)](#)
- [Actions, Resources, and Condition Keys for Identity And Access Management \(p. 1224\)](#)
- [Actions, Resources, and Condition Keys for AWS Import Export Disk Service \(p. 1239\)](#)
- [Actions, Resources, and Condition Keys for Amazon Inspector \(p. 1240\)](#)
- [Actions, Resources, and Condition Keys for AWS IoT \(p. 1244\)](#)
- [Actions, Resources, and Condition Keys for AWS IoT 1-Click \(p. 1262\)](#)
- [Actions, Resources, and Condition Keys for AWS IoT Analytics \(p. 1265\)](#)
- [Actions, Resources, and Condition Keys for AWS IoT Events \(p. 1270\)](#)
- [Actions, Resources, and Condition Keys for AWS IoT Greengrass \(p. 1273\)](#)
- [Actions, Resources, and Condition Keys for AWS IoT SiteWise \(p. 1285\)](#)
- [Actions, Resources, and Condition Keys for AWS IoT Things Graph \(p. 1293\)](#)
- [Actions, Resources, and Condition Keys for AWS IQ \(p. 1298\)](#)
- [Actions, Resources, and Condition Keys for AWS IQ Permissions \(p. 1299\)](#)
- [Actions, Resources, and Condition Keys for Amazon Kendra \(p. 1299\)](#)
- [Actions, Resources, and Condition Keys for AWS Key Management Service \(p. 1302\)](#)
- [Actions, Resources, and Condition Keys for Amazon Kinesis \(p. 1314\)](#)
- [Actions, Resources, and Condition Keys for Amazon Kinesis Analytics \(p. 1317\)](#)
- [Actions, Resources, and Condition Keys for Amazon Kinesis Analytics V2 \(p. 1320\)](#)
- [Actions, Resources, and Condition Keys for Amazon Kinesis Firehose \(p. 1323\)](#)
- [Actions, Resources, and Condition Keys for Amazon Kinesis Video Streams \(p. 1325\)](#)
- [Actions, Resources, and Condition Keys for AWS Lake Formation \(p. 1330\)](#)
- [Actions, Resources, and Condition Keys for AWS Lambda \(p. 1331\)](#)
- [Actions, Resources, and Condition Keys for Launch Wizard \(p. 1336\)](#)
- [Actions, Resources, and Condition Keys for Amazon Lex \(p. 1338\)](#)

- [Actions, Resources, and Condition Keys for AWS License Manager \(p. 1341\)](#)
- [Actions, Resources, and Condition Keys for Amazon Lightsail \(p. 1344\)](#)
- [Actions, Resources, and Condition Keys for Amazon Machine Learning \(p. 1356\)](#)
- [Actions, Resources, and Condition Keys for Amazon Macie \(p. 1360\)](#)
- [Actions, Resources, and Condition Keys for Manage Amazon API Gateway \(p. 1362\)](#)
- [Actions, Resources, and Condition Keys for AWS Managed Apache Cassandra Service \(p. 1364\)](#)
- [Actions, Resources, and Condition Keys for Amazon Managed Blockchain \(p. 1366\)](#)
- [Actions, Resources, and Condition Keys for Amazon Managed Streaming for Kafka \(p. 1369\)](#)
- [Actions, Resources, and Condition Keys for AWS Marketplace \(p. 1372\)](#)
- [Actions, Resources, and Condition Keys for AWS Marketplace Catalog \(p. 1374\)](#)
- [Actions, Resources, and Condition Keys for AWS Marketplace Entitlement Service \(p. 1376\)](#)
- [Actions, Resources, and Condition Keys for AWS Marketplace Image Building Service \(p. 1377\)](#)
- [Actions, Resources, and Condition Keys for AWS Marketplace Management Portal \(p. 1378\)](#)
- [Actions, Resources, and Condition Keys for AWS Marketplace Metering Service \(p. 1379\)](#)
- [Actions, Resources, and Condition Keys for AWS Marketplace Procurement Systems Integration \(p. 1381\)](#)
- [Actions, Resources, and Condition Keys for Amazon Mechanical Turk \(p. 1382\)](#)
- [Actions, Resources, and Condition Keys for Amazon Message Delivery Service \(p. 1387\)](#)
- [Actions, Resources, and Condition Keys for AWS Migration Hub \(p. 1388\)](#)
- [Actions, Resources, and Condition Keys for Amazon Mobile Analytics \(p. 1390\)](#)
- [Actions, Resources, and Condition Keys for AWS Mobile Hub \(p. 1391\)](#)
- [Actions, Resources, and Condition Keys for Amazon MQ \(p. 1394\)](#)
- [Actions, Resources, and Condition Keys for Amazon Neptune \(p. 1397\)](#)
- [Actions, Resources, and Condition Keys for Network Manager \(p. 1398\)](#)
- [Actions, Resources, and Condition Keys for AWS OpsWorks \(p. 1404\)](#)
- [Actions, Resources, and Condition Keys for AWS OpsWorks Configuration Management \(p. 1410\)](#)
- [Actions, Resources, and Condition Keys for AWS Organizations \(p. 1411\)](#)
- [Actions, Resources, and Condition Keys for AWS Outposts \(p. 1417\)](#)
- [Actions, Resources, and Condition Keys for AWS Performance Insights \(p. 1419\)](#)
- [Actions, Resources, and Condition Keys for Amazon Personalize \(p. 1420\)](#)
- [Actions, Resources, and Condition Keys for Amazon Pinpoint \(p. 1424\)](#)
- [Actions, Resources, and Condition Keys for Amazon Pinpoint Email Service \(p. 1435\)](#)
- [Actions, Resources, and Condition Keys for Amazon Pinpoint SMS and Voice Service \(p. 1442\)](#)
- [Actions, Resources, and Condition Keys for Amazon Polly \(p. 1443\)](#)
- [Actions, Resources, and Condition Keys for AWS Price List \(p. 1445\)](#)
- [Actions, Resources, and Condition Keys for AWS Private Marketplace \(p. 1446\)](#)
- [Actions, Resources, and Condition Keys for Amazon QLDB \(p. 1451\)](#)
- [Actions, Resources, and Condition Keys for Amazon QuickSight \(p. 1454\)](#)
- [Actions, Resources, and Condition Keys for Amazon RDS \(p. 1457\)](#)
- [Actions, Resources, and Condition Keys for Amazon RDS Data API \(p. 1478\)](#)
- [Actions, Resources, and Condition Keys for Amazon RDS IAM Authentication \(p. 1479\)](#)
- [Actions, Resources, and Condition Keys for Amazon Redshift \(p. 1480\)](#)
- [Actions, Resources, and Condition Keys for Amazon Rekognition \(p. 1491\)](#)

- [Actions, Resources, and Condition Keys for AWS Resource Access Manager \(p. 1495\)](#)
- [Actions, Resources, and Condition Keys for Amazon Resource Group Tagging API \(p. 1502\)](#)
- [Actions, Resources, and Condition Keys for AWS Resource Groups \(p. 1503\)](#)
- [Actions, Resources, and Condition Keys for AWS RoboMaker \(p. 1506\)](#)
- [Actions, Resources, and Condition Keys for Amazon Route 53 \(p. 1511\)](#)
- [Actions, Resources, and Condition Keys for Amazon Route 53 Resolver \(p. 1518\)](#)
- [Actions, Resources, and Condition Keys for Amazon Route53 Domains \(p. 1522\)](#)
- [Actions, Resources, and Condition Keys for Amazon S3 \(p. 1525\)](#)
- [Actions, Resources, and Condition Keys for Amazon SageMaker \(p. 1579\)](#)
- [Actions, Resources, and Condition Keys for AWS Savings Plans \(p. 1608\)](#)
- [Actions, Resources, and Condition Keys for AWS Secrets Manager \(p. 1611\)](#)
- [Actions, Resources, and Condition Keys for AWS Security Hub \(p. 1618\)](#)
- [Actions, Resources, and Condition Keys for AWS Security Token Service \(p. 1623\)](#)
- [Actions, Resources, and Condition Keys for AWS Server Migration Service \(p. 1631\)](#)
- [Actions, Resources, and Condition Keys for AWS Serverless Application Repository \(p. 1634\)](#)
- [Actions, Resources, and Condition Keys for AWS Service Catalog \(p. 1636\)](#)
- [Actions, Resources, and Condition Keys for Service Quotas \(p. 1644\)](#)
- [Actions, Resources, and Condition Keys for Amazon SES \(p. 1647\)](#)
- [Actions, Resources, and Condition Keys for Amazon Session Manager Message Gateway Service \(p. 1655\)](#)
- [Actions, Resources, and Condition Keys for AWS Shield \(p. 1656\)](#)
- [Actions, Resources, and Condition Keys for Amazon Simple Workflow Service \(p. 1659\)](#)
- [Actions, Resources, and Condition Keys for Amazon SimpleDB \(p. 1667\)](#)
- [Actions, Resources, and Condition Keys for AWS Snowball \(p. 1669\)](#)
- [Actions, Resources, and Condition Keys for Amazon SNS \(p. 1671\)](#)
- [Actions, Resources, and Condition Keys for Amazon SQS \(p. 1676\)](#)
- [Actions, Resources, and Condition Keys for AWS SSO \(p. 1678\)](#)
- [Actions, Resources, and Condition Keys for AWS SSO Directory \(p. 1682\)](#)
- [Actions, Resources, and Condition Keys for AWS Step Functions \(p. 1686\)](#)
- [Actions, Resources, and Condition Keys for Amazon Storage Gateway \(p. 1689\)](#)
- [Actions, Resources, and Condition Keys for Amazon Sumerian \(p. 1698\)](#)
- [Actions, Resources, and Condition Keys for AWS Support \(p. 1699\)](#)
- [Actions, Resources, and Condition Keys for AWS Systems Manager \(p. 1702\)](#)
- [Actions, Resources, and Condition Keys for Amazon Textract \(p. 1716\)](#)
- [Actions, Resources, and Condition Keys for Amazon Transcribe \(p. 1718\)](#)
- [Actions, Resources, and Condition Keys for AWS Transfer for SFTP \(p. 1720\)](#)
- [Actions, Resources, and Condition Keys for Amazon Translate \(p. 1723\)](#)
- [Actions, Resources, and Condition Keys for AWS Trusted Advisor \(p. 1725\)](#)
- [Actions, Resources, and Condition Keys for AWS WAF \(p. 1726\)](#)
- [Actions, Resources, and Condition Keys for AWS WAF Regional \(p. 1735\)](#)
- [Actions, Resources, and Condition Keys for AWS WAF V2 \(p. 1743\)](#)
- [Actions, Resources, and Condition Keys for AWS Well-Architected Tool \(p. 1750\)](#)
- [Actions, Resources, and Condition Keys for Amazon WorkDocs \(p. 1751\)](#)

- [Actions, Resources, and Condition Keys for Amazon WorkLink \(p. 1756\)](#)
- [Actions, Resources, and Condition Keys for Amazon WorkMail \(p. 1760\)](#)
- [Actions, Resources, and Condition Keys for Amazon WorkMail Message Flow \(p. 1768\)](#)
- [Actions, Resources, and Condition Keys for Amazon WorkSpaces \(p. 1769\)](#)
- [Actions, Resources, and Condition Keys for Amazon WorkSpaces Application Manager \(p. 1772\)](#)
- [Actions, Resources, and Condition Keys for AWS X-Ray \(p. 1773\)](#)

Actions, Resources, and Condition Keys for AWS Accounts

AWS Accounts (service prefix: account) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Accounts \(p. 693\)](#)
- [Resource Types Defined by AWS Accounts \(p. 694\)](#)
- [Condition Keys for AWS Accounts \(p. 694\)](#)

Actions Defined by AWS Accounts

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DisableRegion	Grants permission to disable a region	Write		account:TargetRegion (p. 694)	
EnableRegion	Grants permission to enable a region	Write		account:TargetRegion (p. 694)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListRegions	Grants permission to list regions	List			

Resource Types Defined by AWS Accounts

AWS Accounts does not support specifying a resource ARN in the `Resource` element of an IAM policy statement. To allow access to AWS Accounts, specify “`Resource`”: “`*`” in your policy.

Condition Keys for AWS Accounts

AWS Accounts defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
account:TargetRegion	Filters access by a list of regions	String

Actions, Resources, and Condition Keys for Alexa for Business

Alexa for Business (service prefix: `a4b`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- View a list of the [API operations available for this service](#).

Topics

- [Actions Defined by Alexa for Business \(p. 694\)](#)
- [Resource Types Defined by Alexa for Business \(p. 700\)](#)
- [Condition Keys for Alexa for Business \(p. 701\)](#)

Actions Defined by Alexa for Business

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources (“`*`”) in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ApproveSkill	Associates a skill with the organization under the customer's AWS account. If a skill is private, the user implicitly accepts access to this skill during enablement.	Write			
AssociateContactWithAddressBook	Associates a contact with a given address book.	Write	addressbook* (p. 701)		
			contact* (p. 701)		
AssociateDeviceWithRoom	Associates device with given room.	Write	device* (p. 700)		
			room* (p. 700)		
AssociateSkillGroupWithRoom	Associates the skill group with given room. SkillGroup ARN and Room ARN must be specified.	Write	room* (p. 700)		
			skillgroup* (p. 701)		
AssociateSkillWithSkillGroup	Associates a skill with a skill group.	Write	skillgroup* (p. 701)		
AssociateSkillWithUser	Makes a private skill available to enrolled users to enable on their devices.	Write			
CompleteRegistration [permission only]	Completes the operation of registering an Alexa device.	Write			
CreateAddressBook	Creates an address book with the specified details.	Write			
CreateBusinessReport	Creates a recurring schedule for usage reports to deliver to the specified S3 location with a specified daily or weekly interval.	Write			
CreateConferenceProvider	Adds a new conference provider under the user's AWS account.	Write			
CreateContact	Creates a contact with the specified details.	Write			
CreateProfile	Creates a new profile.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateRoom	Create room with the specified details.	Write	profile* (p. 700)		
CreateSkillGroup	Creates a skill group with given name and description.	Write			
CreateUser	Creates a user.	Write	user* (p. 701)		
DeleteAddressBook	Deletes an address book by the address book ARN.	Write	addressbook* (p. 701)		
DeleteBusinessReportDeliverySchedule	Deletes the recurring report delivery schedule with the specified schedule ARN.	Write	schedule* (p. 701)		
DeleteConferenceProvider	Deletes a conference provider.	Write	conferenceprovider* (p. 701)		
DeleteContact	Deletes a contact by the contact ARN.	Write	contact* (p. 701)		
DeleteDevice	Removes a device from Alexa For Business.	Write	device* (p. 700)		
DeleteProfile	Delete profile by profile ARN.	Write	profile* (p. 700)		
DeleteRoom	Delete room.	Write	room* (p. 700)		
DeleteRoomSkillParameter	Delete a parameter from a skill and room.	Write	room* (p. 700)		
DeleteSkillAuthorizations	Unlinks a third-party account from a skill.	Write	room* (p. 700)		
DeleteSkillGroup	Deletes skill group with skill group ARN. Skillgroup ARN must be specified.	Write	skillgroup* (p. 701)		
DeleteUser	Delete a user.	Write	user* (p. 701)		
DisassociateContactFromAddressBook	Disassociates a contact from a given address book.	Write	addressbook* (p. 701)		
			contact* (p. 701)		
DisassociateDeviceFromRoom	Disassociates device from its current Room.	Write	device* (p. 700)		
DisassociateSkillFromSkillGroup	Disassociates a skill from a skill group.	Write	skillgroup* (p. 701)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DisassociateSkillForRoom	Makes a private skill unavailable for remote users and prevents them from enabling it on their devices.	Write	user* (p. 701)		
DisassociateSkillGroupFromRoom	Disassociates the skill group <code>GroupArn</code> from the room <code>RoomArn</code> . SkillGroup ARN and Room ARN must be specified.	Write	room* (p. 700)		
			skillgroup* (p. 701)		
ForgetSmartHomeAssociated	Forgets smart home appliances associated to a room.	Write	room* (p. 700)		
GetAddressBook	Gets the address book details by the address book ARN.	Read	addressbook* (p. 701)		
GetConferencePreferences	Retrieves the existing conference preferences.	Read			
GetConferenceProvider	Gets details about a specific conference provider.	Read	conferenceprovider* (p. 701)		
GetContact	Gets the contact details by the contact ARN.	Read	contact* (p. 701)		
GetDevice	Get device details.	Read	device* (p. 700)		
GetNetworkProfile	Gets the network profile details by the network profile ARN.	Read	networkprofile* (p. 701)		
GetProfile	Gets profile when provided with Profile ARN.	Read	profile* (p. 700)		
GetRoom	Get room details.	Read	room* (p. 700)		
GetRoomSkillParameter	Get an existing parameter that has been set for a skill and room.	Read	room* (p. 700)		
GetSkillGroup	Gets skill group details with skill group ARN. Skillgroup ARN must be specified.	Read	skillgroup* (p. 701)		
ListBusinessReporters	Lists the details of the schedules that have been configured.	List			
ListConferenceProviders	Lists conference providers under a specific AWS account.	List			
ListDeviceEvents	Lists the device event history, including device connection status, for up to 30 days.	List	device* (p. 700)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListSkills	Lists skills.	List			
ListSkillsStoreCategories	Lists all categories in the Alexa skill store.	List			
ListSkillsStoreSkillsByCategory	Lists all skills in the Alexa skill store by category.	List			
ListSmartHomeAppliances	Lists all of the smart home appliances associated with a room.	List	room* (p. 700)		
ListTags	Lists all tags on a resource.	Read	device (p. 700)		
			room (p. 700)		
			user (p. 701)		
PutConferencePreferences	Sets the conference preferences for a specific conference provider at the account level.	Write			
PutDeviceSetupEvents [permission only]	Publishes Alexa device setup events.	Write			
PutRoomSkillParameters	Put a room specific parameter for a skill.	Write	room* (p. 700)		
PutSkillAuthorization	Links a user's account to a third-party skill provider. If this API operation is called by an assumed IAM role, the skill being linked must be a private skill. Also, the skill must be owned by the AWS account that assumed the IAM role.	Write	room* (p. 700)		
RegisterAVSDevice	Registers an Alexa-enabled device built by an Original Equipment Manufacturer (OEM) using Alexa Voice Service (AVS).	Write			
RegisterDevice [permission only]	Registers an Alexa device.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
RejectSkill	Disassociates a skill from the organization under a user's AWS account. If the skill is a private skill, it moves to an AcceptStatus of PENDING.	Write			
ResolveRoom	Returns resolved room information.	Read			
RevokeInvitation	Revoke an invitation.	Write	user* (p. 701)		
SearchAddressBooks	Searches address books and lists the ones that meet a set of filter and sort criteria.	List			
SearchContacts	Searches contacts and lists the ones that meet a set of filter and sort criteria.	List			
SearchDevices	Search for devices.	List			
SearchNetworkProfiles	Searches network profiles and lists the ones that meet a set of filter and sort criteria.	List			
SearchProfiles	Search for profiles.	List			
SearchRooms	Search for rooms.	List			
SearchSkillGroups	Search for skill groups.	List			
SearchUsers	Search for users.	List			
SendInvitation	Send an invitation to a user.	Write	user* (p. 701)		
StartDeviceSync	Restore the device and its account to its known, default settings by clearing all information and settings set by its previous users.	Write			
StartSmartHomeApplianceDiscovery	Initiates the discovery of Any smart home appliances associated with the room.	Read	room* (p. 700)		
TagResource	Adds metadata tags to a resource.	Tagging	device (p. 700)		
			room (p. 700)		
			user (p. 701)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UntagResource	Removes metadata tags from a resource.	Tagging	device (p. 700)		
			room (p. 700)		
			user (p. 701)		
UpdateAddressBook	Updates address book details by the address book ARN.	Write	addressbook*(p. 701)		
UpdateBusinessReportDelivery	Updates the configuration of the report delivery schedule with the specified schedule ARN.	Write	schedule*(p. 701)		
UpdateConferenceProvider	Updates an existing conference provider's settings.	Write	conferenceprovider*(p. 701)		
UpdateContact	Updates the contact details by the contact ARN.	Write	contact*(p. 701)		
UpdateDevice	Updates device name.	Write	device*(p. 700)		
UpdateProfile	Updates an existing profile.	Write	profile*(p. 700)		
UpdateRoom	Update room details.	Write	room*(p. 700)		
UpdateSkillGroup	Updates skill group details with skill group ARN. Skillgroup ARN must be specified.	Write	skillgroup*(p. 701)		

Resource Types Defined by Alexa for Business

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 694\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
profile	arn:\${Partition}:a4b:\${Region}:\${Account}:profile/\${Resource_id}	
room	arn:\${Partition}:a4b:\${Region}:\${Account}:room/\${Resource_id}	aws:ResourceTag/ {\$TagKey} (p. 701)
device	arn:\${Partition}:a4b:\${Region}:\${Account}:device/\${Resource_id}	aws:ResourceTag/ {\$TagKey} (p. 701)

Resource Types	ARN	Condition Keys
skillgroup	arn:\${Partition}:a4b:\${Region}: \${Account}:skill-group/\${Resource_id}	
user	arn:\${Partition}:a4b:\${Region}: \${Account}:user/\${Resource_id}	aws:ResourceTag/\${TagKey} (p. 701)
addressbook	arn:\${Partition}:a4b:\${Region}: \${Account}:address-book/\${Resource_id}	
conferenceprovider	arn:\${Partition}:a4b:\${Region}: \${Account}:conference-provider/ \${Resource_id}	
contact	arn:\${Partition}:a4b:\${Region}: \${Account}:contact/\${Resource_id}	
schedule	arn:\${Partition}:a4b:\${Region}: \${Account}:schedule/\${Resource_id}	
networkprofile	arn:\${Partition}:a4b:\${Region}: \${Account}:network-profile/\${Resource_id}	

Condition Keys for Alexa for Business

Alexa for Business defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
a4b:amazonId	Filters actions based on the Amazon Id in the request	String
a4b:filters_deviceType	Filters actions based on the device type in the request	String
aws:RequestTag/\${TagKey}	Filters actions based on the allowed set of values for each of the tags	String
aws:ResourceTag/\${TagKey}	Filters actions based on tag-value associated with the resource	String
aws:TagKeys	Filters actions based on the presence of mandatory tags in the request	String

Actions, Resources, and Condition Keys for AWS Amplify

AWS Amplify (service prefix: `amplify`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Amplify \(p. 702\)](#)
- [Resource Types Defined by AWS Amplify \(p. 705\)](#)
- [Condition Keys for AWS Amplify \(p. 705\)](#)

Actions Defined by AWS Amplify

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateApp	Creates a new Amplify App.	Write	apps* (p. 705)		
				aws:RequestTag/ \${TagKey} (p. 706)	
				aws:TagKeys (p. 706)	
CreateBackendEnvironment	Creates a new backend environment for an Amplify App.	Write	apps* (p. 705)		
				aws:RequestTag/ \${TagKey} (p. 706)	
CreateBranch	Creates a new Branch for an Amplify App.	Write	apps* (p. 705)		
				aws:RequestTag/ \${TagKey} (p. 706)	
				aws:TagKeys (p. 706)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateDeployment	Create a deployment for manual deploy apps. (Apps are not connected to repository)	Write	branches* (p. 705)		
CreateDomainAssociation	Create a new DomainAssociation App	Write	apps* (p. 705)	aws:RequestTag/ \${TagKey} (p. 706) aws:TagKeys (p. 706)	
CreateWebHook	Create a new webhook on an App.	Write	branches* (p. 705)		
DeleteApp	Delete an existing Amplify App by appId.	Write	apps* (p. 705)		
DeleteBackendEnvironment	Deletes a branch for an Amplify App.	Write	apps* (p. 705)		
DeleteBranch	Deletes a branch for an Amplify App.	Write	branches* (p. 705)		
DeleteDomainAssociation	Deletes a DomainAssociation.	Write	domains* (p. 705)		
DeleteJob	Delete a job, for an Amplify branch, part of Amplify App.	Write	jobs* (p. 705)		
DeleteWebHook	Delete a webhook by id.	Write	apps* (p. 705)		
GenerateAccessLogs	Generate website access logs for a specific time range via a pre-signed URL.	Write	apps* (p. 705)		
GetApp	Retrieves an existing Amplify App by appId.	Read	apps* (p. 705)		
GetArtifactUrl	Retrieves artifact info that corresponds to a artifactId.	Read	apps* (p. 705)		
GetBackendEnvironment	Retrieves a backend environment for an Amplify App.	Read	apps* (p. 705)		
GetBranch	Retrieves a branch for an Amplify App.	Read	branches* (p. 705)		
GetDomainAssociation	Retrieves domain info that corresponds to an appId and domainName.	Read	domains* (p. 705)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetJob	Get a job for a branch, part of an Amplify App.	Read	jobs* (p. 705)		
GetWebHook	Retrieves webhook info that corresponds to a webhookId.	Read	apps* (p. 705)		
ListApps	Lists existing Amplify Apps.	List			
ListArtifacts	List artifacts with an app, a branch, a job and an artifact type.	List	apps* (p. 705)		
ListBackendEnvironments	Lists backend environments for an Amplify App.	List	apps* (p. 705)		
ListBranches	Lists branches for an Amplify App.	List	apps* (p. 705)		
ListDomainAssociations	List domains with an app	List	apps* (p. 705)		
ListJobs	List Jobs for a branch, part of an Amplify App.	List	branches* (p. 705)		
ListWebHooks	List webhooks on an App.	List	apps* (p. 705)		
StartDeployment	Start a deployment for manual deploy apps. (Apps are not connected to repository)	Write	branches* (p. 705)		
StartJob	Starts a new job for a branch, part of an Amplify App.	Write	jobs* (p. 705)		
StopJob	Stop a job that is in progress, for an Amplify branch, part of Amplify App.	Write	jobs* (p. 705)		
TagResource	This action tags an AWS Amplify Console resource.	Tagging	apps (p. 705)		
			branches (p. 705)		
			jobs (p. 705)		
				aws:TagKeys (p. 706) aws:RequestTag/ {\$TagKey} (p. 706)	
UntagResource	This action removes a tag from an AWS Amplify Console resource.	Tagging	apps (p. 705)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			branches (p. 705)		
			jobs (p. 705)		
				aws:TagKeys (p. 706)	
UpdateApp	Updates an existing Amplify App.	Write	apps* (p. 705)		
UpdateBranch	Updates a branch for an Amplify App.	Write	branches* (p. 705)		
UpdateDomainAssociation	Update a DomainAssociation on CreateApp	Write	domains* (p. 705)		
UpdateWebHook	Update a webhook.	Write	apps* (p. 705)		

Resource Types Defined by AWS Amplify

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 702\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
apps	arn:\${Partition}:amplify:\${Region}: \${Account}:apps/\${AppId}	aws:ResourceTag/ \${TagKey} (p. 706)
branches	arn:\${Partition}:amplify:\${Region}: \${Account}:apps/\${AppId}/branches/ \${BranchName}	aws:ResourceTag/ \${TagKey} (p. 706)
jobs	arn:\${Partition}:amplify:\${Region}: \${Account}:apps/\${AppId}/branches/ \${BranchName}/jobs/\${JobId}	
domains	arn:\${Partition}:amplify:\${Region}: \${Account}:apps/\${AppId}/domains/ \${DomainName}	aws:ResourceTag/ \${TagKey} (p. 706)

Condition Keys for AWS Amplify

AWS Amplify defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}		String
aws:ResourceTag/\${TagKey}		String
aws:TagKeys		String

Actions, Resources, and Condition Keys for Amazon API Gateway

Amazon API Gateway (service prefix: `execute-api`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon API Gateway \(p. 706\)](#)
- [Resource Types Defined by Amazon API Gateway \(p. 707\)](#)
- [Condition Keys for Amazon API Gateway \(p. 707\)](#)

Actions Defined by Amazon API Gateway

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
InvalidateCache	Used to invalidate API cache upon a client request	Write	execute-api-		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			general* (p. 707)		
Invoke	Used to invoke an API upon a client request	Write	execute-api-general* (p. 707)		
ManageConnections	ManageConnections controls access to the @connections API	Write	execute-api-general* (p. 707)		

Resource Types Defined by Amazon API Gateway

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 706\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
execute-api-general	arn:\${Partition}:execute-api:\${Region}: \${Account}:[\${ApiId}]/[\${Stage}]/[\${Method}]/ \${ApiSpecificResourcePath}	

Condition Keys for Amazon API Gateway

ExecuteAPI has no service-specific context keys that can be used in the Condition element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the [IAM Policy Reference](#).

Actions, Resources, and Condition Keys for AWS App Mesh

AWS App Mesh (service prefix: appmesh) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS App Mesh \(p. 708\)](#)
- [Resource Types Defined by AWS App Mesh \(p. 711\)](#)
- [Condition Keys for AWS App Mesh \(p. 712\)](#)

Actions Defined by AWS App Mesh

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateMesh	Creates a service mesh.	Write	mesh* (p. 712)		
				aws:TagKeys (p. 712)	
				aws:RequestTag/ {\$TagKey} (p. 712)	
CreateRoute	Creates a route that is associated with a virtual router.	Write	route* (p. 712)		
			virtualNode (p. 712)		
				aws:TagKeys (p. 712)	
CreateVirtualNode	Creates a virtual node within a service mesh.	Write	virtualNode* (p. 712)		
			virtualService (p. 712)		
				aws:TagKeys (p. 712)	
				aws:RequestTag/ {\$TagKey} (p. 712)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateVirtualRouter	Creates a virtual router within a service mesh.	Write	virtualRouter* (p. 712)		
				aws:TagKeys (p. 712)	
				aws:RequestTag/ \${TagKey} (p. 712)	
CreateVirtualService	Creates a virtual service within a service mesh.	Write	virtualService* (p. 712)		
			virtualNode (p. 712)		
			virtualRouter (p. 712)		
				aws:TagKeys (p. 712)	
				aws:RequestTag/ \${TagKey} (p. 712)	
DeleteMesh	Deletes an existing service mesh.	Write	mesh* (p. 712)		
DeleteRoute	Deletes an existing route.	Write	route* (p. 712)		
DeleteVirtualNode	Deletes an existing virtual node.	Write	virtualNode* (p. 712)		
DeleteVirtualRouter	Deletes an existing virtual router.	Write	virtualRouter* (p. 712)		
DeleteVirtualService	Deletes an existing virtual service.	Write	virtualService* (p. 712)		
DescribeMesh	Describes an existing service mesh.	Read	mesh* (p. 712)		
DescribeRoute	Describes an existing route.	Read	route* (p. 712)		
DescribeVirtualNode	Describes an existing virtual node.	Read	virtualNode* (p. 712)		
DescribeVirtualRouter	Describes an existing virtual router.	Read	virtualRouter* (p. 712)		
DescribeVirtualService	Describes an existing virtual service.	Read	virtualService* (p. 712)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListMeshes	Returns a list of existing service meshes.	List			
ListRoutes	Returns a list of existing routes in a service mesh.	List	virtualRouter* (p. 712)		
ListTagsForResource	List the tags for an App Mesh resource.	List	mesh (p. 712)		
			route (p. 712)		
			virtualNode (p. 712)		
			virtualRouter (p. 712)		
			virtualService (p. 712)		
ListVirtualNodes	Returns a list of existing virtual nodes.	List	mesh* (p. 712)		
ListVirtualRouters	Returns a list of existing virtual routers in a service mesh.	List	virtualRouter* (p. 712)		
ListVirtualServices	Returns a list of existing virtual services in a service mesh.	List	virtualService* (p. 712)		
StreamAggregatedResources	Allows an Envoy Proxy to receive streamed resources for a VirtualNode.	Read	virtualNode* (p. 712)		
TagResource	Associates the specified tags to a resource with the specified resourceArn.	Write	mesh (p. 712)		
				route (p. 712)	
			virtualNode (p. 712)		
			virtualRouter (p. 712)		
			virtualService (p. 712)		
				aws:TagKeys (p. 712)	
					aws:RequestTag/\${TagKey} (p. 712)

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UntagResource	Deletes specified tags from a resource.	Write	mesh (p. 712)		
			route (p. 712)		
			virtualNode (p. 712)		
			virtualRouter (p. 712)		
			virtualService (p. 712)		
				aws:TagKeys (p. 712)	
UpdateMesh	Updates an existing service mesh.	Write	mesh* (p. 712)		
UpdateRoute	Updates an existing route for a specified service mesh and virtual router.	Write	route* (p. 712)		
			virtualNode (p. 712)		
UpdateVirtualNode	Updates an existing virtual node in a specified service mesh.	Write	virtualNode* (p. 712)		
UpdateVirtualRouter	Updates an existing virtual router in a specified service mesh.	Write	virtualRouter* (p. 712)		
UpdateVirtualService	Updates an existing virtual service in a specified service mesh.	Write	mesh* (p. 712)		
			virtualNode (p. 712)		
			virtualRouter (p. 712)		

Resource Types Defined by AWS App Mesh

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 708\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
mesh	arn:\${Partition}:appmesh:\${Region}: \${Account}:mesh/\${MeshName}	aws:ResourceTag/\${TagKey} (p. 712)
virtualService	arn:\${Partition}:appmesh:\${Region}: \${Account}:mesh/\${MeshName}/virtualService/ \${VirtualServiceName}	aws:ResourceTag/\${TagKey} (p. 712)
virtualNode	arn:\${Partition}:appmesh:\${Region}: \${Account}:mesh/\${MeshName}/virtualNode/ \${VirtualNodeName}	aws:ResourceTag/\${TagKey} (p. 712)
virtualRouter	arn:\${Partition}:appmesh:\${Region}: \${Account}:mesh/\${MeshName}/virtualRouter/ \${VirtualRouterName}	aws:ResourceTag/\${TagKey} (p. 712)
route	arn:\${Partition}:appmesh:\${Region}: \${Account}:mesh/\${MeshName}/virtualRouter/ \${VirtualRouterName}/route/\${RouteName}	aws:ResourceTag/\${TagKey} (p. 712)

Condition Keys for AWS App Mesh

AWS App Mesh defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Filters actions based on the presence of tag key-value pairs in the request.	String
aws:ResourceTag/\${TagKey}	Filters actions based on tag key-value pairs attached to the resource.	String
aws:TagKeys	Filters actions based on the presence of tag keys in the request.	String

Actions, Resources, and Condition Keys for AWS App Mesh Preview

AWS App Mesh Preview (service prefix: `appmesh-preview`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS App Mesh Preview \(p. 713\)](#)
- [Resource Types Defined by AWS App Mesh Preview \(p. 715\)](#)
- [Condition Keys for AWS App Mesh Preview \(p. 715\)](#)

Actions Defined by AWS App Mesh Preview

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateMesh	Creates a service mesh.	Write	mesh* (p. 715)		
CreateRoute	Creates a route that is associated with a virtual router.	Write	route* (p. 715)		
			virtualNode (p. 715)		
CreateVirtualNode	Creates a virtual node within a service mesh.	Write	virtualNode* (p. 715)		
			virtualService (p. 715)		
CreateVirtualRouter	Creates a virtual router within a service mesh.	Write	virtualRouter* (p. 715)		
			virtualService* (p. 715)		
CreateVirtualService	Creates a virtual service within a service mesh.	Write	virtualNode (p. 715)		
			virtualRouter (p. 715)		
			mesh* (p. 715)		
DeleteMesh	Deletes an existing service mesh.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteRoute	Deletes an existing route.	Write	route* (p. 715)		
DeleteVirtualNode	Deletes an existing virtual node.	Write	virtualNode* (p. 715)		
DeleteVirtualRouter	Deletes an existing virtual router.	Write	virtualRouter* (p. 715)		
DeleteVirtualService	Deletes an existing virtual service.	Write	virtualService* (p. 715)		
DescribeMesh	Describes an existing service mesh.	Read	mesh* (p. 715)		
DescribeRoute	Describes an existing route.	Read	route* (p. 715)		
DescribeVirtualNode	Describes an existing virtual node.	Read	virtualNode* (p. 715)		
DescribeVirtualRouter	Describes an existing virtual router.	Read	virtualRouter* (p. 715)		
DescribeVirtualService	Describes an existing virtual service.	Read	virtualService* (p. 715)		
ListMeshes	Returns a list of existing service meshes.	List			
ListRoutes	Returns a list of existing routes in a service mesh.	List	virtualRouter* (p. 715)		
ListVirtualNodes	Returns a list of existing virtual nodes.	List	mesh* (p. 715)		
ListVirtualRouters	Returns a list of existing virtual routers in a service mesh.	List	virtualRouter* (p. 715)		
ListVirtualServices	Returns a list of existing virtual services in a service mesh.	List	virtualService* (p. 715)		
StreamAggregatedResources	Allows an Envoy Proxy to receive streamed resources for a VirtualNode.	Read	virtualNode* (p. 715)		
UpdateMesh	Updates an existing service mesh.	Write	mesh* (p. 715)		
UpdateRoute	Updates an existing route for a specified service mesh and virtual router.	Write	route* (p. 715)		
			virtualNode (p. 715)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UpdateVirtualNode	Updates an existing virtual node in a specified service mesh.	Write	virtualNode* (p. 715)		
UpdateVirtualRouter	Updates an existing virtual router in a specified service mesh.	Write	virtualRouter* (p. 715)		
UpdateVirtualService	Updates an existing virtual service in a specified service mesh.	Write	mesh* (p. 715)		
			virtualNode (p. 715)		
			virtualRouter (p. 715)		

Resource Types Defined by AWS App Mesh Preview

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 713\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
mesh	arn:\${Partition}:appmesh-preview:\${Region}: \${Account}:mesh/\${MeshName}	
virtualService	arn:\${Partition}:appmesh-preview:\${Region}: \${Account}:mesh/\${MeshName}/virtualService/ \${VirtualServiceName}	
virtualNode	arn:\${Partition}:appmesh-preview:\${Region}: \${Account}:mesh/\${MeshName}/virtualNode/ \${VirtualNodeName}	
virtualRouter	arn:\${Partition}:appmesh-preview:\${Region}: \${Account}:mesh/\${MeshName}/virtualRouter/ \${VirtualRouterName}	
route	arn:\${Partition}:appmesh-preview:\${Region}: \${Account}:mesh/\${MeshName}/virtualRouter/ \${VirtualRouterName}/route/\${RouteName}	

Condition Keys for AWS App Mesh Preview

App Mesh Preview has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Application Auto Scaling

Application Auto Scaling (service prefix: `application-autoscaling`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Application Auto Scaling \(p. 716\)](#)
- [Resource Types Defined by Application Auto Scaling \(p. 717\)](#)
- [Condition Keys for Application Auto Scaling \(p. 717\)](#)

Actions Defined by Application Auto Scaling

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
<code>DeleteScalingPolicy</code>	Deletes an Application Auto Scaling scaling policy that was previously created.	Write			
<code>DeleteScheduledAction</code>	Deletes an Application Auto Scaling scheduled action that was previously created.	Write			
<code>DeregisterScalableTarget</code>	Deregisters a scalable target that was previously registered.	Write			
<code>DescribeScalableTargets</code>	Provides descriptive information for scalable targets with a specified service namespace.	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeScalingActivities	Provides descriptive information for scaling activities with a specified service namespace for the previous six weeks.	Read			
DescribeScalingPolicies	Provides descriptive information for scaling policies with a specified service namespace.	Read			
DescribeScheduledActions	Provides descriptive information for scheduled actions with a specified service namespace.	Read			
PutScalingPolicy	Creates or updates a policy for an existing Application Auto Scaling scalable target.	Write			
PutScheduledAction	Creates or updates a scheduled action for an existing Application Auto Scaling scalable target.	Write			
RegisterScalableTarget	Registers or updates a scalable target. A scalable target is a resource that can be scaled out or in with Application Auto Scaling.	Write			

Resource Types Defined by Application Auto Scaling

Application Auto Scaling does not support specifying a resource ARN in the `Resource` element of an IAM policy statement. To allow access to Application Auto Scaling, specify “`Resource`”: “`*`” in your policy.

Condition Keys for Application Auto Scaling

Application Auto Scaling has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Application Discovery

Application Discovery (service prefix: `discovery`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Application Discovery \(p. 718\)](#)
- [Resource Types Defined by Application Discovery \(p. 721\)](#)
- [Condition Keys for Application Discovery \(p. 721\)](#)

Actions Defined by Application Discovery

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AssociateConfigurationItemsWithApplication	Associates one or more configuration items with an application.	Write			
BatchDeleteImportTasks	Deletes one or more Migration Hub import tasks, each identified by their import ID. Each import task has a number of records, which can identify servers or applications.	Write			
CreateApplication	Creates an application with the given name and description.	Write			
CreateTags	Creates one or more tags for configuration items. Tags are metadata that help you categorize IT assets. This API accepts a list of multiple configuration items.	Tagging			
DeleteApplication	Deletes a list of applications and their associations with configuration items.	Write			
DeleteTags	Deletes the association between configuration items and one or more tags. This API accepts a list of multiple configuration items.	Tagging			
DescribeAgents	Lists agents or the Connector by ID or lists all agents/Connectors	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
	associated with your user account if you did not specify an ID.				
DescribeConfigurationItems	Retrieves attributes for a list of configuration item IDs. All of the supplied IDs must be for the same asset type (server, application, process, or connection). Output fields are specific to the asset type selected. For example, the output for a server configuration item includes a list of attributes about the server, such as host name, operating system, and number of network cards.	Read			
DescribeContinuousExports	Lists exports as specified by <code>DescribeContinuousExports</code> . Continuous exports associated with your user account can be listed if you call <code>DescribeContinuousExports</code> as is without passing any parameters.	Read			
DescribeExportProcesses	Retrieves the status of a given export process. You can retrieve status from a maximum of 100 processes.	Read			
DescribeExportTasks	Retrieve status of one or more export tasks. You can retrieve the status of up to 100 export tasks.	Read			
DescribeImportTasks	Returns an array of import tasks for your account, including status information, times, IDs, the Amazon S3 Object URL for the import file, and more.	List			
DescribeTags	Retrieves a list of configuration items that are tagged with a specific tag. Or retrieves a list of all tags assigned to a specific configuration item.	Read			
DisassociateConfigurationItemsFromApplication	Disassociates one or more configuration items from an application.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ExportConfiguration	Exports all discovered configuration data to an Amazon S3 bucket or an application that enables you to view and evaluate the data. Data includes tags and tag associations, processes, connections, servers, and system performance.	Write			
GetDiscoverySummary	Retrieves a short summary of discovered assets.	Read			
ListConfigurations	Retrieves a list of configuration items according to criteria you specify in a filter. The filter criteria identify relationship requirements.	List			
ListServerNeighbors	Retrieves a list of servers which are one network hop away from a specified server.	List			
StartContinuousExport	Start the continuous flow of Agent's discovered data into Amazon Athena.	Write			
StartDataCollection	Instructs the specified agents or Connectors to start collecting data.	Write			
StartExportTask	Export the configuration data about discovered configuration items and relationships to an S3 bucket in a specified format.	Write			
StartImportTask	Starts an import task. The Migration Hub import feature allows you to import details of your on-premises environment directly into AWS without having to use the Application Discovery Service (ADS) tools such as the Discovery Connector or Discovery Agent. This gives you the option to perform migration assessment and planning directly from your imported data including the ability to group your devices as applications and track their migration status.	Write			
StopContinuousExport	Stop the continuous flow of Agent's discovered data into Amazon Athena.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
StopDataCollection	Instructs the specified agents or connectors to stop collecting data.	Write			
UpdateApplication	Updates metadata about an application.	Write			

Resource Types Defined by Application Discovery

Application Discovery does not support specifying a resource ARN in the Resource element of an IAM policy statement. To allow access to Application Discovery, specify “Resource”: “*” in your policy.

Condition Keys for Application Discovery

Application Discovery has no service-specific context keys that can be used in the Condition element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Application Discovery Arsenal

Application Discovery Arsenal (service prefix: `arsenal`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Application Discovery Arsenal \(p. 721\)](#)
- [Resource Types Defined by Application Discovery Arsenal \(p. 722\)](#)
- [Condition Keys for Application Discovery Arsenal \(p. 722\)](#)

Actions Defined by Application Discovery Arsenal

You can specify the following actions in the Action element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources (“*”) in the Resource element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
RegisterOnPremisesAWSpro [permission only]	Grants permission to register AWS-provided data collectors to the Application Discovery Service	Write			

Resource Types Defined by Application Discovery Arsenal

Application Discovery Arsenal does not support specifying a resource ARN in the Resource element of an IAM policy statement. To allow access to Application Discovery Arsenal, specify "Resource": "*" in your policy.

Condition Keys for Application Discovery Arsenal

Application Discovery Arsenal has no service-specific context keys that can be used in the Condition element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Amazon AppStream 2.0

Amazon AppStream 2.0 (service prefix: `appstream`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon AppStream 2.0 \(p. 722\)](#)
- [Resource Types Defined by Amazon AppStream 2.0 \(p. 730\)](#)
- [Condition Keys for Amazon AppStream 2.0 \(p. 730\)](#)

Actions Defined by Amazon AppStream 2.0

You can specify the following actions in the Action element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the Resource element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you

specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AssociateFleet	Grants permission to associate the specified fleet with the specified stack	Write	fleet* (p. 730)		
			stack* (p. 730)		
				aws:ResourceTag/ \${TagKey} (p. 731)	
BatchAssociateUsersWithStacks	Grants permission to associate the specified users with the specified stacks. Users in a user pool cannot be assigned to stacks with fleets that are joined to an Active Directory domain	Write	stack* (p. 730)		
				aws:ResourceTag/ \${TagKey} (p. 731)	
BatchDisassociateUsersFromStacks	Grants permission to disassociate the specified users from the specified stacks	Write	stack* (p. 730)		
				aws:ResourceTag/ \${TagKey} (p. 731)	
CopyImage	Grants permission to copy the specified image within the same Region or to a new Region within the same AWS account	Write	image* (p. 730)		
				aws:ResourceTag/ \${TagKey} (p. 731)	
.CreateDirectoryConfig	Grants permission to create a Directory Config object in AppStream 2.0. This object includes the configuration information required to join fleets and image builders to Microsoft Active Directory domains	Write			
CreateFleet	Grants permission to create a fleet. A fleet is a group of streaming instances from which applications are launched and streamed to users	Write	fleet* (p. 730)		
			image* (p. 730)		
				aws:RequestTag/ \${TagKey} (p. 731)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:TagKeys (p. 731)	
CreateImageBuilder	Grants permission to create an image builder. An image builder is a virtual machine that is used to create an image	Write	image* (p. 730)		
			image-builder* (p. 730)		
				aws:RequestTag/ \${TagKey} (p. 731)	aws:TagKeys (p. 731)
CreateImageBuilderURLToStartStreamingSession	Grants permission to create a URL to start an image builder streaming session	Write	image-builder* (p. 730)		
				aws:ResourceTag/ \${TagKey} (p. 731)	
CreateStack	Grants permission to create a stack to start streaming applications to users. A stack consists of an associated fleet, user access policies, and storage configurations	Write	stack* (p. 730)		
				aws:RequestTag/ \${TagKey} (p. 731)	
				aws:TagKeys (p. 731)	
CreateStreamingURLRequest	Grants permission to create a temporary URL to start an AppStream 2.0 streaming session for the specified user. A streaming URL enables application streaming to be tested without user setup	Write	fleet* (p. 730)		
			stack* (p. 730)		
				aws:ResourceTag/ \${TagKey} (p. 731)	
CreateUsageReportSubscription	Grants permission to create a usage report subscription. Usage reports are generated daily	Write			
CreateUser	Grants permission to create a new user in the user pool	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteDirectoryConfig	Grants permission to delete the specified Directory Config object from AppStream 2.0. This object includes the configuration information required to join fleets and image builders to Microsoft Active Directory domains	Write			
DeleteFleet	Grants permission to delete the specified fleet	Write	fleet* (p. 730)		
				aws:ResourceTag/ {\$TagKey} (p. 731)	
DeleteImage	Grants permission to delete the specified image. An image cannot be deleted when it is in use	Write	image* (p. 730)		
				aws:ResourceTag/ {\$TagKey} (p. 731)	
DeleteImageBuilder	Grants permission to delete the specified image builder and release capacity	Write	image-builder* (p. 730)		
				aws:ResourceTag/ {\$TagKey} (p. 731)	
DeleteImagePermissions	Grants permission to delete permissions for the specified private image	Write	image* (p. 730)		
				aws:ResourceTag/ {\$TagKey} (p. 731)	
DeleteStack	Grants permission to delete the specified stack. After the stack is deleted, the application streaming environment provided by the stack is no longer available to users. Also, any reservations made for application streaming sessions for the stack are released	Write	stack* (p. 730)		
				aws:ResourceTag/ {\$TagKey} (p. 731)	
DeleteUsageReport	Grants permission to disable usage report generation	Write			
DeleteUser	Grants permission to delete a user from the user pool	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeDirectories	Grants permission to retrieve a list that describes one or more specified Directory Config objects for AppStream 2.0, if the names for these objects are provided. Otherwise, all Directory Config objects in the account are described. This object includes the configuration information required to join fleets and image builders to Microsoft Active Directory domains	Read			
DescribeFleets	Grants permission to retrieve a list that describes one or more specified fleets, if the fleet names are provided. Otherwise, all fleets in the account are described	Read	fleet (p. 730)		
DescribeImageBuilders	Grants permission to retrieve a list that describes one or more specified image builders, if the image builder names are provided. Otherwise, all image builders in the account are described	Read	image-builder (p. 730)		
DescribeImagePermissions	Grants permission to retrieve a list that describes the permissions for shared AWS account IDs on a private image that you own	Read	image* (p. 730)		
DescribeImages	Grants permission to retrieve a list that describes one or more specified images, if the image names or image ARNs are provided. Otherwise, all images in the account are described	Read	image (p. 730)		
DescribeSessions	Grants permission to retrieve a list that describes the streaming sessions for the specified stack and fleet. If a user ID is provided for the stack and fleet, only the streaming sessions for that user are described	Read	fleet* (p. 730)		
	stack* (p. 730)				

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeStacks	Grants permission to retrieve a list that describes one or more specified stacks, if the stack names are provided. Otherwise, all stacks in the account are described	Read	stack (p. 730)		
DescribeUsageReportSubscriptions	Grants permission to retrieve a list that describes one or more usage report subscriptions	Read			
DescribeUserStackAssociations	Grants permission to retrieve a list that describes the UserStackAssociation objects	Read	stack (p. 730)		
DescribeUsers	Grants permission to retrieve a list that describes users in the user pool	Read			
DisableUser	Grants permission to disable the specified user in the user pool. This action does not delete the user	Write			
DisassociateFleet	Grants permission to disassociate the specified fleet from the specified stack	Write	fleet* (p. 730)		
			stack* (p. 730)		
				aws:ResourceTag/\${TagKey} (p. 731)	
EnableUser	Grants permission to enable a user in the user pool	Write			
ExpireSession	Grants permission to immediately stop the specified streaming session	Write			
GetImageBuilders [permission only]	Grants permission to retrieve a list that describes one or more specified image builders, if the image builder names are provided. Otherwise, all image builders in the account are described	Read			
GetParametersForAssetUpload [permission only]	Grants permission to upload the asset	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListAssociatedFleets	Grants permission to retrieve the name of the fleet that is associated with the specified stack	Read	stack* (p. 730)		
ListAssociatedStacks	Grants permission to retrieve the name of the stack with which the specified fleet is associated	Read	fleet* (p. 730)		
ListTagsForResource	Grants permission to retrieve a list of all tags for the specified AppStream 2.0 resource. The following resources can be tagged: Image builders, images, fleets, and stacks	Read			
StartFleet	Grants permission to start the specified fleet	Write	fleet* (p. 730)		
				aws:ResourceTag/ \${TagKey} (p. 731)	
StartImageBuilder	Grants permission to start the specified image builder	Write	image-builder* (p. 730)		
				aws:ResourceTag/ \${TagKey} (p. 731)	
StopFleet	Grants permission to stop the specified fleet	Write	fleet* (p. 730)		
				aws:ResourceTag/ \${TagKey} (p. 731)	
StopImageBuilder	Grants permission to stop the specified image builder	Write	image-builder* (p. 730)		
				aws:ResourceTag/ \${TagKey} (p. 731)	
Stream	Grants permission to federated users to sign in by using their existing credentials and stream applications from the specified stack	Write	stack* (p. 730)		
				appstream:userId (p. 731)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
TagResource	Grants permission to add or overwrite one or more tags for the specified AppStream 2.0 resource. The following resources can be tagged: Image builders, images, fleets, and stacks	Tagging	fleet (p. 730)		
	image (p. 730)				
	image-builder (p. 730)				
	stack (p. 730)				
			aws:RequestTag/\${TagKey} (p. 731)		
UntagResource	Grants permission to disassociate one or more tags from the specified AppStream 2.0 resource	Tagging	fleet (p. 730)		
	image (p. 730)				
	image-builder (p. 730)				
	stack (p. 730)				
			aws:TagKeys (p. 731)		
UpdateDirectory	Grants permission to update the specified Directory Config object in AppStream 2.0. This object includes the configuration information required to join fleets and image builders to Microsoft Active Directory domains	Write			
	fleet* (p. 730)				
UpdateFleet	Grants permission to update the specified fleet. All attributes except the fleet name can be updated when the fleet is in the STOPPED state	Write	image (p. 730)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:ResourceTag/ \${TagKey} (p. 731)	
UpdateImagePermissions	Grants permission to add or update permissions for the specified private image	Write	image*	(p. 730)	
				aws:ResourceTag/ \${TagKey} (p. 731)	
UpdateStack	Grants permission to update the specified fields for the specified stack	Write	stack*	(p. 730)	
				aws:ResourceTag/ \${TagKey} (p. 731)	

Resource Types Defined by Amazon AppStream 2.0

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 722\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
fleet	arn:\${Partition}:appstream:\${Region}: \${Account}:fleet/\${FleetName}	aws:ResourceTag/ \${TagKey} (p. 731)
image	arn:\${Partition}:appstream:\${Region}: \${Account}:image/\${ImageName}	aws:ResourceTag/ \${TagKey} (p. 731)
image-builder	arn:\${Partition}:appstream:\${Region}: \${Account}:image-builder/\${ImageBuilderName}	aws:ResourceTag/ \${TagKey} (p. 731)
stack	arn:\${Partition}:appstream:\${Region}: \${Account}:stack/\${StackName}	aws:ResourceTag/ \${TagKey} (p. 731)

Condition Keys for Amazon AppStream 2.0

Amazon AppStream 2.0 defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
appstream:userId	Filters access by the ID of the AppStream 2.0 user	String
aws:RequestTag/\${TagKey}	Filters actions based on the presence of tag key-value pairs in the request	String
aws:ResourceTag/\${TagKey}	Filters actions based on tag key-value pairs attached to the resource	String
aws:TagKeys	Filters actions based on the presence of tag keys in the request	String

Actions, Resources, and Condition Keys for AWS AppSync

AWS AppSync (service prefix: `appsync`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS AppSync \(p. 731\)](#)
- [Resource Types Defined by AWS AppSync \(p. 734\)](#)
- [Condition Keys for AWS AppSync \(p. 735\)](#)

Actions Defined by AWS AppSync

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateApiKey	Creates a unique key that you can distribute to clients who are executing your API.	Write			
CreateDataSource	Creates a DataSource object.	Write			
CreateFunction	Create a new Function object.	Write			
CreateGraphqlApi	Creates a GraphqlApi object, which is the top level AppSync resource.	Tagging		aws:RequestTag/\${TagKey} (p. 735) aws:TagKeys (p. 735)	
CreateResolver	Creates a Resolver object. A resolver converts incoming requests into a format that a data source can understand, and converts the data source's responses into GraphQL.	Write			
CreateType	Creates a Type object.	Write			
DeleteApiKey	Deletes an API key.	Write			
DeleteDataSource	Deletes a DataSource object.	Write			
DeleteFunction	Deletes a Function object.	Write			
DeleteGraphqlApi	Deletes a GraphqlApi object. This will also clean up every AppSync resource below that API.	Write	graphqlapi* (p. 735)		
				aws:ResourceTag/\${TagKey} (p. 735)	
DeleteResolver	Deletes a Resolver object.	Write			
DeleteType	Deletes a Type object.	Write			
GetDataSource	Retrieves a DataSource object.	Read			
GetFunction	Retrieves a Function object.	Read			
GetGraphqlApi	Retrieves a GraphqlApi object.	Read	graphqlapi* (p. 735)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:ResourceTag/ \${TagKey} (p. 735)	
GetIntrospectionSchema	Retrieves the introspection schema for a GraphQL API.	Read			
GetResolver	Retrieves a Resolver object.	Read			
GetSchemaCreationStatus	Retrieves the current status of a schema creation operation.	Read			
GetType	Retrieves a Type object.	Read			
GraphQL	Sends a GraphQL query to a GraphQL API.	Write	field* (p. 735)		
			graphqlapi* (p. 735)		
ListApiKeys	Lists the API keys for a given API.	List			
ListDataSources	Lists the data sources for a given API.	List			
ListFunctions	Lists the functions for a given API.	List			
ListGraphqlApis	Lists your GraphQL APIs.	List			
ListResolvers	Lists the resolvers for a given API and type.	List			
ListResolversByFunction	List the resolvers that are associated with a specific function.	List			
ListTagsForResource	List the tags for a resource.	Read	graphqlapi (p. 735)		
				aws:ResourceTag/ \${TagKey} (p. 735)	
ListTypes	Lists the types for a given API.	List			
StartSchemaCreation	Adds a new schema to your GraphQL API. This operation is asynchronous - GetSchemaCreationStatus can show when it has completed.	Write			
TagResource	Tag a resource.	Tagging	graphqlapi (p. 735)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:RequestTag/\${TagKey} (p. 735) aws:ResourceTag/\${TagKey} (p. 735) aws:TagKeys (p. 735)	
UntagResource	Untag a resource.	Tagging	graphqlapi (p. 735)		
				aws:TagKeys (p. 735)	
UpdateApiKey	Updates an API key for a given API.	Write			
UpdateDataSource	Updates a DataSource object.	Write			
UpdateFunction	Updates an existing Function object.	Write			
UpdateGraphqlApi	Updates a GraphqlApi object.	Write	graphqlapi* (p. 735)		
				aws:ResourceTag/\${TagKey} (p. 735)	
UpdateResolver	Updates a Resolver object.	Write			
UpdateType	Updates a Type object.	Write			

Resource Types Defined by AWS AppSync

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 731\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
datasource	<code>arn:\${Partition}:appsync:\${Region}:\${Account}:apis/\${GraphQLAPIId}/datasources/\${DatasourceName}</code>	

Resource Types	ARN	Condition Keys
graphqlapi	arn:\${Partition}:appsync:\${Region}: \${Account}:apis/\${GraphQLAPIId}	aws:ResourceTag/\${TagKey} (p. 735)
field	arn:\${Partition}:appsync:\${Region}: \${Account}:apis/\${GraphQLAPIId}/types/ \${TypeName}/fields/\${FieldName}	
type	arn:\${Partition}:appsync:\${Region}: \${Account}:apis/\${GraphQLAPIId}/types/ \${TypeName}	
function	arn:\${Partition}:appsync:\${Region}: \${Account}:apis/\${GraphQLAPIId}/functions/ \${FunctionId}	

Condition Keys for AWS AppSync

AWS AppSync defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Filters actions based on the presence of tag key-value pairs in the request	String
aws:ResourceTag/\${TagKey}	Filters actions based on tag key-value pairs attached to the resource	String
aws:TagKeys	Filters actions based on the presence of tag keys in the request	String

Actions, Resources, and Condition Keys for AWS Artifact

AWS Artifact (service prefix: `artifact`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Artifact \(p. 736\)](#)
- [Resource Types Defined by AWS Artifact \(p. 736\)](#)
- [Condition Keys for AWS Artifact \(p. 737\)](#)

Actions Defined by AWS Artifact

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AcceptAgreement	Grants permission to accept an AWS agreement that has not yet been accepted by the customer account.	Write	agreement* (p. 737)		
DownloadAgreement	Grants permission to download an AWS agreement that has not yet been accepted or a customer agreement that has been accepted by the customer account.	Read	agreement (p. 737)		
			customer-agreement (p. 737)		
Get	Grants permission to download an AWS compliance report package.	Read	report-package* (p. 736)		
TerminateAgreement	Grants permission to terminate a customer agreement that was previously accepted by the customer account.	Write	customer-agreement* (p. 737)		

Resource Types Defined by AWS Artifact

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 736\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
report-package	<code>arn:\${Partition}:artifact:::report-package/*</code>	

Resource Types	ARN	Condition Keys
customer-agreement	arn:\${Partition}:artifact::\${Account}:customer-agreement/*	
agreement	arn:\${Partition}:artifact:::agreement/*	

Condition Keys for AWS Artifact

Artifact has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Amazon Athena

Amazon Athena (service prefix: `athena`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Athena \(p. 737\)](#)
- [Resource Types Defined by Amazon Athena \(p. 740\)](#)
- [Condition Keys for Amazon Athena \(p. 741\)](#)

Actions Defined by Amazon Athena

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
BatchGetNamedQueryInformation	Grants permissions to get information about one or more named queries.	Read	workgroup* (p. 741)		
BatchGetQueryExecutionInformation	Grants permissions to get information about one or more query executions.	Read	workgroup* (p. 741)		
CancelQueryExecution	Deprecated. Applies only to AWS services and principals that use Athena JDBC driver earlier than 1.1.0. Use StopQueryExecution otherwise.	Write	workgroup* (p. 741)		
CreateNamedQuery	Grants permissions to create a named query.	Write	workgroup* (p. 741)		
CreateWorkGroup	Grants permissions to create a workgroup.	Tagging	workgroup* (p. 741)		
			aws:RequestTag/ {\$TagKey} (p. 741)	aws:TagKeys (p. 741)	
DeleteNamedQuery	Grants permissions to delete a named query specified.	Write	workgroup* (p. 741)		
DeleteWorkGroup	Grants permissions to delete a workgroup.	Write	workgroup* (p. 741)		
GetCatalogs	Applies only to AWS services managed policy and principals that use an Athena JDBC driver version 1.1.0. Grants permissions to enable access to databases and tables.	Read			
GetExecutionEngine	Applies only to AWS services managed policy and principals that use an Athena JDBC driver version 1.1.0. Grants permissions to enable access to the specified database and table.	Read			
GetExecutionEngine	Applies only to AWS services managed policy and principals that use an Athena JDBC driver version 1.1.0. Grants permissions to enable access to databases and tables.	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetNamedQuery	Grants permissions to get information about the specified named query.	Read	workgroup* (p. 741)		
GetNamespace	Applies only to AWS services managed policy and principals that use an Athena JDBC driver version 1.1.0. Grants permissions to enable access to the specified database and table.	Read			
GetNamespaces	Applies only to AWS services managed policy and principals that use an Athena JDBC driver version 1.1.0. Grants permissions to enable access to databases and tables.	Read			
GetQueryExecution	Grants permissions to get information about the specified query execution.	Read	workgroup* (p. 741)		
GetQueryExecutions	Deprecated. Applies only to AWS services and principals that use Athena JDBC driver earlier than 1.1.0. Use ListQueryExecutions otherwise.	Read			
GetQueryResults	Grants permissions to get the query results.	Read	workgroup* (p. 741)		
GetQueryResultsStream	Grants permissions to get the query results stream.	Read	workgroup* (p. 741)		
GetTable	Applies only to AWS services managed policy and principals that use an Athena JDBC driver version 1.1.0. Grants permissions to enable access to the specified table.	Read			
GetTables	Applies only to AWS services managed policy and principals that use an Athena JDBC driver version 1.1.0. Grants permissions to enable access to tables.	Read			
GetWorkGroup	Grants permissions to get a workgroup.	Read	workgroup* (p. 741)		
ListNamedQueries	Grants permissions to return a list of named queries in Amazon Athena for the specified AWS account.	List	workgroup* (p. 741)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListQueryExecutions	Grants permissions to return a list of query executions for the specified AWS account.	List	workgroup* (p. 741)		
ListTagsForResource	Grants permissions to return a list of tags for a workgroup.	Read	workgroup* (p. 741)		
ListWorkGroups	Grants permissions to return a list of workgroups for the specified AWS account.	List			
RunQuery	Deprecated. Applies only to AWS services and principals that use Athena JDBC driver earlier than 1.1.0. Use StartQueryExecution otherwise.	Write			
StartQueryExecution	Grants permissions to start a query execution using an SQL query provided as a string.	Write	workgroup* (p. 741)		
StopQueryExecution	Grants permissions to stop the specified query execution.	Write	workgroup* (p. 741)		
TagResource	Grants permissions to add a tag to a workgroup.	Tagging	workgroup* (p. 741)		
			aws:RequestTag/ {\$TagKey} (p. 741)	aws:TagKeys (p. 741)	
UntagResource	Grants permissions to remove a tag from a workgroup.	Tagging	workgroup* (p. 741)		
				aws:TagKeys (p. 741)	
UpdateWorkGroup	Grants permissions to update a workgroup.	Write	workgroup* (p. 741)		

Resource Types Defined by Amazon Athena

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 737\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
workgroup	arn:\${Partition}:athena:\${Region}:\${Account}:workgroup/\${WorkGroupName}	aws:ResourceTag/\${TagKey} (p. 741)

Condition Keys for Amazon Athena

Amazon Athena defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
<code>aws:RequestTag/\${TagKey}</code>	Filters actions based on the presence of tag key-value pairs in the request	String
<code>aws:ResourceTag/\${TagKey}</code>	Filters actions based on tag key-value pairs attached to the resource	String
<code>aws:TagKeys</code>	Filters actions based on the presence of tag keys in the request	String

Actions, Resources, and Condition Keys for AWS Auto Scaling

AWS Auto Scaling (service prefix: `autoscaling-plans`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Auto Scaling \(p. 741\)](#)
- [Resource Types Defined by AWS Auto Scaling \(p. 742\)](#)
- [Condition Keys for AWS Auto Scaling \(p. 742\)](#)

Actions Defined by AWS Auto Scaling

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your

policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateScalingPlan	Creates a scaling plan.	Write			
DeleteScalingPlan	Deletes the specified scaling plan.	Write			
DescribeScalingPlans	Describes the scalable resources in the specified scaling plan.	Read			
DescribeScalingPlans	Describes the specified scaling plans or all of your scaling plans.	Read			
GetScalingPlanForecastData	Retrieves the forecast data for a scalable resource.	Read			
UpdateScalingPlan	Updates a scaling plan.	Write			

Resource Types Defined by AWS Auto Scaling

AWS Auto Scaling does not support specifying a resource ARN in the `Resource` element of an IAM policy statement. To allow access to AWS Auto Scaling, specify “`Resource`”: “`*`” in your policy.

Condition Keys for AWS Auto Scaling

Auto Scaling has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for AWS Backup

AWS Backup (service prefix: `backup`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Backup \(p. 743\)](#)
- [Resource Types Defined by AWS Backup \(p. 746\)](#)
- [Condition Keys for AWS Backup \(p. 746\)](#)

Actions Defined by AWS Backup

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CopyIntoBackupVault [permission only]	Copy into a backup vault	Write			
CreateBackupPlan	Creates a new backup plan	Write	backupPlan* (p. 746)		
			aws:RequestTag/\${TagKey} (p. 747)		
			aws:TagKeys (p. 747)		
CreateBackupSelectionAssignment	Creates a new resource assignment in a backup plan.	Write	backupPlan* (p. 746)		iam:PassRole
CreateBackupVault	Creates a new backup vault.	Write	backupVault* (p. 746)		
			aws:RequestTag/\${TagKey} (p. 747)		
			aws:TagKeys (p. 747)		
DeleteBackupPlan	Deletes a backup plan.	Write	backupPlan* (p. 746)		
DeleteBackupSelectionAssignment	Deletes a resource assignment from a backup plan.	Write	backupPlan* (p. 746)		
DeleteBackupVault	Deletes a backup vault.	Write	backupVault* (p. 746)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteBackupVaultAccessPolicy	Deletes backup vault access policy.	Write	backupVault* (p. 746)		
DeleteBackupVaultNotifications	Remove notifications from backup vaults.	Write	backupVault* (p. 746)		
DeleteRecoveryPoint	Deletes a recovery point from a backup vault.	Write	recoveryPoint* (p. 746)		
DescribeBackupJob	Describes a backup job	Read			
DescribeBackupVault	Creates a new backup vault with the specified name.	Read	backupVault* (p. 746)		
DescribeCopyJob	Describes a copy job	Read		aws:RequestTag/\${TagKey} (p. 747) aws:TagKeys (p. 747)	
DescribeProtectedResource	Describes a protected resource.	Read			
DescribeRecoveryPoint	Describes a recovery point.	Read	recoveryPoint* (p. 746)		
DescribeRestoreJob	Describes a restore job.	Read			
ExportBackupPlanJSON	Exports a backup plan as a JSON template.	Read			
GetBackupPlan	Gets a backup plan.	Read	backupPlan* (p. 746)		
GetBackupPlanFromJSON	Transforms a JSON to a backup plan.	Read			
GetBackupPlanFromTemplate	Transforms a template to a backup plan.	Read			
GetBackupSelectionAssignment	Gets a backup plan resource assignment.	Read	backupPlan* (p. 746)		
GetBackupVaultAccessPolicy	Gets backup vault access policy.	Read	backupVault* (p. 746)		
GetBackupVaultNotifications	Gets backup vault notifications.	Read	backupVault* (p. 746)		
GetRecoveryPointMetadata	Gets recovery point restore metadata.	Read	recoveryPoint* (p. 746)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetSupportedResourceTypes	Gets supported resource types.	Read			
ListBackupJobs	Lists backup jobs.	List			
ListBackupPlanTemplates	Lists backup plan templates provided by AWS Backup.	List			
ListBackupPlanVersions	Lists backup plan versions.	List	backupPlan* (p. 746)		
ListBackupPlans	Lists backup plans.	List			
ListBackupSelections	Lists resource assignments for a specific backup plan.	List	backupPlan* (p. 746)		
ListBackupVaults	Lists backup vaults.	List			
ListCopyJobs	List copy jobs	List			
ListProtectedResources	Lists protected resources by AWS Backup.	List			
ListRecoveryPointsByBackupVault	Lists recovery points inside a backup vault	List	backupVault* (p. 746)		
ListRecoveryPointsByResource	Lists recovery points for a resource	List			
ListRestoreJobs	Lists restore jobs.	List			
ListTags	Lists tags for a resource.	List			
PutBackupVaultAccessPolicy	Adds an access policy to the backup vault.	Write	backupVault* (p. 746)		
PutBackupVaultNotifications	Adds an SNS topic to the backup vault.	Write	backupVault* (p. 746)		
StartBackupJob	Starts a new backup job.	Write	backupVault* (p. 746)		iam:PassRole
StartCopyJob	Copy a backup from a source region to a destination region.	Write	recoveryPoint* (p. 746)		iam:PassRole
				aws:RequestTag/\${TagKey} (p. 747)	
				aws:TagKeys (p. 747)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
StartRestoreJob	Starts a new restore job.	Write	recoveryPoint* (p. 746)		iam:PassRole
StopBackupJob	Stops a backup job.	Write			
TagResource	Tags a resource.	Tagging		aws:RequestTag/\${TagKey} (p. 747) aws:TagKeys (p. 747)	
UntagResource	Untags a resource.	Tagging		aws:TagKeys (p. 747)	
UpdateBackupPlan	Updates a backup plan.	Write	backupPlan* (p. 746)		
UpdateRecoveryPoint	Updates the lifecycle of the recovery point.	Write	recoveryPoint* (p. 746)		

Resource Types Defined by AWS Backup

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 743\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
backupVault	<code>arn:\${Partition}:backup:\${Region}: \${Account}:backup-vault:\${BackupVaultName}</code>	
backupPlan	<code>arn:\${Partition}:backup:\${Region}: \${Account}:backup-plan:\${BackupPlanId}</code>	
recoveryPoint	<code>arn:\${Partition}: \${Vendor}: \${Region}:*: \${ResourceType}: \${RecoveryPointId}</code>	

Condition Keys for AWS Backup

AWS Backup defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the [IAM Policy Reference](#).

Condition Keys	Description	Type
<code>aws:RequestTag/\${TagKey}</code>	Filters actions based on the allowed set of values for each of the tags	String
<code>aws:TagKeys</code>	Filters actions based on the presence of mandatory tags in the request	String

Actions, Resources, and Condition Keys for AWS Backup storage

AWS Backup storage (service prefix: `backup-storage`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Backup storage \(p. 747\)](#)
- [Resource Types Defined by AWS Backup storage \(p. 748\)](#)
- [Condition Keys for AWS Backup storage \(p. 748\)](#)

Actions Defined by AWS Backup storage

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
<code>MountCapsule [permission only]</code>	Associates a KMS key to a backup vault	Write			

Resource Types Defined by AWS Backup storage

AWS Backup storage does not support specifying a resource ARN in the `Resource` element of an IAM policy statement. To allow access to AWS Backup storage, specify “`Resource`”: “`*`” in your policy.

Condition Keys for AWS Backup storage

Backup Storage has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for AWS Batch

AWS Batch (service prefix: `batch`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Batch \(p. 748\)](#)
- [Resource Types Defined by AWS Batch \(p. 750\)](#)
- [Condition Keys for AWS Batch \(p. 750\)](#)

Actions Defined by AWS Batch

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources (“`*`”) in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CancelJob	Cancels jobs in an AWS Batch job queue.	Write			
CreateComputeEnvironment	Creates an AWS Batch compute environment.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateJobQueue	Creates an AWS Batch job queue.	Write			
DeleteComputeEnvironment	Deletes an AWS Batch compute environment.	Write			
DeleteJobQueue	Deletes the specified job queue.	Write			
DeregisterJobDefinition	Deregisters an AWS Batch job definition.	Write	job-definition* (p. 750)		
DescribeComputeEnvironments	Describes one or more of your compute environments.	Read			
DescribeJobDefinitions	Describes a list of job definitions.	Read			
DescribeJobQueues	Describes one or more of your job queues.	Read			
DescribeJobs	Describes a list of AWS Batch jobs.	Read			
ListJobs	Returns a list of task jobs for a specified job queue.	List			
RegisterJobDefinition	Registers an AWS Batch job definition.	Write	job-definition* (p. 750)		
				batch:User (p. 750)	batch:Privileged (p. 750)
SubmitJob	Submits an AWS Batch job from a job definition.	Write	job-definition* (p. 750)		
				job-queue* (p. 750)	
TerminateJob	Terminates jobs in a job queue.	Write			
UpdateComputeEnvironment	Updates an AWS Batch compute environment.	Write			
UpdateJobQueue	Updates a job queue.	Write			

Resource Types Defined by AWS Batch

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 748\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
job-queue	arn:\${Partition}:batch:\${Region}: \${Account}:job-queue/\${JobQueueName}	
job-definition	arn:\${Partition}:batch:\${Region}: \${Account}:job-definition/ \${JobDefinitionName}:\${Revision}	

Condition Keys for AWS Batch

AWS Batch defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
batch:Image	The image used to start a container.	String
batch:Privileged	When this parameter is true, the container is given elevated privileges on the host container instance (similar to the root user).	Boolean
batch:User	The user name or numeric uid to use inside the container.	String

Actions, Resources, and Condition Keys for AWS Billing

AWS Billing (service prefix: aws-portal) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Billing \(p. 751\)](#)
- [Resource Types Defined by AWS Billing \(p. 751\)](#)

- [Condition Keys for AWS Billing \(p. 752\)](#)

Actions Defined by AWS Billing

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ModifyAccount	Allow or deny IAM users permission to modify Account Settings.	Write			
ModifyBilling	Allow or deny IAM users permission to modify billing settings.	Write			
ModifyPaymentMethod	Allow or deny IAM users permission to modify payment methods.	Write			
ViewAccount	Allow or deny IAM users permission to view account settings.	Read			
ViewBilling	Allow or deny IAM users permission to view billing pages in the console.	Read			
ViewPaymentMethod	Allow or deny IAM users permission to view payment methods.	Read			
ViewUsage	Allow or deny IAM users permission to view AWS usage reports.	Read			

Resource Types Defined by AWS Billing

AWS Billing does not support specifying a resource ARN in the `Resource` element of an IAM policy statement. To allow access to AWS Billing, specify “`Resource`”: “*” in your policy.

Condition Keys for AWS Billing

Billing has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for AWS Budget Service

AWS Budget Service (service prefix: `budgets`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Budget Service \(p. 752\)](#)
- [Resource Types Defined by AWS Budget Service \(p. 753\)](#)
- [Condition Keys for AWS Budget Service \(p. 753\)](#)

Actions Defined by AWS Budget Service

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Note

The actions in this table are not APIs, but are instead permissions that grant access to the AWS Billing and Cost Management APIs that access budgets.

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ModifyBudget	Modify budgets and budget details	Write	budget* (p. 753)		
ViewBudget	View budgets and budget details	Read	budget* (p. 753)		

Resource Types Defined by AWS Budget Service

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 752\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
budget	<code>arn:\${Partition}:budgets::\${Account}:budget/\${BudgetName}</code>	

Condition Keys for AWS Budget Service

Budget has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for AWS Certificate Manager

AWS Certificate Manager (service prefix: `acm`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Certificate Manager \(p. 753\)](#)
- [Resource Types Defined by AWS Certificate Manager \(p. 755\)](#)
- [Condition Keys for AWS Certificate Manager \(p. 755\)](#)

Actions Defined by AWS Certificate Manager

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AddTagsToCertificate	Adds one or more tags to a certificate.	Tagging	certificate* (p. 755)		
				aws:RequestTag/ {\$TagKey} (p. 755)	
				aws:TagKeys (p. 755)	
DeleteCertificate	Deletes a certificate and its associated private key.	Write	certificate* (p. 755)		
DescribeCertificate	Returns a list of the fields contained in the specified certificate.	Read	certificate* (p. 755)		
ExportCertificate	Exports a private certificate issued by a private certificate authority (CA) for use anywhere.	Read	certificate* (p. 755)		
GetCertificate	Retrieves a certificate and certificate chain for the certificate specified by an ARN.	Read	certificate* (p. 755)		
ImportCertificate	Imports a 3rd party SSL/TLS certificate into AWS Certificate Manager (ACM).	Write	certificate* (p. 755)		
				aws:RequestTag/ {\$TagKey} (p. 755)	
				aws:TagKeys (p. 755)	
ListCertificates	Retrieves a list of the certificate ARNs and the domain name for each ARN.	List			
ListTagsForCertificate	Lists the tags that have been applied to the certificate.	Read			
RemoveTagsFromCertificate	Remove one or more tags from a certificate. A tag consists of a key-value pair	Tagging	certificate* (p. 755)		
				aws:RequestTag/ {\$TagKey} (p. 755)	
				aws:TagKeys (p. 755)	
RenewCertificate	Renews an eligible private certificate.	Write	certificate* (p. 755)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
RequestCertificate	Requests a public or private certificate.	Write		aws:RequestTag/\${TagKey} (p. 755) aws:TagKeys (p. 755)	
ResendValidationEmail	Resends an email to request domain ownership validation.	Write	certificate* (p. 755)		
UpdateCertificateOptions	Updates a certificate. Use to specify whether to opt in to or out of certificate transparency logging.	Write	certificate* (p. 755)		

Resource Types Defined by AWS Certificate Manager

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 753\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
certificate	<code>arn:\${Partition}:acm:\${Region}:\${Account}:certificate/\${CertificateId}</code>	aws:ResourceTag/\${TagKey} (p. 755)

Condition Keys for AWS Certificate Manager

AWS Certificate Manager defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Filters actions based on the presence of tag key-value pairs in the request	String
aws:ResourceTag/\${TagKey}	Filters actions based on tag key-value pairs attached to the resource	String
aws:TagKeys	Filters actions based on the presence of tag keys in the request	String

Actions, Resources, and Condition Keys for AWS Certificate Manager Private Certificate Authority

AWS Certificate Manager Private Certificate Authority (service prefix: acm-pca) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Certificate Manager Private Certificate Authority \(p. 756\)](#)
- [Resource Types Defined by AWS Certificate Manager Private Certificate Authority \(p. 758\)](#)
- [Condition Keys for AWS Certificate Manager Private Certificate Authority \(p. 759\)](#)

Actions Defined by AWS Certificate Manager Private Certificate Authority

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateCertificateAuthority	Creates an ACM Private CA and its associated private key and configuration.	Tagging		aws:RequestTag/\${TagKey} (p. 759) aws:TagKeys (p. 759)	
CreateCertificateAuthorityReport	Creates an audit report for an ACM Private CA.	Write	certificate-authority* (p. 758)		
CreatePermission	Creates a permission for an ACM Private CA.	Permissions management	certificate-authority* (p. 758)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteCertificateAuthority	Deletes an ACM Private CA and its associated private key and configuration.	Write	certificate-authority* (p. 758)		
DeletePermission	Deletes a permission for an ACM Private CA.	Permissions-management	certificate-authority* (p. 758)		
DescribeCertificateAuthority	Returns a list of the configuration and status fields contained in the specified ACM Private CA.	Read	certificate-authority* (p. 758)		
DescribeCertificateInformationAuditReport	Returns the status and information about an ACM Private CA audit report.	Read	certificate-authority* (p. 758)		
GetCertificate	Retrieves an ACM Private CA certificate and certificate chain for the certificate authority specified by an ARN.	Read	certificate-authority* (p. 758)		
GetCertificateAuthorityCertificate	Retrieves an ACM Private CA certificate and certificate chain for the certificate authority specified by an ARN.	Read	certificate-authority* (p. 758)		
GetCertificateAuthorityCertificate	Retrieves an ACM Private CA certificate signing request (CSR) for the certificate-authority specified by an ARN.	Read	certificate-authority* (p. 758)		
ImportCertificateIntoACMPrivateCA	Imports an SSL/TLS certificate into an ACM Private CA for use as the CA certificate of an ACM Private CA.	Write	certificate-authority* (p. 758)		
IssueCertificate	Issues an ACM Private CA certificate.	Write	certificate-authority* (p. 758)		
ListCertificateAuthorities	Retrieves a list of the ACM Private CA certificate authority ARNs, and a summary of the status of each CA in the calling account.	List			
ListPermissions	Lists the permissions that have been applied to the ACM Private CA certificate authority.	Read	certificate-authority* (p. 758)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListTags	Lists the tags that have been applied to the ACM Private CA certificate authority.	Read	certificate-authority* (p. 758)		
RestoreCertificateFromDelete	Restores an ACM Private CA from the deleted state to the state it was in when deleted.	Write	certificate-authority* (p. 758)		
RevokeCertificate	Revokes a certificate issued by an ACM Private CA.	Write	certificate-authority* (p. 758)		
TagCertificateAuthority	Adds one or more tags to an ACM Private CA.	Tagging	certificate-authority* (p. 758)		
				aws:TagKeys (p. 759) aws:RequestTag/\${TagKey} (p. 759)	
UntagCertificateAuthority	Remove one or more tags from an ACM Private CA.	Tagging	certificate-authority* (p. 758)		
				aws:TagKeys (p. 759)	
UpdateCertificateAuthority	Updates the configuration of an ACM Private CA.	Write	certificate-authority* (p. 758)		

Resource Types Defined by AWS Certificate Manager Private Certificate Authority

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 756\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
certificate-authority	<code>arn:\${Partition}:acm-pca:\${Region}:\${Account}:certificate-authority/\${CertificateAuthorityId}</code>	aws:ResourceTag/\${TagKey} (p. 759)

Condition Keys for AWS Certificate Manager Private Certificate Authority

AWS Certificate Manager Private Certificate Authority defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
acm-pca:TemplateArn	Filters issue certificate requests based on the presence of TemplateArn in the request.	String
aws:RequestTag/\${TagKey}	Filters create requests based on the allowed set of values for each of the tags.	String
aws:ResourceTag/\${TagKey}	Filters actions based on tag-value associated with the resource.	String
aws:TagKeys	Filters create requests based on the presence of mandatory tags in the request.	String

Actions, Resources, and Condition Keys for AWS Chatbot

AWS Chatbot (service prefix: `chatbot`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

Topics

- [Actions Defined by AWS Chatbot \(p. 759\)](#)
- [Resource Types Defined by AWS Chatbot \(p. 760\)](#)
- [Condition Keys for AWS Chatbot \(p. 761\)](#)

Actions Defined by AWS Chatbot

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateChimeWebhookConfiguration.	Creates an AWS Chatbot Chime Webhook configuration.	Write			
CreateSlackChannelConfiguration.	Creates an AWS Chatbot Slack Channel configuration.	Write			
DeleteChimeWebhookConfiguration.	Deletes an AWS Chatbot Chime Webhook configuration.	Write			
DeleteSlackChannelConfiguration.	Deletes an AWS Chatbot Slack Channel configuration.	Write			
DescribeChimeWebhookConfigurations.	Lists all AWS Chatbot Chime Webhook configurations in an AWS Account.	Read			
DescribeSlackChannelConfigurations.	Lists all AWS Chatbot Slack Channel configurations in an AWS account.	Read			
DescribeSlackChannels.	Lists all public Slack channels in the Slack workspace connected to the AWS Account onboarded with AWS Chatbot service.	Read			
DescribeSlackWorkspaces.	Lists all authorized Slack Workspaces connected to the AWS Account onboarded with AWS Chatbot service.	Read			
GetSlackOauthParameters.	Generate OAuth parameters to request Slack OAuth code to be used by the AWS Chatbot service.	Read			
RedeemSlackOAuthCode.	Redeem previously generated OAuth code with Slack API, to acquire OAuth tokens to be used by the AWS Chatbot service.	Write			
UpdateChimeWebhookConfiguration.	Updates an AWS Chatbot Chime Webhook configuration.	Write			
UpdateSlackChannelConfiguration.	Updates an AWS Chatbot Slack Channel configuration.	Write			

Resource Types Defined by AWS Chatbot

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 759\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
ChatbotConfiguration	arn:\${Partition}:chatbot::\${account}:ChatbotConfiguration/\${resourceType}/\${resourceName}	

Condition Keys for AWS Chatbot

Chatbot has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Amazon Chime

Amazon Chime (service prefix: `chime`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Chime \(p. 761\)](#)
- [Resource Types Defined by Amazon Chime \(p. 773\)](#)
- [Condition Keys for Amazon Chime \(p. 773\)](#)

Actions Defined by Amazon Chime

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AcceptDelegate	Grants permission to accept the delegate invitation to share management of an Amazon	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
	Chime account with another AWS Account				
ActivateUsers	Grants permission to activate users in an Amazon Chime Enterprise account	Write			
AddDomain	Grants permission to add a domain to your Amazon Chime account	Write			
AddOrUpdateGroupMembership	Grants permission to add new or update existing Active Directory or Okta user groups associated with your Amazon Chime Enterprise account	Write			
AssociatePhoneNumberWithUser	Grants permission to associate a Phone Number with an Amazon Chime user	Write			
AssociatePhoneNumberWithVoiceConnector	Grants permission to associate multiple Phone Numbers with an Amazon Chime Voice Connector	Write			
AssociatePhoneNumberWithVoiceConnectorGroup	Grants permission to associate multiple Phone Numbers with an Amazon Chime Voice Connector Group	Write			
AuthorizeDirectory	Grants permission to authorize an Active Directory for your Amazon Chime Enterprise account	Write			
BatchCreateAttendee	Grants permission to create new attendees for an active Amazon Chime SDK meeting	Write			
BatchCreateRoomMembers	Grants permission to batch add Room members	Write			
BatchDeletePhoneNumbers	Grants permission to move up to 50 phone numbers to the deletion queue	Write			
BatchSuspendUser	Grants permission to suspend up to 50 users from a Team or EnterpriseLWA Amazon Chime account	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
BatchUnsuspendUser	Grants permission to remove the suspension from up to 50 previously suspended users for the specified Amazon Chime EnterpriseLWA account	Write			
BatchUpdatePhoneNumber	Grants permission to update phone number details within the UpdatePhoneNumberRequestItem object for up to 50 phone numbers	Write			
BatchUpdateUser	Grants permission to update user details within the UpdateUserRequestItem object for up to 20 users for the specified Amazon Chime account	Write			
ConnectDirectory	Grants permission to connect an Active Directory to your Amazon Chime Enterprise account	Write			ds:ConnectDirectory
CreateAccount	Grants permission to create an Amazon Chime account under the administrator's AWS account	Write			
CreateApiKey	Grants permission to create a new SCIM access key for your Amazon Chime account and Okta configuration	Write			
CreateAttendee	Grants permission to create a new attendee for an active Amazon Chime SDK meeting	Write			
CreateBot	Grants permission to create a bot for an Amazon Chime Enterprise account	Write			
CreateBotMember	Grants permission to add a bot to a chat room in your Amazon Chime Enterprise account	Write			
CreateCDRBucket	Grants permission to create a new Call Detail Record S3 bucket	Write			s3:CreateBucket s3>ListAllMyBuckets
CreateMeeting	Grants permission to create a new Amazon Chime SDK meeting in the specified media Region, with no initial attendees	Write			
CreatePhoneNumberOrder	Grants permission to create a phone number order with the Carriers	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateRoom	Grants permission to create a room	Write			
CreateRoomMember	Grants permission to add a room member	Write			
CreateVoiceConnector	Grants permission to create a Amazon Chime Voice Connector under the administrator's AWS account	Write			
CreateVoiceConnectorGroup	Grants permission to create a Amazon Chime Voice Connector Group under the administrator's AWS account	Write			
DeleteAccount	Grants permission to delete the specified Amazon Chime account	Write			
DeleteAccountOpConfig	Grants permission to delete the OpenID Connect config attributes from your Amazon Chime account	Write			
DeleteApiKey	Grants permission to delete the specified SCIM access key associated with your Amazon Chime account and Okta configuration	Write			
DeleteAttendee	Grants permission to delete the specified attendee from an Amazon Chime SDK meeting	Write			
DeleteCDRBucket	Grants permission to delete a Call Detail Record S3 bucket from your Amazon Chime account	Write			s3:DeleteBucket
DeleteDelegate	Grants permission to delete delegated AWS account management from your Amazon Chime account	Write			
DeleteDomain	Grants permission to delete a domain from your Amazon Chime account	Write			
DeleteEventsConfig	Grants permission to delete an events configuration for a bot to receive outgoing events	Write			
DeleteGroups	Grants permission to delete Active Directory or Okta user groups from your Amazon Chime Enterprise account	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteMeeting	Grants permission to delete the specified Amazon Chime SDK meeting	Write			
DeletePhoneNumber	Grants permission to move a phone number to the deletion queue	Write			
DeleteRoom	Grants permission to delete a room	Write			
DeleteRoomMember	Grants permission to remove a room member	Write			
DeleteVoiceConnector	Grants permission to delete the specified Amazon Chime Voice Connector	Write			
DeleteVoiceConnectorGroup	Grants permission to delete the specified Amazon Chime Voice Connector Group	Write			
DeleteVoiceConnectorOrigination	Grants permission to delete the origination settings for the specified Amazon Chime Voice Connector	Write			
DeleteVoiceConnectorStreamingConfiguration	Grants permission to delete the streaming configuration for the specified Amazon Chime Voice Connector	Write			
DeleteVoiceConnectorTermination	Grants permission to delete the termination settings for the specified Amazon Chime Voice Connector	Write			
DeleteVoiceConnectorTerminationCredentials	Grants permission to delete SIP termination credentials for the specified Amazon Chime Voice Connector	Write			
DisassociatePhoneNumberFromUser	Grants permission to disassociate the primary provisioned number from the specified Amazon Chime user	Write			
DisassociatePhoneNumbersFromVoiceConnector	Grants permission to disassociate multiple phone numbers from the specified Amazon Chime Voice Connector	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DisassociatePhoneNumbersFromMultiplePhoneNumbers	Grants permission to DisassociatePhoneNumbersFromMultiplePhoneNumbers numbers from the specified Amazon Chime Voice Connector Group	Write			
DisconnectDirectory	Grants permission to disconnect the Active Directory from your Amazon Chime Enterprise account	Write			
GetAccount	Grants permission to get details for the specified Amazon Chime account	Read			
GetAccountResources	Grants permission to get details for the account resource associated with your Amazon Chime account	Read			
GetAccountSettings	Grants permission to get account settings for the specified Amazon Chime account ID	Read			
GetAccountWithOpenIdConfig	Grants permission to get the account details and OpenIdConfig attributes for your Amazon Chime account	Read			
GetAttendee	Grants permission to get attendee details for a specified meeting ID and attendee ID	Read			
GetBot	Grants permission to retrieve details for the specified bot	Read			
GetCDRBucket	Grants permission to get details of a Call Detail Record S3 bucket associated with your Amazon Chime account	Read			s3:GetBucketAcl s3:GetBucketLocation s3:GetBucketLogging s3:GetBucketVersioning s3:GetBucketWebsite
GetDomain	Grants permission to get domain details for a domain associated with your Amazon Chime account	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetEventsConfiguration	Grants permission to retrieve details for an events configuration for a bot to receive outgoing events	Read			
GetGlobalSetting	Grants permission to get global settings related to Amazon Chime for the AWS account	Read			
GetMeeting	Grants permission to get the meeting record for a specified meeting ID	Read			
GetMeetingDetail	Grants permission to get attendee, connection, and other details for a meeting	Read			
GetPhoneNumber	Grants permission to get details for the specified phone number	Read			
GetPhoneNumberOrder	Grants permission to get details for the specified phone number order	Read			
GetPhoneNumberNumbers	Grants permission to get phone numbers settings related to Amazon Chime for the AWS account	Read			
GetRoom	Grants permission to retrieve a room	Read			
GetTelephonyLimits	Grants permission to get telephony limits for the AWS account	Read			
 GetUser	Grants permission to get details for the specified user ID	Read			
 GetUserActivityReportData	Grants permission to get a summary of user activity on the user details page	Read			
 GetUserByEmail	Grants permission to get user details for an Amazon Chime user based on the email address in an Amazon Chime Enterprise or Team account	Read			
 GetUserSettings	Grants permission to get user settings related to the specified Amazon Chime user	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetVoiceConnector	Grants permission to get details for the specified Amazon Chime Voice Connector	Read			
GetVoiceConnectorGroup	Grants permission to get details for the specified Amazon Chime Voice Connector Group	Read			
GetVoiceConnectorLoggingConfiguration	Grants permission to get details of the logging configuration for the specified Amazon Chime Voice Connector	Read			
GetVoiceConnectorOrigination	Grants permission to get details of the origination settings for the specified Amazon Chime Voice Connector	Read			
GetVoiceConnectorStreamingConfiguration	Grants permission to get details of the streaming configuration for the specified Amazon Chime Voice Connector	Read			
GetVoiceConnectorTermination	Grants permission to get details of the termination settings for the specified Amazon Chime Voice Connector	Read			
GetVoiceConnectorTerminationHealth	Grants permission to get details of the termination health for the specified Amazon Chime Voice Connector	Read			
InviteDelegate	Grants permission to send an invitation to accept a request for AWS account delegation for an Amazon Chime account	Write			
InviteUsers	Grants permission to invite as many as 50 users to the specified Amazon Chime account	Write			
InviteUsersFromProvider	Grants permission to invite users from a third party provider to your Amazon Chime account	Write			
ListAccountUsage	Grants permission to list Amazon Chime account usage reporting data	List			
ListAccounts	Grants permission to list the Amazon Chime accounts under the administrator's AWS account	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListApiKeys	Grants permission to list the SCIM access keys defined for your Amazon Chime account and Okta configuration	List			
ListAttendees	Grants permission to list up to 100 attendees for a specified Amazon Chime SDK meeting	Read			
ListBots	Grants permission to list the bots associated with the administrator's Amazon Chime Enterprise account	List			
ListCDRBucket	Grants permission to list Call Detail Record S3 buckets	List			s3>ListAllMyBuckets s3>ListBucket
ListCallingRegions	Grants permission to list the calling regions available for the administrator's AWS account	List			
ListDelegates	Grants permission to list account delegate information associated with your Amazon Chime account	List			
ListDirectories	Grants permission to list active Active Directories hosted in the Directory Service of your AWS account	List			
ListDomains	Grants permission to list domains associated with your Amazon Chime account	List			
ListGroups	Grants permission to list Active Directory or Okta user groups associated with your Amazon Chime Enterprise account	List			
ListMeetingEvents	Grants permission to list all events that occurred for a specified meeting	List			
ListMeetings	Grants permission to list up to 100 active Amazon Chime SDK meetings	Read			
ListMeetingsReported	Grants permission to list meetings ended during the specified date range	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListPhoneNumbers	Grants permission to list the phone numbers under the administrator's AWS account	List			
ListPhoneNumbers	Grants permission to list the phone numbers under the administrator's AWS account	List			
ListRoomMembership	Grants permission to list all room members	Read			
ListRooms	Grants permission to list rooms	Read			
ListUsers	Grants permission to list the users that belong to the specified Amazon Chime account	List			
ListVoiceConnects	Grants permission to list the Amazon Chime Voice Connector Groups under the administrator's AWS account	List			
ListVoiceConnectorTerminationCredentials	Grants permission to list the SIP termination credentials for the specified Amazon Chime Voice Connector	List			
ListVoiceConnectors	Grants permission to list the Amazon Chime Voice Connectors under the administrator's AWS account	List			
LogoutUser	Grants permission to log out the specified user from all of the devices they are currently logged into	Write			
PutEventsConfiguration	Grants permission to update details for an events configuration for a bot to receive outgoing events	Write			
PutVoiceConnectorLoggingConfiguration	Grants permission to add logging configuration for the specified Amazon Chime Voice Connector	Write			logs>CreateLogDelivery logs>CreateLogGroup logs>DeleteLogDelivery logs>DescribeLogGroups logs>GetLogDelivery logs>ListLogDeliveries

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
PutVoiceConnectorLogRetention	Grants permission to update the log retention settings for the specified Amazon Chime Voice Connector	Write			
PutVoiceConnectorStreamingConfiguration	Grants permission to add streaming configuration for the specified Amazon Chime Voice Connector	Write			
PutVoiceConnectorTermination	Grants permission to update the termination settings for the specified Amazon Chime Voice Connector	Write			
PutVoiceConnectorTerminationCredentials	Grants permission to add SIP termination credentials for the specified Amazon Chime Voice Connector	Write			
RegenerateSecurityToken	Grants permission to regenerate the security token for the specified bot	Write			
RenameAccount	Grants permission to modify the account name for your Amazon Chime Enterprise or Team account	Write			
RenewDelegate	Grants permission to renew the delegation request associated with an Amazon Chime account	Write			
ResetAccountResource	Grants permission to reset the account resource in your Amazon Chime account	Write			
ResetPersonalPIN	Grants permission to reset the personal meeting PIN for the specified user on an Amazon Chime account	Write			
RestorePhoneNumber	Grants permission to restore the specified phone number from the deletion queue back to the phone number inventory	Write			
RetrieveDataExports	Grants permission to download the file containing links to all user attachments returned as part of the "Request attachments" action	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
SearchAvailablePhoneNumbers	Grants permission to search phone numbers that can be ordered from the carrier	Read			
StartDataExport	Grants permission to submit the "Request attachments" request	Write			
SubmitSupportRequest	Grants permission to submit a customer service support request	Write			
SuspendUsers	Grants permission to suspend users from an Amazon Chime Enterprise account	Write			
UnauthorizeDirectory	Grants permission to unauthorize an Active Directory from your Amazon Chime Enterprise account	Write			
UpdateAccount	Grants permission to update account details for the specified Amazon Chime account	Write			
UpdateAccountOpenIdConfig	Grants permission to update the OpenID Config attributes for your Amazon Chime account	Write			
UpdateAccountResource	Grants permission to update the account resource in your Amazon Chime account	Write			
UpdateAccountSettings	Grants permission to update the settings for the specified Amazon Chime account	Write			
UpdateBot	Grants permission to update the status of the specified bot	Write			
UpdateCDRSettings	Grants permission to update your Call Detail Record S3 bucket	Write			s3:CreateBucket s3:DeleteBucket s3>ListAllMyBuckets
UpdateGlobalSettings	Grants permission to update the global settings related to Amazon Chime for the AWS account	Write			
UpdatePhoneNumber	Grants permission to update phone number details for the specified phone number	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UpdatePhoneNumber	Grants permission to update phone number settings related to Amazon Chime for the AWS account	Write			
UpdateRoom	Grants permission to update a room	Write			
UpdateRoomMembership	Grants permission to update room membership role	Write			
UpdateSupportedLanguages	Grants permission to update the supported languages license tiers available for users in your Amazon Chime account	Write			
UpdateUser	Grants permission to update user details for a specified user ID	Write			
UpdateUserLicenses	Grants permission to update the licenses for your Amazon Chime users	Write			
UpdateUserSettings	Grants permission to update user settings related to the specified Amazon Chime user	Write			
UpdateVoiceConnector	Grants permission to update Amazon Chime Voice Connector details for the specified Amazon Chime Voice Connector	Write			
UpdateVoiceConnectorGroup	Grants permission to update Amazon Chime Voice Connector Group details for the specified Amazon Chime Voice Connector Group	Write			
ValidateAccount	Grants permission to validate the account resource in your Amazon Chime account	Read			

Resource Types Defined by Amazon Chime

Amazon Chime does not support specifying a resource ARN in the `Resource` element of an IAM policy statement. To allow access to Amazon Chime, specify “`Resource`”: “`*`” in your policy.

Condition Keys for Amazon Chime

Chime has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Amazon Cloud Directory

Amazon Cloud Directory (service prefix: `clouddirectory`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Cloud Directory \(p. 774\)](#)
- [Resource Types Defined by Amazon Cloud Directory \(p. 779\)](#)
- [Condition Keys for Amazon Cloud Directory \(p. 780\)](#)

Actions Defined by Amazon Cloud Directory

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AddFacetToObject	Adds a new Facet to an object.	Write	directory* (p. 779)		
ApplySchema	Copies input published schema into Directory with same name and version as that of published schema.	Write	directory* (p. 779)		
			publishedSchema* (p. 780)		
AttachObject	Attaches an existing object to another existing object.	Write	directory* (p. 779)		
AttachPolicy	Attaches a policy object to any other object.	Write	directory* (p. 779)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AttachToIndex	Attaches the specified object to the specified index.	Write	directory* (p. 779)		
AttachTypedLink	Attaches a typed link b/w a source & target object reference.	Write	directory* (p. 779)		
BatchRead	Performs all the read operations in a batch. Each individual operation inside BatchRead needs to be granted permissions explicitly.	Read	directory* (p. 779)		
BatchWrite	Performs all the write operations in a batch. Each individual operation inside BatchWrite needs to be granted permissions explicitly.	Write	directory* (p. 779)		
CreateDirectory	Creates a Directory by copying the published schema into the directory.	Write	publishedSchema* (p. 780)		
CreateFacet	Creates a new Facet in a schema.	Write	appliedSchema* (p. 779)		
			developmentSchema* (p. 779)		
CreateIndex	Creates an index object.	Write	directory* (p. 779)		
CreateObject	Creates an object in a Directory.	Write	directory* (p. 779)		
CreateSchema	Creates a new schema in a development state.	Write			
CreateTypedLinkFacet	Creates a new Typed Link facet schema.	Write	appliedSchema* (p. 779)		
			developmentSchema* (p. 779)		
DeleteDirectory	Deletes a directory. Only disabled directories can be deleted.	Write	directory* (p. 779)		
DeleteFacet	Deletes a given Facet. All attributes and Rules associated with the facet will be deleted.	Write	developmentSchema* (p. 779)		
DeleteObject	Deletes an object and its associated attributes.	Write	directory* (p. 779)		
DeleteSchema	Deletes a given schema.	Write	developmentSchema* (p. 779)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			publishedSchema* (p. 780)		
DeleteTypedLinkFacet	Deletes a given TypedLink Facet. All attributes and Rules associated with the facet will be deleted.	Write	developmentSchema* (p. 779)		
DetachFromIndex	Detaches the specified object from the specified index.	Write	directory* (p. 779)		
DetachObject	Detaches a given object from the parent object.	Write	directory* (p. 779)		
DetachPolicy	Detaches a policy from an object.	Write	directory* (p. 779)		
DetachTypedLink	Detaches a given typed link b/w given source and target object reference.	Write	directory* (p. 779)		
DisableDirectory	Disables the specified directory.	Write	directory* (p. 779)		
EnableDirectory	Enables the specified directory.	Write	directory* (p. 779)		
GetDirectory	Retrieves metadata about a directory.	Read	directory* (p. 779)		
GetFacet	Gets details of the Facet, such as Facet Name, Attributes, Rules, or ObjectType.	Read	appliedSchema* (p. 779)		
			developmentSchema* (p. 779)		
			publishedSchema* (p. 780)		
GetLinkAttributes	Retrieves attributes that are associated with a typed link.	Read	directory* (p. 779)		
GetObjectAttributes	Retrieves attributes within a facet that are associated with an object.	Read	directory* (p. 779)		
GetObjectInformation	Retrieves metadata about an object.	Read	directory* (p. 779)		
GetSchemaAsJson	Retrieves a JSON representation of the schema.	Read	appliedSchema* (p. 779)		
			developmentSchema* (p. 779)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			publishedSchema* (p. 780)		
GetTypedLinkFacetInformation	Returns identity attributes order associated with a given typed link facet.	Read	appliedSchema* (p. 779)		
	developmentSchema* (p. 779)				
	publishedSchema* (p. 780)				
ListAppliedSchemas	Lists schemas applied to a directory.	List	directory* (p. 779)		
ListAttachedIndices	Lists indices attached to an object.	Read	directory* (p. 779)		
ListDevelopmentSchemas	Retrieves the ARNs of schemas in the development state.	List			
ListDirectories	Lists directories created within an account.	List			
ListFacetAttributes	Retrieves attributes attached to the facet.	Read	appliedSchema* (p. 779)		
	developmentSchema* (p. 779)				
	publishedSchema* (p. 780)				
ListFacetNames	Retrieves the names of facets that exist in a schema.	Read	appliedSchema* (p. 779)		
	developmentSchema* (p. 779)				
	publishedSchema* (p. 780)				
ListIncomingTypedLinks	Returns a paginated list of all incoming TypedLinks for a given object.	Read	directory* (p. 779)		
ListIndex	Lists objects attached to the specified index.	Read	directory* (p. 779)		
ListObjectAttributes	Lists all attributes associated with an object.	Read	directory* (p. 779)		
ListObjectChildren	Returns a paginated list of child objects associated with a given object.	Read	directory* (p. 779)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListObjectParentPaths	Retrieves all available parent paths for any object type such as node, leaf node, policy node, and index node objects.	Read	directory* (p. 779)		
ListObjectParents	Lists parent objects associated with a given object in pagination fashion.	Read	directory* (p. 779)		
ListObjectPolicies	Returns policies attached to an object in pagination fashion.	Read	directory* (p. 779)		
ListOutgoingTypedLinks	Returns a paginated list of all TypedLinks for a given object.	Read	directory* (p. 779)		
ListPolicyAttachments	Returns all of the ObjectIdentifiers to which a given policy is attached.	Read	directory* (p. 779)		
ListPublishedSchemas	Retrieves published schema ARNs	List			
ListTagsForResource	Returns tags for a resource.	Read	directory* (p. 779)		
ListTypedLinkFacetAttributes	Returns a paginated list of attributes associated with typed link facet.	Read	appliedSchema* (p. 779)		
			developmentSchema* (p. 779)		
			publishedSchema* (p. 780)		
ListTypedLinkFacetNames	Returns a paginated list of typed link facet names that exist in a schema.	Read	appliedSchema* (p. 779)		
			developmentSchema* (p. 779)		
			publishedSchema* (p. 780)		
LookupPolicy	Lists all policies from the root of the Directory to the object specified.	Read	directory* (p. 779)		
PublishSchema	Publishes a development schema with a version.	Write	developmentSchema* (p. 779)		
PutSchemaFromJSON	Allows a schema to be updated using JSON upload. Only available for development schemas.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
RemoveFacetFromObject	Removes the specified facet from the specified object.	Write	directory* (p. 779)		
TagResource	Adds tags to a resource.	Tagging	directory* (p. 779)		
UntagResource	Removes tags from a resource.	Tagging	directory* (p. 779)		
UpdateFacet	Adds/Updates/Deletes existing Attributes, Rules, or ObjectType of a Facet.	Write	appliedSchema* (p. 779)		
			developmentSchema* (p. 779)		
UpdateLinkAttribute	Updates a given typed link's attributes. Attributes to be updated must not contribute to the typed link's identity, as defined by its IdentityAttributeOrder.	Write	directory* (p. 779)		
UpdateObjectAttribute	Updates a given object's attributes.	Write	directory* (p. 779)		
UpdateSchema	Updates the schema name with a new name.	Write	developmentSchema* (p. 779)		
UpdateTypedLinkAttribute	Adds/Updates/Deletes existing Attributes, Rules, identity attribute order of a TypedLink Facet.	Write	developmentSchema* (p. 779)		

Resource Types Defined by Amazon Cloud Directory

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 774\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
appliedSchema	arn:\${Partition}:clouddirectory:\${Region}: \${Account}:directory/\${DirectoryId}/schema/\${SchemaName}/\${Version}	
developmentSchema	arn:\${Partition}:clouddirectory:\${Region}: \${Account}:schema/development/\${SchemaName}	
directory	arn:\${Partition}:clouddirectory:\${Region}: \${Account}:directory/\${DirectoryId}	

Resource Types	ARN	Condition Keys
publishedSchema	arn:\${Partition}:clouddirectory:\${Region}: \${Account}:schema/published/\${SchemaName}/ \${Version}	

Condition Keys for Amazon Cloud Directory

Cloud Directory has no service-specific context keys that can be used in the Condition element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for AWS Cloud Map

AWS Cloud Map (service prefix: `servicediscovery`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Cloud Map \(p. 780\)](#)
- [Resource Types Defined by AWS Cloud Map \(p. 782\)](#)
- [Condition Keys for AWS Cloud Map \(p. 782\)](#)

Actions Defined by AWS Cloud Map

You can specify the following actions in the Action element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the Resource element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateHttpNamespace	Creates an HTTP namespace.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreatePrivateDnsNamespace	Creates a private namespace based on DNS, which will be visible only inside a specified Amazon VPC.	Write			
CreatePublicDnsNamespace	Creates a public namespace based on DNS, which will be visible on the internet.	Write			
CreateService	Creates a service.	Write		servicediscovery:NamespaceArn (p. 782)	
DeleteNamespace	Deletes a specified namespace.	Write	namespace* (p. 782)		
DeleteService	Deletes a specified service.	Write	service* (p. 782)		
DeregisterInstance	Deletes the records and the health check, if any, that Amazon Route 53 created for the specified instance.	Write		servicediscovery:ServiceArn (p. 782)	
DiscoverInstances	Discovers registered instances for a specified namespace and service.	Read		servicediscovery:NamespaceName (p. 782)	servicediscovery:ServiceName (p. 783)
GetInstance	Gets information about a specified instance.	Read		servicediscovery:ServiceArn (p. 782)	
GetInstancesHealthStatus	Gets the current health status (Healthy, Unhealthy, or Unknown) of one or more instances.	Read		servicediscovery:ServiceArn (p. 782)	
GetNamespace	Gets information about a namespace.	Read	namespace* (p. 782)		
GetOperation	Gets information about a specific operation.	Read			
GetService	Gets the settings for a specified service.	Read	service* (p. 782)		
ListInstances	Gets summary information about the instances that were registered with a specified service.	List		servicediscovery:ServiceArn (p. 782)	
ListNamespaces	Gets information about the namespaces.	List			
ListOperations	Lists operations that match the criteria that you specify.	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListServices	Gets settings for all the services that match specified filters.	List			
RegisterInstance	Registers an instance based on the settings in a specified service.	Write		servicediscovery:ServiceArn (p. 782)	
UpdateInstanceCustomHealthStatusForlahInstance	Updates the current health status for lahInstance that has a custom health check.	Write		servicediscovery:ServiceArn (p. 782)	
UpdateService	Updates the settings in a specified service.	Write	service* (p. 782)		

Resource Types Defined by AWS Cloud Map

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 780\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
namespace	<code>arn:\${Partition}:servicediscovery:\${Region}:\${Account}:namespace/\${NamespaceId}</code>	
service	<code>arn:\${Partition}:servicediscovery:\${Region}:\${Account}:service/\${ServiceId}</code>	

Condition Keys for AWS Cloud Map

AWS Cloud Map defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
servicediscovery:NameResourceName	A filter that lets you get objects by specifying the Amazon Resource Name (ARN) for the related namespace.	String
servicediscovery:NameTheRelatedNamespace	A filter that lets you get objects by specifying the name of the related namespace.	String
servicediscovery:ServiceResourceName	A filter that lets you get objects by specifying the Amazon Resource Name (ARN) for the related service.	String

Condition Keys	Description	Type
servicediscovery:ServiceName	A filter that lets you get objects by specifying the name of the service.	String

Actions, Resources, and Condition Keys for AWS Cloud9

AWS Cloud9 (service prefix: `cloud9`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Cloud9 \(p. 783\)](#)
- [Resource Types Defined by AWS Cloud9 \(p. 785\)](#)
- [Condition Keys for AWS Cloud9 \(p. 785\)](#)

Actions Defined by AWS Cloud9

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateEnvironment	Grants permission to create an AWS Cloud9 development environment, launches an Amazon Elastic Compute Cloud (Amazon EC2) instance, and then hosts the environment on the instance.	Write		cloud9:EnvironmentArn cloud9:InstanceType cloud9:SubnetId cloud9:UserArn	ec2:DescribeSubnets ec2:DescribeVpcs iam:CreateServiceLinkedRole

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateEnvironmentMember	Grants permission to add an environment member to an AWS Cloud9 development environment.	Write		cloud9:UserArn (p. 786) cloud9:EnvironmentId (p. 785) cloud9:Permissions (p. 785)	
DeleteEnvironmentMember	Grants permission to delete an AWS Cloud9 development environment. If the environment is hosted on an Amazon Elastic Compute Cloud (Amazon EC2) instance, also terminates the instance.	Write	environment* (p. 785)		iam:CreateServiceLinkedRole
DeleteEnvironmentMemberFromEnvironment	Grants permission to delete an environment member from an AWS Cloud9 development environment.	Write			
DescribeEnvironmentMembers	Grants permission to get information about environment members for an AWS Cloud9 development environment.	Read		cloud9:UserArn (p. 786) cloud9:EnvironmentId (p. 785)	
DescribeEnvironmentInformation	Grants permission to get status information for an AWS Cloud9 development environment.	Read			
DescribeEnvironments	Grants permission to get information about AWS Cloud9 development environments.	Read	environment* (p. 785)		
GetUserSettings [permission only]	Grants permission to get IDE-specific settings of an AWS Cloud9 user.	Read			
ListEnvironments	Grants permission to get a list of AWS Cloud9 development environment identifiers.	Read			
UpdateEnvironment	Grants permission to change the settings of an existing AWS Cloud9 development environment.	Write	environment* (p. 785)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UpdateEnvironment	Grants permission to change the settings of an existing environment member for an AWS Cloud9 development environment.	Write		cloud9:UserArn (p. 786) cloud9:EnvironmentId (p. 785) cloud9:Permissions (p. 785)	
UpdateUserSettings [permission only]	Grants permission to update IDE-specific settings of an AWS Cloud9 user.	Write			

Resource Types Defined by AWS Cloud9

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 783\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
environment	arn:\${Partition}:cloud9:\${Region}:\${Account}:environment:\${ResourceId}	

Condition Keys for AWS Cloud9

AWS Cloud9 defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
cloud9:EnvironmentId	Filters access by the AWS Cloud9 environment ID	String
cloud9:EnvironmentName	Filters access by the AWS Cloud9 environment name	String
cloud9:InstanceType	Filters access by the instance type of the AWS Cloud9 environment's Amazon EC2 instance	String
cloud9:Permissions	Filters access by the type of AWS Cloud9 permissions	String

Condition Keys	Description	Type
cloud9:SubnetId	Filters access by the subnet ID that the AWS Cloud9 environment will be created in	String
cloud9:UserArn	Filters access by the user ARN specified	ARN

Actions, Resources, and Condition Keys for AWS CloudFormation

AWS CloudFormation (service prefix: `cloudformation`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS CloudFormation \(p. 786\)](#)
- [Resource Types Defined by AWS CloudFormation \(p. 792\)](#)
- [Condition Keys for AWS CloudFormation \(p. 792\)](#)

Actions Defined by AWS CloudFormation

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CancelUpdateStack	Cancels an update on the specified stack.	Write	stack* (p. 792)		
ContinueUpdateRollback	For a specified stack that is in the <code>UPDATE_ROLLBACK_FAILED</code> state, continues rolling it back to the <code>UPDATE_ROLLBACK_COMPLETE</code> state.	Write	stack* (p. 792)		
				cloudformation:RoleArn (p. 793)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateChangeSet	Creates a list of changes for a stack.	Write	stack* (p. 792)		cloudformation:ChangeSetName (p. 793) cloudformation:ResourceTypes (p. 793) cloudformation:RoleArn (p. 793) cloudformation:StackPolicyUrl (p. 793) cloudformation:TemplateUrl (p. 793) aws:RequestTag/\${TagKey} (p. 793) aws:TagKeys (p. 793)
CreateStack	Creates a stack as specified in the template.	Write	stack* (p. 792)		cloudformation:ResourceTypes (p. 793) cloudformation:RoleArn (p. 793) cloudformation:StackPolicyUrl (p. 793) cloudformation:TemplateUrl (p. 793) aws:RequestTag/\${TagKey} (p. 793) aws:TagKeys (p. 793)
CreateStackInstances	Creates stack instances for the specified accounts, within the specified regions.	Write	stackset* (p. 792)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateStackSet	Creates a stackset as specified in the template.	Write		cloudformation:RoleArn (p. 793) cloudformation:TemplateUrl (p. 793) aws:RequestTag/\${TagKey} (p. 793) aws:TagKeys (p. 793)	
CreateUploadBucket [permission only]		Write			
DeleteChangeSet	Deletes the specified change set. Deleting change sets ensures that no one executes the wrong change set.	Write	stack* (p. 792)		
				cloudformation:ChangeSetName (p. 793)	
DeleteStack	Deletes a specified stack.	Write	stack* (p. 792)		
				cloudformation:RoleArn (p. 793)	
DeleteStackInstances	Deletes stack instances for the specified accounts, in the specified regions.	Write	stackset* (p. 792)		
DeleteStackSet	Deletes a specified stackset.	Write	stackset* (p. 792)		
DescribeAccountLimits	Retrieves your account's AWS CloudFormation limits.	Read			
DescribeChangeSet	Returns the description for the specified change set.	Read	stack* (p. 792)		
			cloudformation:ChangeSetName (p. 793)		
DescribeStackDriftDetection	Returns information about a Stack drift detection operation.	Read			
DescribeStackEvents	Returns all stack related events for a specified stack.	Read	stack* (p. 792)		
DescribeStackInstances	Returns the stack instance that's associated with the specified stack set, AWS account, and region.	Read	stackset* (p. 792)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeStackResources	Returns a description of the specified resource in the specified stack.	Read	stack* (p. 792)		
DescribeStackResourceDrifts	Returns drift information for the resources that have been checked for drift in the specified stack.	Read	stack* (p. 792)		
DescribeStackResourceDescriptions	Returns AWS resource descriptions for running and deleted stacks.	Read	stack* (p. 792)		
DescribeStackSets	Returns the description of the specified stack set.	Read	stackset* (p. 792)		
DescribeStackSetOperations	Returns the description of the specified stack set operation.	Read	stackset* (p. 792)		
DescribeStacks	Returns the description for the specified stack.	List	stack* (p. 792)		
DetectStackDrift	Detects whether a stack's actual configuration differs, or has drifted, from its expected configuration, as defined in the stack template and any values specified as template parameters.	Read	stack* (p. 792)		
DetectStackResourceDrifts	Returns information about whether a resource's actual configuration differs, or has drifted, from its expected configuration, as defined in the stack template and any values specified as template parameters.	Read	stack* (p. 792)		
EstimateTemplateCost	Returns the estimated monthly cost of a template.	Read			
ExecuteChangeSet	Updates a stack using the input information that was provided when the specified change set was created.	Write	stack* (p. 792)		
					cloudformation:ChangeSetName (p. 793)
GetStackPolicy	Returns the stack policy for a specified stack.	Read	stack* (p. 792)		
GetTemplate	Returns the template body for a specified stack.	Read	stack* (p. 792)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetTemplateSummary	Returns information about a new or existing template.	Read	stack (p. 792)		
			stackset (p. 792)		
ListChangeSets	Returns the ID and status of each active change set for a stack. For example, AWS CloudFormation lists change sets that are in the CREATE_IN_PROGRESS or CREATE_PENDING state.	List	stack* (p. 792)		
ListExports	Lists all exported output values in the account and region in which you call this action.	List			
ListImports	Lists all stacks that are importing an exported output value.	List			
ListStackInstances	Returns summary information about stack instances that are associated with the specified stack set.	List	stackset* (p. 792)		
ListStackResources	Returns descriptions of all resources of the specified stack.	List	stack* (p. 792)		
ListStackSetOperations	Returns summary information about the results of a stack set operation.	List	stackset* (p. 792)		
ListStackSetOperationResults	Returns summary information about operations performed on a stack set.	List	stackset* (p. 792)		
ListStackSets	Returns summary information about stack sets that are associated with the user.	List	stackset* (p. 792)		
ListStacks	Returns the summary information for stacks whose status matches the specified StackStatusFilter.	List			
SetStackPolicy	Sets a stack policy for a specified stack.	Permissions management	stack* (p. 792)		
					cloudformation:StackPolicyUrl (p. 793)

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
SignalResource	Sends a signal to the specified resource with a success or failure status.	Write	stack* (p. 792)		
StopStackSetOperation	Stops an in-progress operation on a stack set and its associated stack instances.	Write	stackset* (p. 792)		
TagResource	Tagging cloudformation resources.	Tagging	stack (p. 792)		
			stackset (p. 792)		
UntagResource	Untagging cloudformation resources.	Tagging	stack (p. 792)		
			stackset (p. 792)		
UpdateStack	Updates a stack as specified in the template.	Write	stack* (p. 792)		
	cloudformation:ResourceTypes (p. 793) cloudformation:RoleArn (p. 793) cloudformation:StackPolicyUrl (p. 793) cloudformation:TemplateUrl (p. 793) aws:RequestTag/\${TagKey} (p. 793) aws:TagKeys (p. 793)		CloudFormation		
UpdateStackInstances	Updates the parameter values for stack instances for the specified accounts, within the specified regions.	Write	stackset* (p. 792)		
UpdateStackSet	Updates a stackset as specified in the template.	Write	stackset* (p. 792)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				cloudformation:RoleArn (p. 793) cloudformation:TemplateUrl (p. 793) aws:RequestTag/\${TagKey} (p. 793) aws:TagKeys (p. 793)	
UpdateTerminationProtection	Updates termination protection for the specified stack.	Write	stack* (p. 792)		
ValidateTemplate	Validates a specified template.	Write			

Resource Types Defined by AWS CloudFormation

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 786\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
stack	arn:\${Partition}:cloudformation:\${Region}: \${Account}:stack/\${StackName}/\${Id}	aws:ResourceTag/\${TagKey} (p. 793)
stackset	arn:\${Partition}:cloudformation:\${Region}: \${Account}:stackset/\${StackSetName}:\${Id}	aws:ResourceTag/\${TagKey} (p. 793)
changeset	arn:\${Partition}:cloudformation:\${Region}: \${Account}:changeSet/\${ChangeSetName}:\${Id}	

Condition Keys for AWS CloudFormation

AWS CloudFormation defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
<code>aws:RequestTag/\${TagKey}</code>		String
<code>aws:ResourceTag/\${TagKey}</code>		String
<code>aws:TagKeys</code>		String
<code>cloudformation:ChangeSetName</code>	An AWS CloudFormation change set name. Use to control which IAM users can execute or delete.	String
<code>cloudformation:ResourceType</code>	The template resource types, such as <code>AWS::EC2::Instance</code> . Use to control which resource types IAM users can work with when they create or update a stack	String
<code>cloudformation:RoleArn</code>	The ARN of an IAM service role. Use to control which service IAM users can use to work with stacks or change sets.	ARN
<code>cloudformation:StackPolicyURL</code>	An Amazon S3 stack policy URL. Use to control which IAM users can associate with a stack during a create or update stack action.	String
<code>cloudformation:TemplateURL</code>	An Amazon S3 template URL. Use to control which IAM users can use when they create or update stacks.	String

Actions, Resources, and Condition Keys for Amazon CloudFront

Amazon CloudFront (service prefix: `cloudfront`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon CloudFront \(p. 793\)](#)
- [Resource Types Defined by Amazon CloudFront \(p. 799\)](#)
- [Condition Keys for Amazon CloudFront \(p. 799\)](#)

Actions Defined by Amazon CloudFront

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateCloudFrontOriginAccessIdentity	This action creates a new CloudFront origin access identity (POST /2019-03-26/origin-access-identity/cloudfront).	Write	origin-access-identity* (p. 799)		
CreateDistribution	This action creates a new web distribution (POST /2019-03-26/distribution).	Write	distribution* (p. 799)		
CreateDistributionWithTags	This action creates a new web distribution with tags (POST /2019-03-26/distribution?WithTags).	Tagging	distribution* (p. 799)		
				aws:RequestTag/\${TagKey} (p. 800) aws:TagKeys (p. 800)	
CreateFieldLevelEncryption	This action creates a new field-level encryption configuration. (POST /2019-03-26/field-level-encryption).	Write			
CreateFieldLevelEncryptionProfile	This action creates a field-level encryption profile. (POST /2019-03-26/field-level-encryption-profile).	Write			
CreateInvalidation	This action creates a new invalidation batch request (POST /2019-03-26/distribution/<DISTRIBUTION_ID>/invalidation).	Write	distribution* (p. 799)		
CreatePublicKey	This action adds a new public key to CloudFront. (POST /2019-03-26/public-key).	Write			
CreateStreamingDistribution	This action creates a new RTMP distribution (POST /2019-03-26/streaming-distribution).	Write	streaming-distribution* (p. 799)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateStreamingDistribution	This action creates a new RTMP distribution with tags (POST /2019-03-26/streaming-distribution?WithTags).	Tagging	streaming-distribution* (p. 799)		
				aws:RequestTag/ {\$TagKey} (p. 800)	
					aws:TagKeys (p. 800)
DeleteCloudFrontOriginAccessIdentity	This action deletes a CloudFront origin access identity (DELETE /2019-03-26/origin-access-identity/cloudfront/<OAI_ID>).	Write	origin-access-identity* (p. 799)		
DeleteDistribution	This action deletes a web distribution (DELETE /2019-03-26/distribution/<distribution_id>).	Write	distribution* (p. 799)		
DeleteFieldLevelEncryption	This action deletes a field-level encryption configuration. (DELETE /2019-03-26/field-level-encryption/Id/<Id>).	Write			
DeleteFieldLevelEncryptionProfile	This action deletes a field-level encryption profile. (DELETE /2019-03-26/field-level-encryption-profile/<Id>).	Write			
DeletePublicKey	This action deletes a public key from CloudFront. (DELETE /2019-03-26/public-key/<Id>).	Write			
DeleteStreamingDistribution	This action deletes an RTMP distribution (DELETE /2019-03-26/streaming-distribution/<distribution_id>).	Write	streaming-distribution* (p. 799)		
GetCloudFrontOriginAccessIdentity	Get the information about a CloudFront origin access identity (GET /2019-03-26/origin-access-identity/cloudfront/<OAI_ID>).	Read	origin-access-identity* (p. 799)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetCloudFrontOriginAccessIdentityConfig	Get the configuration information about CloudFront origin access identity (GET /2019-03-26/origin-access-identity/cloudfront/<OAI_ID>/config).	Read	origin-access-identity* (p. 799)		
GetDistribution	Get the information about a web distribution (GET /2019-03-26/distribution/<distribution_id>).	Read	distribution* (p. 799)		
GetDistributionConfig	Get the configuration information about a distribution (GET /2019-03-26/distribution/<distribution_id>/config).	Read	distribution* (p. 799)		
GetFieldLevelEncryptionConfiguration	Get the field-level encryption configuration information. (GET /2019-03-26/field-level-encryption/<Id>).	Read			
GetFieldLevelEncryptionConfig	Get the field-level encryption configuration information (GET /2019-03-26/field-level-encryption/<Id>/config).	Read			
GetFieldLevelEncryptionConfiguration	Get the field-level encryption configuration information (GET /2019-03-26/field-level-encryption/<Id>/config).	Read			
GetFieldLevelEncryptionProfileConfig	Get the field-level encryption profile configuration information (GET /2019-03-26/field-level-encryption-profile/<Id>/config).	Read			
GetInvalidation	Get the information about an invalidation (GET /2019-03-26/distribution/<distribution_id>/invalidation/<invalidation_id>).	Read	distribution* (p. 799)		
GetPublicKey	Get the public key information (GET /2019-03-26/public-key/<Id>).	Read			
GetPublicKeyConfig	Get the public key configuration information (GET /2019-03-26/public-key/<Id>/config).	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetStreamingDistribution	Get the information about an RTMP distribution (GET /2019-03-26/streaming-distribution/<STREAMING_DISTRIBUTION_ID>).	Read	streaming-distribution* (p. 799)		
GetStreamingDistributionConfig	Get the configuration information about a streaming distribution (GET /2019-03-26/streaming-distribution/<STREAMING_DISTRIBUTION_ID>/config).	Read	streaming-distribution* (p. 799)		
ListCloudFrontOriginAccessIdentities	List your CloudFront origin access identities (GET /2019-03-26/origin-access-identity/cloudfront?Marker=<MARKER>&MaxItems=<MAX_ITEMS>).	List			
ListDistributions	List the distributions associated with your AWS account (GET /2019-03-26/distribution?Marker=<MARKER>&MaxItems=<MAX_ITEMS>).	List			
ListDistributionsByWebACL	List the distributions associated with your AWS account with given AWS WAF web ACL (GET /2019-03-26/distributionsByWebACLId/<WEB_ACL_ID>?Marker=<MARKER>&MaxItems=<MAX_ITEMS>).	List			
ListFieldLevelEncryptionConfigurations	List all field-level encryption configurations that have been created in CloudFront for this account (GET /2019-03-26/field-level-encryption?Marker=<MARKER>&MaxItems=<MAX_ITEMS>).	List			
ListFieldLevelEncryptionProfiles	List all field-level encryption profiles that have been created in CloudFront for this account. (GET /2019-03-26/field-level-encryption-profile?Marker=<MARKER>&MaxItems=<MAX_ITEMS>).	List			
ListInvalidations	List your invalidation batches (GET /2019-03-26/distribution/<STREAMING_DISTRIBUTION_ID>/invalidation?Marker=<MARKER>&MaxItems=<MAX_ITEMS>).	List	distribution* (p. 799)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListPublicKeys	List all public keys that have been added to CloudFront for this account. (GET /2019-03-26/public-key? Marker=<MARKER>&MaxItems=<MAX_ITEMS>).	List			
ListStreamingDistribution	List your RTMP distributions (GET /2019-03-26/ streaming-distribution? Marker=<MARKER>&MaxItems=<MAX_ITEMS>).	List			
ListTagsForResource	List tags for a CloudFront resource (GET /2019-03-26/tagging? Resource=<RESOURCE>).	Read	distribution (p. 799)		
	streaming-distribution (p. 799)				
TagResource	Add tags to a CloudFront resource (POST /2019-03-26/ tagging?Operation=Tag? Resource=<RESOURCE>).	Tagging	distribution (p. 799)		
streaming-distribution (p. 799)					
	aws:RequestTag / {\$TagKey} (p. 800)				
UntagResource	Remove tags from a CloudFront resource (POST /2019-03-26/ tagging?Operation=Untag? Resource=<RESOURCE>).	Tagging	distribution (p. 799)		
streaming-distribution (p. 799)					
	aws:TagKeys (p. 800)				
UpdateCloudFrontConfiguration	This action sets the configuration for a CloudFront origin access identity (PUT /2019-03-26/origin-access-identity/cloudfront/<OAI_ID>/config).	Write	origin-access-identity* (p. 799)		
UpdateDistribution	This action updates the configuration for a web distribution (PUT /2019-03-26/distribution/< DISTRIBUTION_ID >/config).	Write	distribution* (p. 799)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UpdateFieldLevelEncryptionConfig	This action updates a field-level encryption configuration. (PUT /2019-03-26/field-level-encryption/<Id>/config).	Write			
UpdateFieldLevelEncryptionProfile	This action updates a field-level encryption profile. (PUT /2019-03-26/field-level-encryption-profile/<Id>/config).	Write			
UpdatePublicKey	This action updates public key information. (PUT /2019-03-26/public-key/<Id>/config).	Write			
UpdateStreamingDistributionConfig	This action updates the configuration for an RTMP distribution (PUT /2019-03-26/streaming-distribution/<Distribution_ID>/config).	Write	streaming-distribution* (p. 799)		

Resource Types Defined by Amazon CloudFront

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 793\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
distribution	<code>arn:\${Partition}:cloudfront::\${Account}:distribution/\${DistributionId}</code>	aws:ResourceTag/\${TagKey} (p. 800)
streaming-distribution	<code>arn:\${Partition}:cloudfront::\${Account}:streaming-distribution/\${DistributionId}</code>	aws:ResourceTag/\${TagKey} (p. 800)
origin-access-identity	<code>arn:\${Partition}:cloudfront::\${Account}:origin-access-identity/\${Id}</code>	

Condition Keys for Amazon CloudFront

Amazon CloudFront defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Filters actions based on the presence of tag key-value pairs in the request	String
aws:ResourceTag/\${TagKey}	Filters actions based on tag key-value pairs attached to the resource	String
aws:TagKeys	Filters actions based on the presence of tag keys in the request	String

Actions, Resources, and Condition Keys for AWS CloudHSM

AWS CloudHSM (service prefix: `clouhdsm`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS CloudHSM \(p. 800\)](#)
- [Resource Types Defined by AWS CloudHSM \(p. 803\)](#)
- [Condition Keys for AWS CloudHSM \(p. 803\)](#)

Actions Defined by AWS CloudHSM

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AddTagsToResource	Adds or overwrites one or more tags for the specified AWS CloudHSM resource	Tagging			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CopyBackupToRegion	Creates a copy of a backup in the specified region	Write	backup* (p. 803)		
			aws:RequestTag/ \${TagKey} (p. 803)	aws:TagKeys (p. 804)	
CreateCluster	Creates a new AWS CloudHSM cluster	Write	backup (p. 803)		
				aws:RequestTag/ \${TagKey} (p. 803)	aws:TagKeys (p. 804)
CreateHapg	Creates a high-availability partition group	Write			
CreateHsm	Creates a new hardware security module (HSM) in the specified AWS CloudHSM cluster	Write	cluster* (p. 803)		
CreateLunaClient	Creates an HSM client	Write			
DeleteBackup	Deletes the specified CloudHSM backup	Write	backup* (p. 803)		
DeleteCluster	Deletes the specified AWS CloudHSM cluster	Write	cluster* (p. 803)		
DeleteHapg	Deletes a high-availability partition group	Write			
DeleteHsm	Deletes the specified HSM	Write			
DeleteLunaClient	Deletes a client	Write			
DescribeBackups	Gets information about backups of AWS CloudHSM clusters	Read			
DescribeClusters	Gets information about AWS CloudHSM clusters	Read			
DescribeHapg	Retrieves information about a high-availability partition group	Read			
DescribeHsm	Retrieves information about an HSM. You can identify the HSM by its ARN or its serial number	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeLunaClient	Retrieves information about an HSM client	Read			
GetConfig	Gets the configuration files necessary to connect to all high availability partition groups the client is associated with	Read			
InitializeCluster	Claims an AWS CloudHSM cluster	Write	cluster* (p. 803)		
ListAvailableZone	Lists the Availability Zones that have available AWS CloudHSM capacity	List			
ListHapgs	Lists the high-availability partition groups for the account	List			
ListHsms	Retrieves the identifiers of all of the HSMs provisioned for the current customer	List			
ListLunaClients	Lists all of the clients	List			
ListTags	Gets a list of tags for the specified AWS CloudHSM cluster	Read	backup (p. 803)		
			cluster (p. 803)		
ListTagsForResource	Returns a list of all tags for the specified AWS CloudHSM resource	Read			
ModifyHapg	Modifies an existing high-availability partition group	Write			
ModifyHsm	Modifies an HSM	Write			
ModifyLunaClient	Modifies the certificate used by the client	Write			
RemoveTagsFromResource	Removes one or more tags from the specified AWS CloudHSM resource	Tagging			
RestoreBackup	Restores the specified CloudHSM backup	Write	backup* (p. 803)		
TagResource	Adds or overwrites one or more tags for the specified AWS CloudHSM cluster	Tagging	backup (p. 803)		
			cluster (p. 803)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:RequestTag/\${TagKey} (p. 803) aws:TagKeys (p. 804)	
UntagResource	Removes the specified tag or tags from the specified AWS CloudHSM cluster	Tagging	backup (p. 803)		
			cluster (p. 803)		
				aws:TagKeys (p. 804)	

Resource Types Defined by AWS CloudHSM

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 800\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
backup	<code>arn:\${Partition}:clouhdhsm:\${Region}:\${Account}:backup/\${CloudHsmBackupInstanceName}</code>	aws:ResourceTag/\${TagKey} (p. 804)
cluster	<code>arn:\${Partition}:clouhdhsm:\${Region}:\${Account}:cluster/\${CloudHsmClusterInstanceName}</code>	aws:ResourceTag/\${TagKey} (p. 804)

Condition Keys for AWS CloudHSM

AWS CloudHSM defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Filters actions based on the presence of tag key-value pairs in the request	String

Condition Keys	Description	Type
aws:ResourceTag/\${TagKey}	Filters actions based on tag key-value pairs attached to the resource	String
aws:TagKeys	Filters actions based on the presence of tag keys in the request	String

Actions, Resources, and Condition Keys for Amazon CloudSearch

Amazon CloudSearch (service prefix: `cloudsearch`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon CloudSearch \(p. 804\)](#)
- [Resource Types Defined by Amazon CloudSearch \(p. 806\)](#)
- [Condition Keys for Amazon CloudSearch \(p. 807\)](#)

Actions Defined by Amazon CloudSearch

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AddTags	Attaches resource tags to an Amazon CloudSearch domain.	Tagging	domain* (p. 807)		
BuildSuggesters	Indexes the search suggestions.	Write	domain* (p. 807)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateDomain	Creates a new search domain.	Write	domain* (p. 807)		
DefineAnalysisScheme	Configures an analysis scheme that can be applied to a text or text-array field to define language-specific text processing options.	Write	domain* (p. 807)		
DefineExpression	Configures an Expression for the search domain.	Write	domain* (p. 807)		
DefineIndexField	Configures an IndexField for the search domain.	Write	domain* (p. 807)		
DefineSuggester	Configures a suggester for a domain.	Write	domain* (p. 807)		
DeleteAnalysisScheme	Deletes an analysis scheme.	Write	domain* (p. 807)		
DeleteDomain	Permanently deletes a search domain and all of its data.	Write	domain* (p. 807)		
DeleteExpression	Removes an Expression from the search domain.	Write	domain* (p. 807)		
DeleteIndexField	Removes an IndexField from the search domain.	Write	domain* (p. 807)		
DeleteSuggester	Deletes a suggester.	Write	domain* (p. 807)		
DescribeAnalysisScheme	Gets the analysis schemes configured for a domain.	Read	domain* (p. 807)		
DescribeAvailabilityConfig	Gets the availability options configured for a domain.	Read	domain* (p. 807)		
DescribeDomainEndpointConfig	Gets the domain endpoint options configured for a domain.	Read	domain* (p. 807)		
DescribeDomains	Gets information about the search domains owned by this account.	List	domain* (p. 807)		
DescribeExpressions	Gets the expressions configured for the search domain.	Read	domain* (p. 807)		
DescribeIndexFields	Gets information about the index fields configured for the search domain.	Read	domain* (p. 807)		
DescribeScalingParameters	Gets the scaling parameters configured for a domain.	Read	domain* (p. 807)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeServiceAccessPolicies	Gets information about the access policies that control access to the domain's document and search endpoints.	Read	domain* (p. 807)		
DescribeSuggester	Gets the suggesters configured for a domain.	Read	domain* (p. 807)		
IndexDocuments	Tells the search domain to start indexing its documents using the latest indexing options.	Write	domain* (p. 807)		
ListDomainNames	Lists all search domains owned by an account.	List	domain* (p. 807)		
ListTags	Displays all of the resource tags for an Amazon CloudSearch domain.	Read	domain* (p. 807)		
RemoveTags	Removes the specified resource tags from an Amazon ES domain.	Tagging	domain* (p. 807)		
UpdateAvailabilityOptions	Configures the availability options for a domain.	Write	domain* (p. 807)		
UpdateDomainEndpointOptions	Configures the domain endpoint options for a domain.	Write	domain* (p. 807)		
UpdateScalingParameters	Configures scaling parameters for a domain.	Write	domain* (p. 807)		
UpdateServiceAccessPolicies	Configures the access rules that control access to the domain's document and search endpoints.	Permissions management	domain* (p. 807)		
document [permission only]	Allows access to the document service operations.	Write	domain (p. 807)		
search [permission only]	Allows access to the search operations.	Read	domain (p. 807)		
suggest [permission only]	Allows access to the suggest operations.	Read	domain (p. 807)		

Resource Types Defined by Amazon CloudSearch

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 804\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you

can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Note

For information about using Amazon CloudSearch resource ARNs in an IAM policy, see [Amazon CloudSearch ARNs](#) in the *Amazon CloudSearch Developer Guide*.

Resource Types	ARN	Condition Keys
domain	arn:\${Partition}:cloudsearch:\${Region}:\${Account}:domain/\${DomainName}	

Condition Keys for Amazon CloudSearch

CloudSearch has no service-specific context keys that can be used in the Condition element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for AWS CloudTrail

AWS CloudTrail (service prefix: `cloudtrail`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS CloudTrail \(p. 807\)](#)
- [Resource Types Defined by AWS CloudTrail \(p. 809\)](#)
- [Condition Keys for AWS CloudTrail \(p. 809\)](#)

Actions Defined by AWS CloudTrail

You can specify the following actions in the Action element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the Resource element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AddTags	Grants permission to add one or more tags to a trail, up to a limit of 10	Tagging	trail* (p. 809)		
CreateTrail	Grants permission to create a trail that specifies the settings for delivery of log data to an Amazon S3 bucket	Write	trail* (p. 809)		s3:PutObject
DeleteTrail	Grants permission to delete a trail	Write	trail* (p. 809)		
DescribeTrails	Grants permission to list settings for the trails associated with the current region for your account	Read			
GetEventSelectors	Grants permission to list settings for event selectors configured for a trail	Read	trail* (p. 809)		
GetInsightSelectors	Grants permission to list CloudTrail Insights selectors that are configured for a trail	Read	trail* (p. 809)		
GetTrail	Grants permission to list settings for the trail	Read			
GetTrailStatus	Grants permission to retrieve a JSON-formatted list of information about the specified trail	Read	trail* (p. 809)		
ListPublicKeys	Grants permission to list the public keys whose private keys were used to sign trail digest files within a specified time range	Read			
ListTags	Grants permission to list the tags for trails in the current region	Read	trail* (p. 809)		
ListTrails	Grants permission to list trails associated with the current region for your account	List			
LookupEvents	Grants permission to look up API activity events captured by CloudTrail that create, update, or delete resources in your account	Read			
PutEventSelectors	Grants permission to create and update event selectors for a trail	Write	trail* (p. 809)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
PutInsightSelectors	Grants permission to create and update CloudTrail Insights selectors for a trail	Write	trail* (p. 809)		
RemoveTags	Grants permission to remove tags from a trail	Tagging	trail* (p. 809)		
StartLogging	Grants permission to start the recording of AWS API calls and log file delivery for a trail	Write	trail* (p. 809)		
StopLogging	Grants permission to stop the recording of AWS API calls and log file delivery for a trail	Write	trail* (p. 809)		
UpdateTrail	Grants permission to update the settings that specify delivery of log files	Write	trail* (p. 809)		

Resource Types Defined by AWS CloudTrail

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 807\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Note

For policies that control access to CloudTrail actions, the `Resource` element is always set to `"*"`. For information about using resource ARNs in an IAM policy, see [Granting Custom Permissions](#) in the *AWS CloudTrail User Guide*.

Resource Types	ARN	Condition Keys
trail	<code>arn:\${Partition}:cloudtrail:\${Region}: \${Account}:trail/\${TrailName}</code>	

Condition Keys for AWS CloudTrail

CloudTrail has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Amazon CloudWatch

Amazon CloudWatch (service prefix: `cloudwatch`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).

- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon CloudWatch \(p. 810\)](#)
- [Resource Types Defined by Amazon CloudWatch \(p. 812\)](#)
- [Condition Keys for Amazon CloudWatch \(p. 813\)](#)

Actions Defined by Amazon CloudWatch

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteAlarms	Deletes all specified alarms. In the event of an error, no alarms are deleted	Write	alarm* (p. 813)		
DeleteAnomalyDetectionModel	Deletes the specified anomaly detection model from your account.	Write			
DeleteDashboard	Deletes all CloudWatch dashboards that you specify	Write	dashboard* (p. 813)		
DeleteInsightRule	Grants permission to delete a collection of insight rules.	Write	insight-rule* (p. 813)		
DescribeAlarmHistory	Retrieves history for the specified alarm	Read	alarm* (p. 813)		
DescribeAlarms	Retrieves alarms with the specified names	Read	alarm* (p. 813)		
DescribeAlarmsForMetric	Retrieves all alarms for a single metric	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeAnomalyModels	Lists the anomaly detection models that you have created in your account.	Read			
DescribeInsightRules	Grants permission to describe all insight rules, currently owned by the user's account.	Read			
DisableAlarmActions	Disables actions for the specified alarms	Write	alarm* (p. 813)		
DisableInsightRules	Grants permission to disable a collection of insight rules.	Write	insight-rule* (p. 813)		
EnableAlarmActions	Enables actions for the specified alarms	Write	alarm* (p. 813)		
EnableInsightRules	Grants permission to enable a collection of insight rules.	Write	insight-rule* (p. 813)		
GetDashboard	Displays the details of the CloudWatch dashboard you specify	Read	dashboard* (p. 813)		
GetInsightRuleReport	Grants permission to return the top-N report of unique contributors over a time range for a given insight rule.	Read	insight-rule* (p. 813)		
GetMetricData	Required to retrieve batch amounts of CloudWatch metric data and perform metric math on retrieved data	Read			
GetMetricStatistics	Gets statistics for the specified metric	Read			
GetMetricWidgetDefinition	Required to retrieve snapshots of metric widgets	Read			
ListDashboards	Returns a list of all CloudWatch dashboards in your account	List			
ListMetrics	Returns a list of valid metrics stored for the AWS account owner	List			
ListTagsForResource	This action lists tags for an Amazon CloudWatch resource.	List	alarm* (p. 813)		
PutAnomalyDetectionModel	Creates or updates an anomaly detection model for a CloudWatch metric.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
PutDashboard	Creates a CloudWatch dashboard, or updates an existing dashboard if it already exists	Write	dashboard* (p. 813)		
PutInsightRule	Grants permission to create a new insight rule or replace an existing insight rule.	Write	insight-rule* (p. 813)		
PutMetricAlarm	Creates or updates an alarm and associates it with the specified Amazon CloudWatch metric	Write	alarm* (p. 813)		
				aws:RequestTag/\${TagKey} (p. 813)	
PutMetricData	Publishes metric data points to Amazon CloudWatch	Write		cloudwatch:namespace (p. 813)	
SetAlarmState	Temporarily sets the state of an alarm for testing purposes	Write	alarm* (p. 813)		
TagResource	This action tags an Amazon CloudWatch resource.	Tagging	alarm* (p. 813)		
				aws:TagKeys (p. 813)	
UntagResource	This action removes a tag from an Amazon CloudWatch resource.	Tagging	alarm* (p. 813)		
				aws:TagKeys (p. 813)	

Resource Types Defined by Amazon CloudWatch

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 810\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
alarm	arn:\${Partition}:cloudwatch:\${Region}: \${Account}:alarm:\${AlarmName}	aws:ResourceTag/\${TagKey} (p. 813)
dashboard	arn:\${Partition}:cloudwatch:: \${Account}:dashboard/\${DashboardName}	
insight-rule	arn:\${Partition}:cloudwatch:\${Region}: \${Account}:insight-rule/\${InsightRuleName}	

Condition Keys for Amazon CloudWatch

Amazon CloudWatch defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Filters actions based on the allowed set of values for each of the tags	String
aws:ResourceTag/\${TagKey}	Filters actions based on tag-value associated with the resource	String
aws:TagKeys	Filters actions based on the presence of mandatory tags in the request	String
cloudwatch:namespacevalues	Filters actions based on the presence of optional namespace	String

Actions, Resources, and Condition Keys for CloudWatch Application Insights

CloudWatch Application Insights (service prefix: `applicationinsights`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by CloudWatch Application Insights \(p. 814\)](#)
- [Resource Types Defined by CloudWatch Application Insights \(p. 815\)](#)
- [Condition Keys for CloudWatch Application Insights \(p. 815\)](#)

Actions Defined by CloudWatch Application Insights

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateApplication	Creates an application from a resource group	Write			
CreateComponent	Creates a component from a group of resources	Write			
DeleteApplication	Deletes an application	Write			
DeleteComponent	Deletes a component	Write			
DescribeApplication	Describes an application	Read			
DescribeComponent	Describes a component	Read			
DescribeComponentConfiguration	Describes a component configuration	Read			
DescribeComponentRecommendation	Describe the recommended application component configuration	Read			
DescribeObservation	Describes an observation	Read			
DescribeProblem	Describes a problem	Read			
DescribeProblems	Describes the observation in a problem	Read			
ListApplications	Lists all applications	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListComponents	List an application's components	List			
ListProblems	Lists the problems in an application	List			
UpdateApplication	Updates an application	Write			
UpdateComponent	Updates a component	Write			
UpdateComponentConfiguration	Updates a component configuration	Write			

Resource Types Defined by CloudWatch Application Insights

CloudWatch Application Insights does not support specifying a resource ARN in the Resource element of an IAM policy statement. To allow access to CloudWatch Application Insights, specify “Resource”: “*” in your policy.

Condition Keys for CloudWatch Application Insights

CloudWatch Application Insights has no service-specific context keys that can be used in the Condition element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Amazon CloudWatch Logs

Amazon CloudWatch Logs (service prefix: logs) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon CloudWatch Logs \(p. 815\)](#)
- [Resource Types Defined by Amazon CloudWatch Logs \(p. 820\)](#)
- [Condition Keys for Amazon CloudWatch Logs \(p. 820\)](#)

Actions Defined by Amazon CloudWatch Logs

You can specify the following actions in the Action element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually

allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the Resource element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AssociateKmsKey	Associates the specified AWS Key Management Service (AWS KMS) customer master key (CMK) with the specified log group.	Write	log-group* (p. 820)		
CancelExportTask	Cancels an export task if it is in PENDING or RUNNING state	Write			
CreateExportTask	Creates an ExportTask which allows you to efficiently export data from a Log Group to your Amazon S3 bucket	Write	log-group* (p. 820)		
CreateLogDelivery [permission only]	Creates the log delivery	Write			
CreateLogGroup	Creates a new log group with the specified name	Write			
CreateLogStream	Creates a new log stream with the specified name	Write	log-group* (p. 820)		
DeleteDestination	Deletes the destination with the specified name and eventually disables all the subscription filters that publish to it	Write			
DeleteLogDelivery [permission only]	Deletes the log delivery information for specified log delivery	Write			
DeleteLogGroup	Deletes the log group with the specified name and permanently deletes all the archived log events associated with it	Write	log-group* (p. 820)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteLogStream	Deletes a log stream and permanently deletes all the archived log events associated with it	Write	log-stream* (p. 820)		
DeleteMetricFilter	Deletes a metric filter associated with the specified log group	Write	log-group* (p. 820)		
DeleteResourcePolicy	Deletes a resource policy from this account	Write			
DeleteRetentionPolicy	Deletes the retention policy of the specified log group	Write	log-group* (p. 820)		
DeleteSubscriptionFilter	Deletes a subscription filter associated with the specified log group	Write	log-group* (p. 820)		
DescribeDestinations	Returns all the destinations that are associated with the AWS account making the request	List			
DescribeExportTasks	Returns all the export tasks that are associated with the AWS account making the request	List			
DescribeLogGroups	Returns all the log groups that are associated with the AWS account making the request	List	log-group* (p. 820)		
DescribeLogStreams	Returns all the log streams that are associated with the specified log group	List	log-group* (p. 820)		
DescribeMetricFilters	Returns all the metrics filters associated with the specified log group	List	log-group* (p. 820)		
DescribeQueries	Returns a list of CloudWatch Logs Insights queries that are scheduled, executing, or have been executed recently in this account. You can request all queries, or limit it to queries of a specific log group or queries with a certain status.	List			
DescribeResourcePolicies	Return all the resource policies for this account.	List			
DescribeSubscriptionFilters	Returns all the subscription filters associated with the specified log group	List	log-group* (p. 820)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DisassociateKmsKey	Disassociates the associated AWS Key Management Service (AWS KMS) customer master key (CMK) from the specified log group	Write	log-group* (p. 820)		
FilterLogEvents	Retrieves log events, optionally filtered by a filter pattern from the specified log group	Read	log-group* (p. 820)		
GetLogDelivery [permission only]	Gets the log delivery information for specified log delivery	Read			
GetLogEvents	Retrieves log events from the specified log stream	Read	log-stream* (p. 820)		
GetLogGroupFields	Returns a list of the fields that are included in log events in the specified log group, along with the percentage of log events that contain each field. The search is limited to a time period that you specify.	Read	log-group* (p. 820)		
GetLogRecord	Retrieves all the fields and values of a single log event. All fields are retrieved, even if the original query that produced the logRecordPointer retrieved only a subset of fields. Fields are returned as field name/field value pairs.	Read			
GetQueryResults	Returns the results from the specified query. If the query is in progress, partial results of that current execution are returned. Only the fields requested in the query are returned.	Read			
ListLogDeliveries [permission only]	Lists all the log deliveries for specified account and/or log source	List			
ListTagsLogGroup	Lists the tags for the specified log group	List	log-group* (p. 820)		
PutDestination	Creates or updates a Destination	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
PutDestinationPolicy	Creates or updates an access policy associated with an existing Destination	Write			
PutLogEvents	Uploads a batch of log events to the specified log stream	Write	log-stream* (p. 820)		
PutMetricFilter	Creates or updates a metric filter and associates it with the specified log group	Write	log-group* (p. 820)		
PutResourcePolicy	Creates or updates a resource policy allowing other AWS services to put log events to this account	Write			
PutRetentionPolicy	Sets the retention of the specified log group	Write	log-group* (p. 820)		
PutSubscriptionFilter	Creates or updates a subscription filter and associates it with the specified log group	Write	log-group* (p. 820)		
StartQuery	Schedules a query of a log group using CloudWatch Logs Insights. You specify the log group and time range to query, and the query string to use.	Read	log-group* (p. 820)		
StopQuery	Stops a CloudWatch Logs Insights query that is in progress. If the query has already ended, the operation returns an error indicating that the specified query is not running.	Read			
TagLogGroup	Adds or updates the specified tags for the specified log group	Write	log-group* (p. 820)		
TestMetricFilter	Tests the filter pattern of a metric filter against a sample of log event messages	Read			
UntagLogGroup	Removes the specified tags from the specified log group	Write	log-group* (p. 820)		
UpdateLogDelivery [permission only]	Updates the log delivery information for specified log delivery	Write			

Resource Types Defined by Amazon CloudWatch Logs

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 815\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
log-group	<code>arn:\${Partition}:logs:\${Region}:\${Account}:log-group:\${LogGroupName}</code>	
log-stream	<code>arn:\${Partition}:logs:\${Region}:\${Account}:log-group:\${LogGroupName}:log-stream:\${LogStreamName}</code>	

Condition Keys for Amazon CloudWatch Logs

CloudWatch Logs has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Amazon CloudWatch Synthetics

Amazon CloudWatch Synthetics (service prefix: `synthetics`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon CloudWatch Synthetics \(p. 820\)](#)
- [Resource Types Defined by Amazon CloudWatch Synthetics \(p. 821\)](#)
- [Condition Keys for Amazon CloudWatch Synthetics \(p. 822\)](#)

Actions Defined by Amazon CloudWatch Synthetics

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your

policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateAlarmSettings	Creates alarm settings.	Write			
CreateCanary	Create a canary.	Write			
DeleteCanary	Deletes a canary. Amazon Synthetics deletes all the resources except for the Lambda function and the CloudWatch Alarms if you created one.	Write	canary* (p. 821)		
DescribeCanaries	Returns information of all canaries or a canary.	Read	canary (p. 821)		
DescribeTestRuns	Returns information about all the test runs associated with a canary.	Read	canary (p. 821)		
GetAlarmSettings	Returns information for an alarm setting.	Read			
StartCanary	Starts a canary, so that Amazon Synthetics starts monitoring a website.	Write	canary* (p. 821)		
StopCanary	Stops a canary.	Write	canary* (p. 821)		
UpdateCanary	Updates a canary.	Write	canary* (p. 821)		

Resource Types Defined by Amazon CloudWatch Synthetics

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 820\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
canary	arn:\${Partition}:synthetics::\${Account}:canary:\${CanaryName}	

Condition Keys for Amazon CloudWatch Synthetics

CloudWatch Synthetics has no service-specific context keys that can be used in the Condition element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for AWS Code Signing for Amazon FreeRTOS

AWS Code Signing for Amazon FreeRTOS (service prefix: `signer`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Code Signing for Amazon FreeRTOS \(p. 822\)](#)
- [Resource Types Defined by AWS Code Signing for Amazon FreeRTOS \(p. 823\)](#)
- [Condition Keys for AWS Code Signing for Amazon FreeRTOS \(p. 824\)](#)

Actions Defined by AWS Code Signing for Amazon FreeRTOS

You can specify the following actions in the Action element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the Resource element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CancelSigningProfile	Cancels a signing profile.	Write	signing-profile* (p. 824)		
DescribeSigningJob	Describe a signing job.	Read	signing-job* (p. 824)		
GetSigningPlatform	Retrieves a signing platform.	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetSigningProfile	Retrieves a signing profile.	Read	signing-profile* (p. 824)		
ListSigningJobs	List signing jobs.	List			
ListSigningPlatforms	List all signing platforms.	List			
ListSigningProfile	List all signing profile associated with the account.	List			
ListTagsForResource	Lists the tags associated with the Signing Profile resource.	List	signing-profile* (p. 824)		
PutSigningProfile	Creates a new signing profile if not exists.	Write		aws:RequestTag/\${TagKey} (p. 824) aws:TagKeys (p. 824)	
StartSigningJob	Starts a code signing request.	Write	signing-profile* (p. 824)		
TagResource	Adds one or more tags to an Signing Profile resource	Tagging	signing-profile* (p. 824)		
				aws:TagKeys (p. 824) aws:RequestTag/\${TagKey} (p. 824)	
UntagResource	Removes one or more tags from an Signing Profile resource	Tagging	signing-profile* (p. 824)		
				aws:TagKeys (p. 824) aws:RequestTag/\${TagKey} (p. 824)	

Resource Types Defined by AWS Code Signing for Amazon FreeRTOS

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 822\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you

can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
signing-profile	arn:\${Partition}:signer:\${Region}::/signing-profiles/\${profileName}	aws:ResourceTag/\${TagKey} (p. 824)
signing-job	arn:\${Partition}:signer:\${Region}::/signing-jobs/\${jobId}	

Condition Keys for AWS Code Signing for Amazon FreeRTOS

AWS Code Signing for Amazon FreeRTOS defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Filters create requests based on the allowed set of values for each of the tags.	String
aws:ResourceTag/\${TagKey}	Filters actions based on tag-value associated with the resource.	String
aws:TagKeys	Filters create requests based on the presence of mandatory tags in the request.	String

Actions, Resources, and Condition Keys for AWS CodeBuild

AWS CodeBuild (service prefix: `codebuild`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS CodeBuild \(p. 825\)](#)
- [Resource Types Defined by AWS CodeBuild \(p. 829\)](#)
- [Condition Keys for AWS CodeBuild \(p. 829\)](#)

Actions Defined by AWS CodeBuild

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
BatchDeleteBuilds	Deletes one or more builds.	Write	project* (p. 829)		
BatchGetBuilds	Gets information about one or more builds.	Read	project* (p. 829)		
BatchGetProjects	Gets information about one or more build projects.	Read	project* (p. 829)		
BatchGetReportGroups	Returns an array of ReportGroup objects that are specified by the input reportGroupArns parameter.	Read	report-group* (p. 829)		
BatchGetReports	Returns an array of the Report objects specified by the input reportArns parameter.	Read	report-group* (p. 829)		
BatchPutTestCase [permission only]	Adds or updates information about a report.	Write	report-group* (p. 829)		
CreateProject	Creates a build project.	Write	project* (p. 829)		
			aws:RequestTag/\${TagKey} (p. 829)		
			aws:TagKeys (p. 829)		
CreateReport [permission only]	Creates a report. A report is created when tests specified in the buildspec file for a report	Write	report-group* (p. 829)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
	groups run during the build of a project.				
CreateReportGroup	Creates a report group.	Write	report-group* (p. 829)		
CreateWebhook	For an existing AWS CodeBuild build project that has its source code stored in a GitHub or Bitbucket repository, enables AWS CodeBuild to start rebuilding the source code every time a code change is pushed to the repository.	Write	project* (p. 829)		
DeleteOAuthToken [permission only]	Deletes an OAuth token from a connected third-party OAuth provider. Only used in the AWS CodeBuild console.	Write			
DeleteProject	Deletes a build project.	Write	project* (p. 829)		
DeleteReport	Deletes a report.	Write	report-group* (p. 829)		
DeleteReportGroup	Deletes a report group.	Write	report-group* (p. 829)		
DeleteResourcePolicy	Deletes a resource policy for the associated project or report group.	Permissions management	project (p. 829)		
			report-group (p. 829)		
DeleteSourceCredentials	Deletes a set of GitHub, GitHub Enterprise, or Bitbucket source credentials.	Write			
DeleteWebhook	For an existing AWS CodeBuild build project that has its source code stored in a GitHub or Bitbucket repository, stops AWS CodeBuild from rebuilding the source code every time a code change is pushed to the repository.	Write	project* (p. 829)		
DescribeTestCase	Returns an array of TestCase objects.	Read	report-group* (p. 829)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetResourcePolicy	Returns a resource policy for the specified project or report group.	Read	project (p. 829)		
			report-group (p. 829)		
ImportSourceCredentials	Imports the source repository credentials for an AWS CodeBuild project that has its source code stored in a GitHub, GitHub Enterprise, or Bitbucket repository.	Write			
InvalidateProjectCache	Resets the cache for a project.	Write	project* (p. 829)		
ListBuilds	Gets a list of build IDs, with each build ID representing a single build.	List			
ListBuildsForProject	Gets a list of build IDs for the specified build project, with each build ID representing a single build.	List	project* (p. 829)		
ListConnectedOAuthProviders [permission only]	Lists connected third-party OAuth providers. Only used in the AWS CodeBuild console.	List			
ListCuratedEnvironmentImages	Gets information about Docker images that are managed by AWS CodeBuild.	List			
ListProjects	Gets a list of build project names, with each build project name representing a single build project.	List			
ListReportGroups	Returns a list of report group ARNs. Each report group ARN represents one report group.	List			
ListReports	Returns a list of report ARNs. Each report ARN representing one report.	List			
ListReportsForReportGroup	Returns a list of report ARNs that belong to the specified report group. Each report ARN represents one report.	List	report-group* (p. 829)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListRepositories [permission only]	Lists source code repositories from a connected third-party OAuth provider. Only used in the AWS CodeBuild console.	List			
ListSharedProjects	Returns a list of project ARNs that have been shared with the requester. Each project ARN represents one project.	List			
ListSharedReportGroups	Returns a list of report group ARNs that have been shared with the requester. Each report group ARN represents one report group.	List			
ListSourceCredentials	Returns a list of <code>SourceCredentialsInfo</code> objects.	List			
PersistOAuthToken [permission only]	Saves an OAuth token from a connected third-party OAuth provider. Only used in the AWS CodeBuild console.	Write			
PutResourcePolicy	Creates a resource policy for the associated project or report group.	Permissions management	project (p. 829)		
			report-group (p. 829)		
StartBuild	Starts running a build.	Write	project* (p. 829)		
StopBuild	Attempts to stop running a build.	Write	project* (p. 829)		
UpdateProject	Changes the settings of an existing build project.	Write	project* (p. 829)		
				aws:RequestTag / \${TagKey} (p. 829)	
UpdateReport [permission only]	Updates information about a report.	Write	report-group* (p. 829)		
UpdateReportGroup	Changes the settings of an existing report group.			report-group* (p. 829)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UpdateWebhook	Updates the webhook associated with an AWS CodeBuild build project.	Write	project* (p. 829)		

Resource Types Defined by AWS CodeBuild

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 825\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
build	<code>arn:\${Partition}:codebuild:\${Region}: \${Account}:build/\${BuildId}</code>	
project	<code>arn:\${Partition}:codebuild:\${Region}: \${Account}:project/\${ProjectName}</code>	aws:ResourceTag/\${TagKey} (p. 829)
report-group	<code>arn:\${Partition}:codebuild:\${Region}: \${Account}:report-group/\${ReportGroupName}</code>	
report	<code>arn:\${Partition}:codebuild:\${Region}: \${Account}:report/\${ReportGroupName}: \${ReportId}</code>	

Condition Keys for AWS CodeBuild

AWS CodeBuild defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Filters actions based on the presence of tag key-value pairs in the request	String
aws:ResourceTag/\${TagKey}	Filters actions based on tag key-value pairs attached to the resource	String
aws:TagKeys	Filters actions based on the presence of tag keys in the request	String

Actions, Resources, and Condition Keys for AWS CodeCommit

AWS CodeCommit (service prefix: `codecommit`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS CodeCommit \(p. 830\)](#)
- [Resource Types Defined by AWS CodeCommit \(p. 839\)](#)
- [Condition Keys for AWS CodeCommit \(p. 839\)](#)

Actions Defined by AWS CodeCommit

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AssociateApprovalTemplateWithRepository	Grants permission to associate an approval template with a repository	Write	repository* (p. 839)		
BatchAssociateApprovalTemplateWithRepositories	Grants permission to associate an approval template with multiple repositories in a single operation	Write	repository* (p. 839)		
BatchDescribeMergeConflicts	Grants permission to get information about multiple merge conflicts when attempting to merge two commits using either the three-way merge or the squash merge option	Read	repository* (p. 839)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
BatchDisassociateApprovalRuleTemplateFromRepositories	Grants permission to remove the association between an Approval rule template and multiple repositories in a single operation	Write	repository* (p. 839)		
BatchGetCommit	Grants permission to get return information about one or more commits in an AWS CodeCommit repository.	Read	repository* (p. 839)		
BatchGetPullRequest [permission only]	Grants permission to return information about one or more pull requests in an AWS CodeCommit repository	Read	repository* (p. 839)		
BatchGetRepository	Grants permission to get information about multiple repositories	Read	repository* (p. 839)		
CancelUploadArchive [permission only]	Grants permission to cancel the uploading of an archive to a pipeline in AWS CodePipeline	Read	repository* (p. 839)		
CreateApprovalRuleTemplate	Grants permission to create an Approval rule template that will automatically create approval rules in pull requests that match the conditions defined in the template; does not grant permission to create approval rules for individual pull requests	Write			
CreateBranch	Grants permission to create a branch in an AWS CodeCommit repository with this API; does not control Git create branch actions	Write	repository* (p. 839)		
					codecommit:References (p. 840)
CreateCommit	Grants permission to add, copy, move or update single or multiple files in a branch in an AWS CodeCommit repository, and generate a commit for the changes in the specified branch.	Write	repository* (p. 839)		
					codecommit:References (p. 840)
CreatePullRequest	Grants permission to create a pull request in the specified repository	Write	repository* (p. 839)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreatePullRequestApprovalRule	Grants permission to create an approval rule specific to an individual pull request; does not grant permission to create approval rule templates	Write	repository* (p. 839)		
CreateRepository	Grants permission to create an AWS CodeCommit repository	Write	repository* (p. 839)		
				aws:RequestTag/\${TagKey} (p. 839) aws:TagKeys (p. 840)	
CreateUnreferencedMergeCommit	Grants permission to create a merge commit that contains the result of merging two commits using either the three-way or the squash merge option; does not control Git merge actions	Write	repository* (p. 839)		
				codecommit:References (p. 840)	
DeleteApprovalRuleTemplate	Grants permission to delete an approval rule template	Write			
DeleteBranch	Grants permission to delete a branch in an AWS CodeCommit repository with this API; does not control Git delete branch actions	Write	repository* (p. 839)		
				codecommit:References (p. 840)	
DeleteCommentContent	Grants permission to delete the content of a comment made on a change, file, or commit in a repository	Write	repository* (p. 839)		
DeleteFile	Grants permission to delete a specified file from a specified branch	Write	repository* (p. 839)		
				codecommit:References (p. 840)	
DeletePullRequestApprovalRule	Grants permission to delete an approval rule created for a pull request if the rule was not created by an approval rule template	Write	repository* (p. 839)		
DeleteRepository	Grants permission to delete an AWS CodeCommit repository	Write	repository* (p. 839)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeMergeConflict	Grants permission to get information about specific merge conflicts when attempting to merge two commits using either the three-way or the squash merge option	Read	repository* (p. 839)		
DescribePullRequestEvents	Grants permission to return information about one or more pull request events	Read	repository* (p. 839)		
DisassociateApprovalRuleTemplateFromRepository	Grants permission to remove the association between an approval rule template and a repository	Write	repository* (p. 839)		
EvaluatePullRequestMergeability	Grants permission to evaluate whether a pull request is mergable based on its current approval state and approval rule requirements	Read	repository* (p. 839)		
GetApprovalRuleTemplate	Grants permission to return information about an approval rule template	Read			
GetBlob	Grants permission to view the encoded content of an individual file in an AWS CodeCommit repository from the AWS CodeCommit console	Read	repository* (p. 839)		
GetBranch	Grants permission to get details about a branch in an AWS CodeCommit repository with this API; does not control Git branch actions	Read	repository* (p. 839)		
GetComment	Grants permission to get the content of a comment made on a change, file, or commit in a repository	Read	repository* (p. 839)		
GetCommentsForComparison	Grants permission to get information about comments made on the comparison between two commits	Read	repository* (p. 839)		
GetCommentsForPullRequest	Grants permission to get comments made on a pull request	Read	repository* (p. 839)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetCommit	Grants permission to return information about a commit, including commit message and committer information, with this API; does not control Git log actions	Read	repository* (p. 839)		
GetCommitHistory [permission only]	Grants permission to get information about the history of commits in a repository	Read	repository* (p. 839)		
GetCommitsFromMergeInformation [permission only]	Grants permission to get information about the difference between commits in the context of a potential merge	Read	repository* (p. 839)		
GetDifferences	Grants permission to view information about the differences between valid commit specifiers such as a branch, tag, HEAD, commit ID, or other fully qualified reference	Read	repository* (p. 839)		
GetFile	Grants permission to return the base-64 encoded contents of a specified file and its metadata	Read	repository* (p. 839)		
GetFolder	Grants permission to return the contents of a specified folder in a repository	Read	repository* (p. 839)		
GetMergeCommit	Grants permission to get information about a merge commit created by one of the merge options for pull requests that creates merge commits. Not all merge options create merge commits. This permission does not control Git merge actions	Read	repository* (p. 839)		
GetMergeConflict	Grants permission to get information about merge conflicts between the before and after commit IDs for a pull request in a repository		repository* (p. 839)	codecommit:References (p. 840)	
GetMergeOptions	Grants permission to get information about merge options for pull requests that can be used to merge two commits; does not control Git merge actions	Read	repository* (p. 839)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetObjectIdentifier [permission only]	Grants permission to resolve blobs, trees, and commits to their identifier	Read	repository* (p. 839)		
GetPullRequest	Grants permission to get information about a pull request in a specified repository	Read	repository* (p. 839)		
GetPullRequestApprovals	Grants permission to retrieve the current approvals on an inputted pull request	Read	repository* (p. 839)		
GetPullRequestOverrideState	Grants permission to retrieve the override state of a given pull request	Read	repository* (p. 839)		
GetReferences [permission only]	Grants permission to get details about references in an AWS CodeCommit repository; does not control Git reference actions	Read	repository* (p. 839)		
GetRepository	Grants permission to get information about an AWS CodeCommit repository	Read	repository* (p. 839)		
GetRepositoryTriggers	Grants permission to get information about triggers configured for a repository	Read	repository* (p. 839)		
GetTree [permission only]	Grants permission to view the contents of a specified tree in an AWS CodeCommit repository from the AWS CodeCommit console	Read	repository* (p. 839)		
GetUploadArchiveInformation [permission only]	Grants permission to get status information about an archive upload to a pipeline in AWS CodePipeline	Read	repository* (p. 839)		
GitPull [permission only]	Grants permission to pull information from an AWS CodeCommit repository to a local repo	Read	repository* (p. 839)		
GitPush [permission only]	Grants permission to push information from a local repo to an AWS CodeCommit repository	Write	repository* (p. 839)		codecommit:References (p. 840)

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListApprovalRuleTemplates	Grants permission to list all approval rule templates in an AWS Region for the AWS account	List			
ListAssociatedApprovalRuleTemplatesForRepository	Grants permission to list approval rule templates for a repository associated with a repository	List	repository* (p. 839)		
ListBranches	Grants permission to list branches for an AWS CodeCommit repository with this API; does not control Git branch actions	List	repository* (p. 839)		
ListPullRequests	Grants permission to list pull requests for a specified repository	List	repository* (p. 839)		
ListRepositories	Grants permission to list information about AWS CodeCommit repositories in the current Region for your AWS account	List			
ListRepositoriesForApprovalRuleTemplate	Grants permission to list repositories that are associated with an approval rule template	List			
ListTagsForResource	Grants permission to list the resource attached to a CodeCommit resource ARN	List	repository (p. 839)		
MergeBranchesByFastForward	Grants permission to merge two commits into the specified destination branch using the fast-forward merge option	Write	repository* (p. 839)		
					codecommit:References (p. 840)
MergeBranchesBySquash	Grants permission to merge two commits into the specified destination branch using the squash merge option	Write	repository* (p. 839)		
					codecommit:References (p. 840)
MergeBranchesByThreeWay	Grants permission to merge two commits into the specified destination branch using the three-way merge option	Write	repository* (p. 839)		
					codecommit:References (p. 840)

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
MergePullRequest	Grants permission to close a pull request and attempt to merge it into the specified destination branch for that pull request at the specified commit using the fast-forward merge option	Write	repository* (p. 839)		
				codecommit:References (p. 840)	
MergePullRequest	Grants permission to close a pull request and attempt to merge it into the specified destination branch for that pull request at the specified commit using the squash merge option	Write	repository* (p. 839)		
				codecommit:References (p. 840)	
MergePullRequest	Grants permission to close a pull request and attempt to merge it into the specified destination branch for that pull request at the specified commit using the three-way merge option	Write	repository* (p. 839)		
				codecommit:References (p. 840)	
OverridePullRequest	Grants permission to override all approval rules for a pull request, including approval rules created by a template	Write	repository* (p. 839)		
PostCommentForComparison	Grants permission to post a comment on the comparison between two commits	Write	repository* (p. 839)		
PostCommentForPullRequest	Grants permission to post a comment on a pull request	Write	repository* (p. 839)		
PostCommentReply	Grants permission to post a comment in reply to a comment on a comparison between commits or a pull request	Write	repository* (p. 839)		
PutFile	Grants permission to add or update a file in a branch in an AWS CodeCommit repository, and generate a commit for the addition in the specified branch	Write	repository* (p. 839)		
				codecommit:References (p. 840)	
PutRepositoryTrigger	Grants permission to create, update, or delete triggers for a repository	Write	repository* (p. 839)		
TagResource	Grants permission to attach resource tags to a CodeCommit resource ARN	Write	repository (p. 839)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:ResourceTag/ \${TagKey} (p. 840) aws:RequestTag/ \${TagKey} (p. 839) aws:TagKeys (p. 840)	
TestRepositoryTriggers	Grants permission to test the functionality of repository triggers by sending information to the trigger target	Write	repository* (p. 839)		
UntagResource	Grants permission to disassociate resource tags from a CodeCommit resource ARN	Write	repository (p. 839)		
			aws:TagKeys (p. 840)		
UpdateApprovalRuleContent	Grants permission to update the content of approval rule templates; does not grant permission to update content of approval rules created specifically for pull requests	Write			
UpdateApprovalRuleDescription	Grants permission to update the description of approval rule templates	Write			
UpdateApprovalRuleName	Grants permission to update the name of approval rule templates	Write			
UpdateComment	Grants permission to update the contents of a comment if the identity matches the identity used to create the comment	Write	repository* (p. 839)		
UpdateDefaultBranch	Grants permission to change the default branch in an AWS CodeCommit repository	Write	repository* (p. 839)		
UpdatePullRequestContentForApproval	Grants permission to update the content for approval rules created for a specific pull requests; does not grant permission to update approval rule content for rules created with an approval rule template	Write	repository* (p. 839)		
UpdatePullRequestApprovalState	Grants permission to update the approval state for pull requests	Write	repository* (p. 839)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UpdatePullRequestDescription	Grants permission to update the <code>description</code> of a pull request	Write	repository* (p. 839)		
UpdatePullRequestStatus	Grants permission to update the <code>status</code> of a pull request	Write	repository* (p. 839)		
UpdatePullRequestTitle	Grants permission to update the <code>title</code> of a pull request	Write	repository* (p. 839)		
UpdateRepositoryDescription	Grants permission to change the <code>description</code> of an AWS CodeCommit repository	Write	repository* (p. 839)		
UpdateRepositoryName	Grants permission to change the <code>name</code> of an AWS CodeCommit repository	Write	repository* (p. 839)		
UploadArchive [permission only]	Grants permission to the service role for AWS CodePipeline to upload repository changes into a pipeline	Write	repository* (p. 839)		

Resource Types Defined by AWS CodeCommit

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 830\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
repository	<code>arn:\${Partition}:codecommit:\${Region}: \${Account}:\${RepositoryName}</code>	aws:ResourceTag/\${TagKey} (p. 840)

Condition Keys for AWS CodeCommit

AWS CodeCommit defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Filters actions based on the presence of tag key-value pairs in the request	String

Condition Keys	Description	Type
<code>aws:ResourceTag/\${TagKey}</code>	Filters actions based on tag key-value pairs attached to the resource	String
<code>aws:TagKeys</code>	Filters actions based on the presence of tag keys in the request	String
<code>codecommit:Reference</code>	Filters access by Git reference to specified AWS CodeCommit actions	String

Actions, Resources, and Condition Keys for AWS CodeDeploy

AWS CodeDeploy (service prefix: `codedeploy`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS CodeDeploy \(p. 840\)](#)
- [Resource Types Defined by AWS CodeDeploy \(p. 845\)](#)
- [Condition Keys for AWS CodeDeploy \(p. 845\)](#)

Actions Defined by AWS CodeDeploy

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
<code>AddTagsToOnPremisesInstances</code>	Add tags to one or more on-premises instances.	Tagging	<code>instance*</code> (p. 845)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
BatchGetApplication	Gets information about one or more applications .	Read	application* (p. 845)		
BatchGetApplications	Get information about multiple applications associated with the IAM user.	Read	application* (p. 845)		
BatchGetDeploymentGroups	Get information about one or more deployment groups .	Read	deploymentgroup* (p. 845)		
BatchGetDeploymentInstances	Gets information about one or more instances that are part of a deployment group.	Read	deploymentgroup* (p. 845)		
BatchGetDeploymentTargets	Returns an array of one or more targets associated with a deployment. This method works with all compute types and should be used instead of the deprecated BatchGetDeploymentInstances . The maximum number of targets that can be returned is 25.	Read			
BatchGetDeployments	Get information about multiple deployments associated with the IAM user.	Read	deploymentgroup* (p. 845)		
BatchGetOnPremisesInstances	Get information about one or more on-premises instances .	Read	instance* (p. 845)		
ContinueDeployment	Starts the process of rerouting traffic from instances in the original environment to instances in the replacement environment without waiting for a specified wait time to elapse.	Write			
CreateApplication	Create an application associated with the IAM user.	Write	application* (p. 845)		
			aws:RequestTag/\${TagKey} (p. 845) aws:TagKeys (p. 845)		
CreateDeployment	Create a deployment for an application associated with the IAM user.	Write	deploymentgroup* (p. 845)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateDeploymentConfig	Create a custom deployment configuration associated with the IAM user.	Write	deploymentconfig* (p. 845)		
CreateDeploymentGroup	Create a deployment group for an application associated with the IAM user.	Write	deploymentgroup* (p. 845)		
				aws:RequestTag/\${TagKey} (p. 845)	
				aws:TagKeys (p. 845)	
DeleteApplication	Delete an application associated with the IAM user.	Write	application* (p. 845)		
DeleteDeploymentConfig	Delete a custom deployment configuration associated with the IAM user.	Write	deploymentconfig* (p. 845)		
DeleteDeploymentGroup	Delete a deployment group for an application associated with the IAM user.	Write	deploymentgroup* (p. 845)		
DeleteGitHubAccountConnection	Deletes a GitHub account connection.	Write			
DeregisterOnPremisesInstance	Deregister an on-premises instance	Write	instance* (p. 845)		
GetApplication	Get information about a single application associated with the IAM user.	List	application* (p. 845)		
GetApplicationRevision	Get information about a single application revision for an application associated with the IAM user.	List	application* (p. 845)		
GetDeployment	Get information about a single deployment to a deployment group for an application associated with the IAM user.	List	deploymentgroup* (p. 845)		
GetDeploymentConfig	Get information about a single deployment configuration associated with the IAM user.	List	deploymentconfig* (p. 845)		
GetDeploymentGroup	Get information about a single deployment group for an application associated with the IAM user.	List	deploymentgroup* (p. 845)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetDeploymentInstance	Get information about a single instance in a deployment associated with the IAM user.	List	deploymentgroup* (p. 845)		
GetDeploymentTarget	Returns information about a deployment target.	Read			
GetOnPremisesInstance	Get information about a single on-premises instance.	List	instance* (p. 845)		
ListApplicationRevisions	Get information about all application revisions for an application associated with the IAM user.	List	application* (p. 845)		
ListApplications	Get information about all applications associated with the IAM user.	List			
ListDeploymentConfigs	Get information about all deployment configurations associated with the IAM user.	List			
ListDeploymentGroups	Get information about all deployment groups for an application associated with the IAM user.	List	application* (p. 845)		
ListDeploymentInstances	Get information about all instances in a deployment associated with the IAM user.	List	deploymentgroup* (p. 845)		
ListDeploymentTargets	Returns an array of target hosts that are associated a deployment.	List			
ListDeployments	Get information about all deployments to a deployment group associated with the IAM user, or to get all deployments associated with the IAM user.	List	deploymentgroup* (p. 845)		
ListGitHubAccountConnections	Lists the names of stored connections to GitHub accounts.	List			
ListOnPremisesInstances	Get a list of one or more on-premises instance names.	List			
ListTagsForResource	Returns a list of tags for the resource identified by a specified ARN. Tags are used to organize and categorize your CodeDeploy resources.	List	application (p. 845)		
			deploymentgroup (p. 845)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
PutLifecycleEventExecutionStatus	Notify a lifecycle event hook for associated deployment with the IAM user.	Write			
RegisterApplication	Register information about an application revision for an application associated with the IAM user.	Write	application* (p. 845)		
RegisterOnPremisesInstance	Register an on-premises instance	Write	instance* (p. 845)		
RemoveTagsFromOnPremisesInstances	Remove tags from one or more on-premises instances.	Tagging	instance* (p. 845)		
SkipWaitTimeForOverridedAnySpecified	In a blue/green deployment, overrides any specified wait time and starts terminating instances immediately after the traffic routing is complete.	Write			
StopDeployment	Description for StopDeployment	Write			
TagResource	Associates the list of tags in the input Tags parameter with the resource identified by the ResourceArn input parameter.	Tagging	application (p. 845)		
			deploymentgroup (p. 845)		
			aws:RequestTag / \${TagKey} (p. 845)	aws:TagKeys (p. 845)	
UntagResource	Disassociates a resource from a list of tags. The resource is identified by the ResourceArn input parameter. The tags are identified by the list of keys in the TagKeys input parameter.	Tagging	application (p. 845)		
			deploymentgroup (p. 845)		
				aws:TagKeys (p. 845)	
UpdateApplication	Description for UpdateApplication	Write	application* (p. 845)		
UpdateDeploymentGroup	Change information about a single deployment group for an application associated with the IAM user.	Write	deploymentgroup* (p. 845)		

Resource Types Defined by AWS CodeDeploy

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 840\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
application	arn:\${Partition}:codedeploy:\${Region}: \${Account}:application:\${ApplicationName}	
deploymentconfig	arn:\${Partition}:codedeploy: \${Region}: \${Account}:deploymentconfig: \${DeploymentConfigurationName}	
deploymentgroup	arn:\${Partition}:codedeploy: \${Region}: \${Account}:deploymentgroup: \${ApplicationName}/\${DeploymentGroupName}	
instance	arn:\${Partition}:codedeploy:\${Region}: \${Account}:instance:\${InstanceName}	

Condition Keys for AWS CodeDeploy

AWS CodeDeploy defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Filters actions based on the presence of tag key-value pairs in the request	String
aws:ResourceTag/\${TagKey}	Filters actions based on tag key-value pairs attached to the resource	String
aws:TagKeys	Filters actions based on the presence of tag keys in the request	String

Actions, Resources, and Condition Keys for Amazon CodeGuru Profiler

Amazon CodeGuru Profiler (service prefix: `codeguru-profiler`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).

- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon CodeGuru Profiler \(p. 846\)](#)
- [Resource Types Defined by Amazon CodeGuru Profiler \(p. 847\)](#)
- [Condition Keys for Amazon CodeGuru Profiler \(p. 847\)](#)

Actions Defined by Amazon CodeGuru Profiler

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ConfigureAgent [permission only]	Grants permission for an agent to register with the orchestration service and retrieve profiling configuration information	Write	ProfilingGroup* (p. 847)		
CreateProfilingGroup	Grants permission to create a profiling group	Write	ProfilingGroup* (p. 847)		
DeleteProfilingGroup	Grants permission to delete a profiling group	Write	ProfilingGroup* (p. 847)		
DescribeProfilingGroup	Grants permission to describe a profiling group	Read	ProfilingGroup* (p. 847)		
GetFindingsReport	Grants permission to get a recommendations report	Read	ProfilingGroup* (p. 847)		
GetFindingsReportSummary	Grants permission to get a summary of recent recommendations for each profiling group in the account	Read			
GetProfile	Grants permission to get aggregated profiles for a specific profiling group	Read	ProfilingGroup* (p. 847)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListFindingsReport	Grants permissions to list the available recommendations reports for a specific profiling group	List	ProfilingGroup* (p. 847)		
ListProfileTimes	Grants permissions to list the start times of the available aggregated profiles for a specific profiling group	List	ProfilingGroup* (p. 847)		
ListProfilingGroups	Grants permissions to list profiling groups in the account	List			
PostAgentProfile [permission only]	Grants permissions to submit a profile collected by an agent belonging to a specific profiling group for aggregation	Write	ProfilingGroup* (p. 847)		
UpdateProfilingGroup	Grants permissions to update a specific profiling group	Write	ProfilingGroup* (p. 847)		

Resource Types Defined by Amazon CodeGuru Profiler

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 846\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
ProfilingGroup	arn:\${Partition}:codeguru-profiler: \${Region}:\${Account}:profilingGroup/ \${profilingGroupName}	

Condition Keys for Amazon CodeGuru Profiler

CodeGuru Profiler has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Amazon CodeGuru Reviewer

Amazon CodeGuru Reviewer (service prefix: `codeguru-reviewer`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).

- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon CodeGuru Reviewer \(p. 848\)](#)
- [Resource Types Defined by Amazon CodeGuru Reviewer \(p. 849\)](#)
- [Condition Keys for Amazon CodeGuru Reviewer \(p. 849\)](#)

Actions Defined by Amazon CodeGuru Reviewer

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AssociateRepository	Grants permission to associates a repository with Amazon CodeGuru Reviewer.	Write	repository (p. 849)		codecommit>ListRepositories codecommit:TagResource events:PutRule events:PutTargets iam>CreateServiceLinkedRole
CreateConnection	Grants permission to perform Web-based oauth handshake for 3rd party providers.	Read			
DescribeRepositoryAssociation	Grants permission to describe a repository association.	Read			
DisassociateRepository	Grants permission to disassociate a repository with Amazon CodeGuru Reviewer.	Write	association* (p. 849)		codecommit:UntagResource events>DeleteRule events:RemoveTargets
GetMetricsData	Grants permission to view pull request metrics in console.	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
[permission only]					
ListRepositoryAssociations	Grants permission to list summary of repository associations.	List			
ListThirdPartyRepositories	Grants permission to list 3rd party providers repositories in console. [permission only]	Read			

Resource Types Defined by Amazon CodeGuru Reviewer

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 848\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
association	<code>arn:\${Partition}:codeguru-reviewer::\${Account}:association:\${ResourceId}</code>	
repository	<code>arn:\${Partition}:codecommit:\${Region}:\${Account}: \${RepositoryName}</code>	aws:ResourceTag/\${TagKey} (p. 849)

Condition Keys for Amazon CodeGuru Reviewer

Amazon CodeGuru Reviewer defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:ResourceTag/\${TagKey}	Filters actions based on tag key-value pairs attached to the resource	String

Actions, Resources, and Condition Keys for AWS CodePipeline

AWS CodePipeline (service prefix: `codepipeline`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS CodePipeline \(p. 850\)](#)
- [Resource Types Defined by AWS CodePipeline \(p. 854\)](#)
- [Condition Keys for AWS CodePipeline \(p. 854\)](#)

Actions Defined by AWS CodePipeline

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AcknowledgeJob	Returns information about a specified job and whether that job has been received by the job worker.	Write			
AcknowledgeThirdPartyJob	Confirms a job worker has received the specified job. Only used for partner actions.	Write			
CreateCustomActionType	Create a custom action you can use to run the pipelines associated with your AWS account.	Tagging	actiontype* (p. 854)		
			aws:RequestTag/\${TagKey} (p. 854)	aws:TagKeys (p. 854)	
CreatePipeline	Create a uniquely named pipeline.	Tagging	pipeline* (p. 854)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:RequestTag/ \${TagKey} (p. 854) aws:TagKeys (p. 854)	
DeleteCustomActionType	Delete a custom action.	Write	actiontype* (p. 854)		
DeletePipeline	Delete a specified pipeline.	Write	pipeline* (p. 854)		
DeleteWebhook	Delete a specified webhook.	Write	webhook* (p. 854)		
DeregisterWebhook	Remove the registration of a webhook with the third party specified in its configuration.	Write	webhook* (p. 854)		
DisableStageTransition	Prevent revisions from transitioning to the next stage in a pipeline.	Write	stage* (p. 854)		
EnableStageTransition	Enable revisions to transition to the next stage in a pipeline.	Write	stage* (p. 854)		
GetJobDetails	Returns information about a job	Read			
GetPipeline	Retrieve information about a pipeline structure.	Read	pipeline* (p. 854)		
GetPipelineExecution	Returns information about the execution of a pipeline, including details about artifacts, the pipeline execution ID, and the name, version, and status of the pipeline	Read	pipeline* (p. 854)		
GetPipelineState	Retrieve information about the current state of the stages and actions of a pipeline.	Read	pipeline* (p. 854)		
GetThirdPartyJobDetails	Requests the details of a job for a third party action. Only used for partner actions.	Read			
ListActionExecutions	Returns a list of the most recent action executions that have happened in a pipeline.	Read	pipeline* (p. 854)		
ListActionTypes	Retrieve a summary of all the action types available for pipelines in your account.	Read	actiontype* (p. 854)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListPipelineExecutions	Gets a summary of the most recent executions for a pipeline.	List	pipeline* (p. 854)		
ListPipelines	Get a summary of all the pipelines associated with your AWS account.	List	pipeline* (p. 854)		
ListTagsForResource	Lists tags for a CodePipeline resource.	Read	actiontype (p. 854)		
			pipeline (p. 854)		
			webhook (p. 854)		
ListWebhooks	Get all the webhooks associated with your AWS account.	List	webhook* (p. 854)		
PollForJobs	Returns information about any jobs for AWS CodePipeline to act upon.	Write	actiontype* (p. 854)		
PollForThirdPartyJobs	Determines whether there are any third party jobs for a job worker to act on. Only used for partner actions.	Write			
PutActionRevision	Edit actions within a pipeline.	Write	action* (p. 854)		
PutApprovalResult	Provides the response to a manual approval request to AWS CodePipeline. Valid responses include Approved and Rejected.	Write	action* (p. 854)		
PutJobFailureResult	Represents the failure of a job as returned to the pipeline by a job worker. Only used for custom actions.	Write			
PutJobSuccessResult	Represents the success of a job as returned to the pipeline by a job worker. Only used for custom actions.	Write			
PutThirdPartyJobFailureResult	Represents the failure of a third party job as returned to the pipeline by a job worker. Only used for partner actions.	Write			
PutThirdPartyJobSuccessResult	Represents the success of a third party job as returned to the pipeline by a job worker. Only used for partner actions.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
PutWebhook	Create or update a webhook	Tagging	pipeline* (p. 854)		
			webhook* (p. 854)		
				aws:RequestTag/ \${TagKey} (p. 854) aws:TagKeys (p. 854)	
RegisterWebhook	Register a webhook with the third party API specified in its configuration.	Write	webhook* (p. 854)		
RetryStageExecution	Resumes the pipeline execution by retrying the last failed actions in a stage	Write	stage* (p. 854)		
StartPipelineExecution	Run the most recent revision through the pipeline.	Write	pipeline* (p. 854)		
TagResource	Tags a CodePipeline resource.	Tagging	actiontype (p. 854)		
			pipeline (p. 854)		
			webhook (p. 854)		
				aws:RequestTag/ \${TagKey} (p. 854) aws:TagKeys (p. 854)	
UntagResource	Removes a tag from a CodePipeline resource.	Tagging	actiontype (p. 854)		
			pipeline (p. 854)		
			webhook (p. 854)		
				aws:TagKeys (p. 854)	
UpdatePipeline	Update a pipeline with changes to the structure of the pipeline.	Write	pipeline* (p. 854)		

Resource Types Defined by AWS CodePipeline

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 850\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
action	arn:\${Partition}:codepipeline:\${Region}: \${Account}:#\${PipelineName}/\${StageName}/ \${ActionName}	aws:ResourceTag/\${TagKey} (p. 854)
actiontype	arn:\${Partition}:codepipeline:\${Region}: \${Account}:actiontype:\${Owner}/\${Category}/ \${Provider}/\${Version}	aws:ResourceTag/\${TagKey} (p. 854)
pipeline	arn:\${Partition}:codepipeline:\${Region}: \${Account}:#\${PipelineName}	aws:ResourceTag/\${TagKey} (p. 854)
stage	arn:\${Partition}:codepipeline:\${Region}: \${Account}:#\${PipelineName}/\${StageName}	aws:ResourceTag/\${TagKey} (p. 854)
webhook	arn:\${Partition}:codepipeline:\${Region}: \${Account}:webhook:\${WebhookName}	aws:ResourceTag/\${TagKey} (p. 854)

Condition Keys for AWS CodePipeline

AWS CodePipeline defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Filters actions based on the presence of tag key-value pairs in the request	String
aws:ResourceTag/\${TagKey}	Filters actions based on tag key-value pairs attached to the resource	String
aws:TagKeys	Filters actions based on the presence of tag keys in the request	String

Actions, Resources, and Condition Keys for AWS CodeStar

AWS CodeStar (service prefix: codestar) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS CodeStar \(p. 855\)](#)
- [Resource Types Defined by AWS CodeStar \(p. 857\)](#)
- [Condition Keys for AWS CodeStar \(p. 857\)](#)

Actions Defined by AWS CodeStar

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AssociateTeamMember	Adds a user to the team for an AWS CodeStar project.	Permissions management	project* management (p. 857)		
CreateProject	Creates a project with minimal structure, customer policies, and no resources.	Permissions management		aws:RequestTag/\${TagKey} (p. 858) aws:TagKeys (p. 858)	
CreateUserProfile	Creates a profile for a user that includes user preferences, display name, and email.	Write	user* (p. 857)		
DeleteExtendedAccess [permission only]	Grants access to extended delete	Write	project* (p. 857)		
DeleteProject	Deletes a project, including project resources. Does not delete users associated with the project, but does delete the IAM	Permissions management	project* management (p. 857)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
	roles that allowed access to the project.				
DeleteUserProfile	Deletes a user profile in AWS CodeStar, including all personal preference data associated with that profile, such as display name and email address. It does not delete the history of that user, for example the history of commits made by that user.	Write	user* (p. 857)		
DescribeProject	Describes a project and its resources.	Read	project* (p. 857)		
DescribeUserProfile	Describes a user in AWS CodeStar and the user attributes across all projects.	Read			
DisassociateTeam	Removes a user from a project. Removing a user from a project also removes the IAM policies from that user that allowed access to the project and its resources.	Permissions management	project* (p. 857)		
GetExtendedAccessAPIs [permission only]	Grants access to extended read APIs.	Read	project* (p. 857)		
ListProjects	Lists all projects in CodeStar associated with your AWS account.	List			
ListResources	Lists all resources associated with a project in CodeStar.	List	project* (p. 857)		
ListTagsForProject	Lists the tags associated with a project in CodeStar.	List	project* (p. 857)		
ListTeamMembers	Lists all team members associated with a project.	List	project* (p. 857)		
ListUserProfiles	Lists user profiles in AWS CodeStar.	List			
PutExtendedAccessAPIs [permission only]	Grants access to extended write APIs.	Write	project* (p. 857)		
TagProject	Adds tags to a project in CodeStar.	Tagging	project* (p. 857)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:RequestTag/ \${TagKey} (p. 858) aws:TagKeys (p. 858)	
UntagProject	Removes tags from a project in CodeStar.	Tagging	project* (p. 857)		
				aws:TagKeys (p. 858)	
UpdateProject	Updates a project in CodeStar.	Write	project* (p. 857)		
UpdateTeamMember	Updates team member attributes within a CodeStar project.	Permissions management	project* (p. 857)		
UpdateUserProfile	Updates a profile for a user that includes user preferences, display name, and email.	Write	user* (p. 857)		

Resource Types Defined by AWS CodeStar

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 855\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
project	arn:\${Partition}:codestar:\${Region}: \${Account}:project/\${ProjectId}	aws:ResourceTag/ \${TagKey} (p. 858)
user	arn:\${Partition}:iam::\${Account}:user/ \${aws:username}	iam:ResourceTag/ \${TagKey} (p. 858)

Condition Keys for AWS CodeStar

AWS CodeStar defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Filters create requests based on the allowed set of values for each of the tags.	String
aws:ResourceTag/\${TagKey}	Filters actions based on tag-value associated with the resource.	String
aws:TagKeys	Filters create requests based on the presence of mandatory tags in the request.	String
iam:ResourceTag/\${TagKey}		String

Actions, Resources, and Condition Keys for AWS CodeStar Notifications

AWS CodeStar Notifications (service prefix: codestar-notifications) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS CodeStar Notifications \(p. 858\)](#)
- [Resource Types Defined by AWS CodeStar Notifications \(p. 862\)](#)
- [Condition Keys for AWS CodeStar Notifications \(p. 862\)](#)

Actions Defined by AWS CodeStar Notifications

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateNotificationRule	Grants permission to create a notification rule for a resource	Write	notificationrule* (p. 862)		
	aws:RequestTag/ \${TagKey} (p. 862)		aws:TagKeys (p. 862)	codestar-notifications:NotificationsForResource (p. 863)	
DeleteNotificationRule	Grants permission to delete a notification rule for a resource	Write	notificationrule* (p. 862)		
	aws:ResourceTag/ \${TagKey} (p. 862)		aws:RequestTag/ \${TagKey} (p. 862)	aws:TagKeys (p. 862)	
DeleteTarget	Grants permission to delete a target for a notification rule	Write		aws:RequestTag/ \${TagKey} (p. 862)	aws:TagKeys (p. 862)
DescribeNotificationRule	Grants permission to get information about a notification rule	Read	notificationrule* (p. 862)		
	aws:ResourceTag/ \${TagKey} (p. 862)		aws:RequestTag/ \${TagKey} (p. 862)	aws:TagKeys (p. 862)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				codestar-notifications:NotificationsForResource (p. 863)	
ListEventTypes	Grants permission to list notifications event types	List			
ListNotificationRules	Grants permission to list notification rules in an AWS account	List			
ListTagsForResource	Grants permission to list the tags attached to a notification rule resource ARN	List	notificationrule* (p. 862)		
				aws:RequestTag/\${TagKey} (p. 862)	
				aws:TagKeys (p. 862)	
ListTargets	Grants permission to list the notification rule targets for an AWS account	List		aws:RequestTag/\${TagKey} (p. 862)	
				aws:TagKeys (p. 862)	
Subscribe	Grants permission to create an association between a notification rule and an Amazon SNS topic	Write	notificationrule* (p. 862)		
				aws:ResourceTag/\${TagKey} (p. 862)	
				aws:RequestTag/\${TagKey} (p. 862)	
				aws:TagKeys (p. 862)	
				codestar-notifications:NotificationsForResource (p. 863)	
TagResource	Grants permission to attach resource tags to a notification rule resource ARN	Tagging	notificationrule* (p. 862)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:ResourceTag/ \${TagKey} (p. 862) aws:RequestTag/ \${TagKey} (p. 862) aws:TagKeys (p. 862)	
Unsubscribe	Grants permission to remove an association between a notification rule and an Amazon SNS topic	Write	notificationrule* (p. 862)	aws:ResourceTag/ \${TagKey} (p. 862) aws:RequestTag/ \${TagKey} (p. 862) aws:TagKeys (p. 862) codestar-notifications:NotificationsForResource (p. 863)	
UntagResource	Grants permission to disassociate resource tags from a notification rule resource ARN	Tagging	notificationrule* (p. 862)	aws:RequestTag/ \${TagKey} (p. 862) aws:TagKeys (p. 862)	
UpdateNotificationRule	Grants permission to change a notification rule for a resource	Write	notificationrule* (p. 862)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:ResourceTag/\${TagKey} (p. 862) aws:RequestTag/\${TagKey} (p. 862) aws:TagKeys (p. 862) codestar-notifications:NotificationsForResource (p. 863)	

Resource Types Defined by AWS CodeStar Notifications

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 858\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
notificationrule	arn:\${Partition}:codestar-notifications:\${Region}:\${Account}:notificationrule/\${NotificationRuleId}	aws:ResourceTag/\${TagKey} (p. 862)

Condition Keys for AWS CodeStar Notifications

AWS CodeStar Notifications defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Filters actions based on the presence of tag key-value pairs in the request	String
aws:ResourceTag/\${TagKey}	Filters actions based on tag key-value pairs attached to the resource	String
aws:TagKeys	Filters actions based on the presence of tag keys in the request	String

Condition Keys	Description	Type
codestar-notifications:NotificationType	Filters access based on the ARN of the resource for which notifications are configured	ARN

Actions, Resources, and Condition Keys for Amazon Cognito Identity

Amazon Cognito Identity (service prefix: `cognito-identity`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Cognito Identity \(p. 863\)](#)
- [Resource Types Defined by Amazon Cognito Identity \(p. 866\)](#)
- [Condition Keys for Amazon Cognito Identity \(p. 866\)](#)

Actions Defined by Amazon Cognito Identity

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateIdentityPool	Creates a new identity pool.	Write		aws:RequestTag/\${TagKey} (p. 866) aws:TagKeys (p. 866)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:ResourceTag/\${TagKey} (p. 866)	
DeleteIdentities	Deletes identities from an identity pool. You can specify a list of 1-60 identities that you want to delete.	Write			
DeleteIdentityPool	Deletes a user pool. Once a pool is deleted, users will not be able to authenticate with the pool.	Write	identitypool* (p. 866)		
DescribeIdentity	Returns metadata related to the given identity, including when the identity was created and any associated linked logins.	Read			
DescribeIdentityPool	Gets details about a particular identity pool, including the pool name, ID description, creation date, and current number of users.	Read	identitypool* (p. 866)		
GetCredentialsForIdentity	Returns credentials for the provided identity ID.	Read			
GetId	Generates (or retrieves) a Cognito ID. Supplying multiple logins will create an implicit linked account.	Write			
GetIdentityPoolRoles	Gets the roles for an identity pool.	Read	identitypool* (p. 866)		
GetOpenIdToken	Gets an OpenID token, using a known Cognito ID.	Read			
GetOpenIdTokenForDeveloperIdentity	Registers (or retrieves) a Cognito Identity ID and an OpenID Connect token for a user authenticated by your backend authentication process.	Read	identitypool* (p. 866)		
ListIdentities	Lists the identities in a pool.	List	identitypool* (p. 866)		
ListIdentityPools	Lists all of the Cognito identity pools registered for your account.	List			
ListTagsForResource	Lists the tags that are assigned to an Amazon Cognito identity pool.	List	identitypool (p. 866)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:ResourceTag/ \${TagKey} (p. 866)	
LookupDeveloperIdentities	Retrieves the IdentityID associated with a DeveloperUserIdentity or the list of DeveloperUserIdentifiers associated with an IdentityId for an existing identity.	Read	identitypool* (p. 866)		
MergeDeveloperIdentities	Merges two users having different IdentityIds, existing in the same identity pool, and identified by the same developer provider.	Write	identitypool* (p. 866)		
SetIdentityPoolRoles	Sets the roles for an identity pool. These roles are used when making calls to GetCredentialsForIdentity action.	Write			
TagResource	Assigns a set of tags to an Amazon Cognito identity pool.	Tagging	identitypool (p. 866)		
				aws:RequestTag/ \${TagKey} (p. 866)	
	aws:TagKeys (p. 866)				
UnlinkDeveloperIdentity	Unlinks a DeveloperUserIdentity from an existing identity.	Write	identitypool* (p. 866)		
UnlinkIdentity	Unlinks a federated identity from an existing account.	Write			
UntagResource	Removes the specified tags from an Amazon Cognito identity pool.	Tagging	identitypool (p. 866)		
				aws:TagKeys (p. 866)	
				aws:ResourceTag/ \${TagKey} (p. 866)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UpdateIdentityPool	Updates a user pool.	Write	identitypool* (p. 866)		

Resource Types Defined by Amazon Cognito Identity

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 863\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
identitypool	<code>arn:\${Partition}:cognito-identity:\${Region}:\${Account}:identitypool/\${IdentityPoolId}</code>	aws:ResourceTag/\${TagKey} (p. 866)

Condition Keys for Amazon Cognito Identity

Amazon Cognito Identity defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the [IAM Policy Reference](#).

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Filters actions based on the presence of tag key-value pairs in the request.	String
aws:ResourceTag/\${TagKey}	Filters actions based on tag key-value pairs attached to the resource.	String
aws:TagKeys	Filters access by a key that is present in the request.	String

Actions, Resources, and Condition Keys for Amazon Cognito Sync

Amazon Cognito Sync (service prefix: `cognito-sync`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).

- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Cognito Sync \(p. 867\)](#)
- [Resource Types Defined by Amazon Cognito Sync \(p. 868\)](#)
- [Condition Keys for Amazon Cognito Sync \(p. 869\)](#)

Actions Defined by Amazon Cognito Sync

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
BulkPublish	Initiates a bulk publish of all existing datasets for an Identity Pool to the configured stream.	Write	identitypool* (p. 869)		
DeleteDataset	Deletes the specific dataset.	Write	dataset* (p. 869)		
DescribeDataset	Gets meta data about a dataset by identity and dataset name.	Read	dataset* (p. 869)		
DescribeldentityPdatastorage	Gets usage details (for example, data storage) about a particular identity pool.	Read	identitypool* (p. 869)		
DescribeldentityUident	Gets usage information for an Identity, including number of datasets and data usage.	Read	identity* (p. 869)		
GetBulkPublishDeBulka	Get the status of the last BulkPublish operation for an identity pool.	Read	identitypool* (p. 869)		
GetCognitoEvents	Gets the events and the corresponding Lambda functions associated with an identity pool.	Read	identitypool* (p. 869)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetIdentityPoolConfig	Gets the configuration settings of an identity pool.	Read	identitypool* (p. 869)		
ListDatasets	Lists datasets for an identity.	List	dataset* (p. 869)		
ListIdentityPoolUsers	Gets a list of identity pools registered with Cognito.	Read	identitypool* (p. 869)		
ListRecords	Gets paginated records, optionally changed after a particular sync count for a dataset and identity.	Read	dataset* (p. 869)		
QueryRecords [permission only]	A permission that grants the ability to query records.	Read			
RegisterDevice	Registers a device to receive push sync notifications.	Write	identity* (p. 869)		
SetCognitoEvents	Sets the AWS Lambda function for a given event type for an identity pool.	Write	identitypool* (p. 869)		
SetDatasetConfiguration [permission only]	A permission that grants ability to configure datasets.	Write	dataset* (p. 869)		
SetIdentityPoolConfig	Sets the necessary configuration for push sync.	Write	identitypool* (p. 869)		
SubscribeToDatasetNotifications	Subscribes to receive notifications when a dataset is modified by another device.	Write	dataset* (p. 869)		
UnsubscribeFromDatasetNotifications	Unsubscribes from receiving notifications when a dataset is modified by another device.	Write	dataset* (p. 869)		
UpdateRecords	Posts updates to records and adds and deletes records for a dataset and user.	Write	dataset* (p. 869)		

Resource Types Defined by Amazon Cognito Sync

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 867\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
dataset	arn:\${Partition}:cognito-sync:\${Region}: \${Account}:identitypool/\${IdentityPoolId}/ identity/\${IdentityId}/dataset/ \${DatasetName}	
identity	arn:\${Partition}:cognito-sync:\${Region}: \${Account}:identitypool/\${IdentityPoolId}/ identity/\${IdentityId}	
identitypool	arn:\${Partition}:cognito-sync:\${Region}: \${Account}:identitypool/\${IdentityPoolId}	

Condition Keys for Amazon Cognito Sync

Cognito Sync has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Amazon Cognito User Pools

Amazon Cognito User Pools (service prefix: `cognito-idp`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Cognito User Pools \(p. 869\)](#)
- [Resource Types Defined by Amazon Cognito User Pools \(p. 876\)](#)
- [Condition Keys for Amazon Cognito User Pools \(p. 877\)](#)

Actions Defined by Amazon Cognito User Pools

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AddCustomAttributes	Adds additional user attributes to the user pool schema.	Write	userpool* (p. 876)		
AdminAddUserToGroup	Adds the specified user to the specified group.	Write	userpool* (p. 876)		
AdminConfirmSignUp	Confirms user registration as an admin without using a confirmation code. Works on any user.	Write	userpool* (p. 876)		
AdminCreateUser	Creates a new user in the specified user pool and sends a welcome message via email or phone (SMS).	Write	userpool* (p. 876)		
AdminDeleteUser	Deletes a user as an administrator. Works on any user.	Write	userpool* (p. 876)		
AdminDeleteUserAttributes	Deletes the user attributes in a user pool as an administrator. Works on any user.	Write	userpool* (p. 876)		
AdminDisableProviderLink	Disables the user from signing in with the specified external (SAML or social) identity provider.	Write	userpool* (p. 876)		
AdminDisableUser	Disables the specified user as an administrator. Works on any user.	Write	userpool* (p. 876)		
AdminEnableUser	Enables the specified user as an administrator. Works on any user.	Write	userpool* (p. 876)		
AdminForgetDevice	Forgets the device, as an administrator.	Write	userpool* (p. 876)		
AdminGetDevice	Gets the device, as an administrator.	Read	userpool* (p. 876)		
Admin GetUser	Gets the specified user by user name in a user pool as an administrator. Works on any user.	Read	userpool* (p. 876)		
AdminInitiateAuth	Authenticates a user in a user pool as an administrator. Works on any user.	Write	userpool* (p. 876)		
AdminLinkProviderInFolder	Links an existing user account in a user pool (DestinationUser)	Write	userpool* (p. 876)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
	to an identity from an external identity provider (SourceUser) based on a specified attribute name and value from the external identity provider.				
AdminListDevices	Lists devices, as an administrator.	List	userpool* (p. 876)		
AdminListGroups	Lists the groups that the user belongs to.	List	userpool* (p. 876)		
AdminListUserAuthEvents	Lists the authentication events for the user.	Read	userpool* (p. 876)		
AdminRemoveUserFromGroup	Removes the specified user from the specified group.	Write	userpool* (p. 876)		
AdminResetUserPassword	Resets the specified user's password in a user pool as an administrator. Works on any user.	Write	userpool* (p. 876)		
AdminRespondToAuthChallenge	Responds to an authentication challenge as an administrator.	Write	userpool* (p. 876)		
AdminSetUserMFASupport	Sets MFA preference for the user in the userpool	Write	userpool* (p. 876)		
AdminSetUserPassword	Sets the specified user's password in a user pool as an administrator. Works on any user.	Write	userpool* (p. 876)		
AdminSetUserSettings	Sets all the user settings for a specified user name. Works on any user.	Write	userpool* (p. 876)		
AdminUpdateAuthenticationFeedback	Updates the feedback for the user authentication event	Write	userpool* (p. 876)		
AdminUpdateDeviceStatus	Updates the device status as an administrator.	Write	userpool* (p. 876)		
AdminUpdateUserAttributes	Updates the specified user's attributes, including developer attributes, as an administrator.	Write	userpool* (p. 876)		
AdminUserGlobalSignOut	Signs out users from all devices, as an administrator.	Write	userpool* (p. 876)		
AssociateSoftwareToken	Returns a unique generated token secret key code for the user account.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ChangePassword	Changes the password for a specified user in a user pool.	Write			
ConfirmDevice	Confirms tracking of the device. This API call is the call that begins device tracking.	Write			
ConfirmForgotPassword	Allows a user to enter a confirmation code to reset a forgotten password.	Write			
ConfirmSignUp	Confirms registration of a user and handles the existing alias from a previous user.	Write			
CreateGroup	Creates a new group in the specified user pool.	Write	userpool* (p. 876)		
CreateIdentityProvider	Creates an identity provider for a user pool.	Write	userpool* (p. 876)		
CreateResourceServer	Creates a new OAuth2.0 resource server and defines custom scopes in it.	Write	userpool* (p. 876)		
CreateUserImportJob	Creates the user import job.	Write	userpool* (p. 876)		
CreateUserPool	Creates a new Amazon Cognito user pool and sets the password policy for the pool.	Write		aws:RequestTag/\${TagKey} (p. 877) aws:TagKeys (p. 877) aws:ResourceTag/\${TagKey} (p. 877)	
CreateUserPoolClient	Creates the user pool client.	Write	userpool* (p. 876)		
CreateUserPoolDomain	Creates a new domain for a user pool.	Write	userpool* (p. 876)		
DeleteGroup	Deletes a group. Currently only groups with no members can be deleted.	Write	userpool* (p. 876)		
DeleteIdentityProvider	Deletes an identity provider for a user pool.	Write	userpool* (p. 876)		
DeleteResourceServer	Deletes a resource server.	Write	userpool* (p. 876)		
DeleteUser	Allows a user to delete one's self.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteUserAttributes	Deletes the attributes for a user.	Write			
DeleteUserPool	Deletes the specified Amazon Cognito user pool.	Write	userpool* (p. 876)		
DeleteUserPoolClient	Allows the developer to delete the user pool client.	Write	userpool* (p. 876)		
DeleteUserPoolDomain	Deletes a domain for a user pool.	Write	userpool* (p. 876)		
DescribeIdentityProvider	Gets information about a specific identity provider.	Read	userpool* (p. 876)		
DescribeResourceServer	Describes a resource server.	Read	userpool* (p. 876)		
DescribeRiskConfiguration	Describes the risk configuration settings for the userpool / userpool client	Read	userpool* (p. 876)		
DescribeUserImportJob	Describes the user import job.	Read	userpool* (p. 876)		
DescribeUserPool	Returns the configuration information and metadata of the specified user pool.	Read	userpool* (p. 876)		
DescribeUserPoolConfig	Client method for returning the configuration information and metadata of the specified user pool client.	Read	userpool* (p. 876)		
DescribeUserPoolDomain	Gets information about a domain.	Read			
ForgetDevice	Forgets the specified device.	Write			
ForgotPassword	Calling this API causes a message to be sent to the end user with a confirmation code that is required to change the user's password.	Write			
GetCSVHeader	Gets the header information for the .csv file to be used as input for the user import job.	Read	userpool* (p. 876)		
GetDevice	Gets the device.	Read			
GetGroup	Gets a group.	Read	userpool* (p. 876)		
GetIdentityProvider	Gets the specified identity provider identifier	Read	userpool* (p. 876)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetSigningCertificate	Returns the signing certificate.	Read	userpool* (p. 876)		
GetUICustomization	Gets the UI Customization information for a particular app client's app UI, if there is something set.	Read	userpool* (p. 876)		
 GetUser	Gets the user attributes and metadata for a user.	Read			
 GetUserAttributeVerificationCode	Gets the user attribute <code>VerificationCode</code> for the specified attribute name.	Read			
 GetUserPoolMfaConfig	Gets the MFA configuration for the userpool	Read	userpool* (p. 876)		
 GlobalSignOut	Signs out users from all devices.	Write			
 InitiateAuth	Initiates the authentication flow.	Write			
 ListDevices	Lists the devices.	List			
 ListGroups	Lists the groups associated with a user pool.	List	userpool* (p. 876)		
 ListIdentityProviders	Lists information about all identity providers for a user pool.	List	userpool* (p. 876)		
 ListResourceServers	Lists the resource servers for a user pool.	List	userpool* (p. 876)		
 ListTagsForResource	Lists the tags that are assigned to an Amazon Cognito user pool.	List	userpool (p. 876)		
 ListUserImportJobs	Lists the user import jobs..	List	userpool* (p. 876)		
 ListUserPoolClients	Lists the clients that have been created for the specified user pool.	List	userpool* (p. 876)		
 ListUserPools	Lists the user pools associated with an AWS account.	List			
 ListUsers	Lists the users in the Amazon Cognito user pool.	List	userpool* (p. 876)		
 ListUsersInGroup	Lists the users in the specified group.	List	userpool* (p. 876)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ResendConfirmationCode	Resends the confirmation (for confirmation of registration) to a specific user in the user pool.	Write			
RespondToAuthChallenge	Responds to the authentication challenge.	Write			
SetRiskConfiguration	sets the risk configuration setting for the userpool / userpool client	Write	userpool* (p. 876)		
SetUICustomization	Sets the UI customization information for a user pool's built-in app UI.	Write	userpool* (p. 876)		
SetUserMFPrefs	Sets MFA preference for the user in the userpool	Write			
SetUserPoolMFACode	Sets the MFA configuration for the userpool	Write	userpool* (p. 876)		
SetUserSettings	Sets the user settings like multi-factor authentication (MFA).	Write			
SignUp	Registers the user in the specified user pool and creates a user name, password, and user attributes.	Write			
StartUserImportJob	Starts the user import.	Write	userpool* (p. 876)		
StopUserImportJob	Stops the user import job.	Write	userpool* (p. 876)		
TagResource	Assigns a set of tags to an Amazon Cognito user pool.	Tagging	userpool (p. 876)		
			aws:RequestTag / \${TagKey} (p. 877)		aws:TagKeys (p. 877)
UntagResource	Removes the specified tags from an Amazon Cognito user pool.	Tagging	userpool (p. 876)		
				aws:TagKeys (p. 877)	
UpdateAuthEventFeedback	Updates the feedback for the user authentication event	Write	userpool* (p. 876)		
UpdateDeviceStatus	Updates the device status.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UpdateGroup	Updates the specified group with the specified attributes.	Write	userpool* (p. 876)		
UpdateIdentityProviderInformation	Updates identity provider information for a user pool.	Write	userpool* (p. 876)		
UpdateResourceServer	Updates the name and scopes of resource server.	Write	userpool* (p. 876)		
UpdateUserAttribute	Allows a user to update a specific attribute (one at a time).	Write			
UpdateUserPool	Updates the specified user pool with the specified attributes.	Write	userpool* (p. 876)		
				aws:RequestTag/\${TagKey} (p. 877)	
				aws:TagKeys (p. 877)	
UpdateUserPoolClient	Allows the developer to update the specified user pool client and password policy.	Write	userpool* (p. 876)		
UpdateUserPoolDomain	Updates the Secure Sockets Layer(SSL) certificate for the custom domain for your user pool.	Write	userpool* (p. 876)		
VerifySoftwareToken	Registers a user's entered TOTP code and mark the user's software token MFA status as verified if successful.	Write			
VerifyUserAttribute	Verifies a user attribute using a one time verification code.	Write			

Resource Types Defined by Amazon Cognito User Pools

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 869\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
userpool	arn:\${Partition}:cognito-idp:\${Region}:\${Account}:userpool/\${UserPoolId}	aws:ResourceTag/\${TagKey} (p. 877)

Condition Keys for Amazon Cognito User Pools

Amazon Cognito User Pools defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
<code>aws:RequestTag/\${TagKey}</code>	Filters actions based on the presence of tag key-value pairs in the request.	String
<code>aws:ResourceTag/\${TagKey}</code>	Filters actions based on tag key-value pairs attached to the resource.	String
<code>aws:TagKeys</code>	Filters access by a key that is present in the request.	String

Actions, Resources, and Condition Keys for Amazon Comprehend

Amazon Comprehend (service prefix: `comprehend`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Comprehend \(p. 877\)](#)
- [Resource Types Defined by Amazon Comprehend \(p. 882\)](#)
- [Condition Keys for Amazon Comprehend \(p. 883\)](#)

Actions Defined by Amazon Comprehend

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
BatchDetectDominantLanguages	Detects the language or languages present in the list of text documents.	Read			
BatchDetectEntities	Detects the named entities ("People", "Places", "Locations", etc) within the given list of text documents.	Read			
BatchDetectKeyPhrases	Detects the phrases in the list of text documents that are most indicative of the content.	Read			
BatchDetectSentiment	Detects the sentiment of a text in the list of documents (Positive, Negative, Neutral, or Mixed).	Read			
BatchDetectSyntax	Detects syntactic information (like Part of Speech, Tokens) in a list of text documents.	Read			
ClassifyDocument	Creates a new document classification request to analyze a single document in real-time, using a previously created and trained custom model and an endpoint.	Read	document-classifier-endpoint* (p. 883)		
CreateDocumentClassifier	Creates a new document classifier that you can use to categorize documents.	Write		aws:RequestTag/ {\$TagKey} (p. 883) aws:TagKeys (p. 883)	
CreateEndpoint	Creates a model-specific endpoint for synchronous inference for a previously trained custom model	Write	document-classifier (p. 883)		
				aws:RequestTag/ {\$TagKey} (p. 883) aws:TagKeys (p. 883)	
CreateEntityRecognizer	Creates an entity recognizer using submitted files.	Write		aws:RequestTag/ {\$TagKey} (p. 883) aws:TagKeys (p. 883)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteDocumentClassifier	Deletes a previously created document classifier.	Write	document-classifier* (p. 883)		
DeleteEndpoint	Deletes a model-specific endpoint for a previously-trained custom model. All endpoints must be deleted in order for the model to be deleted.	Write	document-classifier-endpoint* (p. 883)		
DeleteEntityRecognizer	Deletes a submitted entity recognizer.	Write	entity-recognizer* (p. 883)		
DescribeDocumentClassificationJob	Gets the properties associated with a document classification job.	Read			
DescribeDocumentClassifier	Gets the properties associated with a document classifier.	Read	document-classifier* (p. 883)		
DescribeDominantLanguageDetectionJob	Gets the properties associated with a dominant language detection job.	Read			
DescribeEndpoint	Gets the properties associated with a specific endpoint. Use this operation to get the status of an endpoint.	Read	document-classifier-endpoint* (p. 883)		
DescribeEntitiesDetectionJob	Gets the properties associated with an entities detection job.	Read			
DescribeEntityRecognizer	Provides details about an entity recognizer including status, S3 buckets containing training data, recognizer metadata, metrics, and so on.	Read	entity-recognizer* (p. 883)		
DescribeKeyPhraseDetectionJob	Gets the properties associated with a key phrases detection job.	Read			
DescribeSentimentDetectionJob	Gets the properties associated with a sentiment detection job.	Read			
DescribeTopicsDetectionJob	Gets the properties associated with a topic detection job.	Read			
DetectDominantLanguage	Detects the language or languages present in the text.	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DetectEntities	Detects the named entities ("People", "Places", "Locations", etc) within the given text document.	Read			
DetectKeyPhrases	Detects the phrases in the text that are most indicative of the content.	Read			
DetectSentiment	Detects the sentiment of a text in a document (Positive, Negative, Neutral, or Mixed).	Read			
DetectSyntax	Detects syntactic information (like Part of Speech, Tokens) in a text document.	Read			
ListDocumentClassificationJobs	Gets a list of the document classification jobs that you have submitted.	List			
ListDocumentClassifiers	Gets a list of the document classifiers that you have created.	List			
ListDominantLanguageDetectionJobs	Gets a list of the dominant language detection jobs that you have submitted.	List			
ListEndpoints	Gets a list of all existing endpoints that you've created.	List			
ListEntitiesDetectionJobs	Gets a list of the entity detection jobs that you have submitted.	List			
ListEntityRecognizerAliases	Gets a list of the properties of all entity recognizers that you created, including recognizers currently in training.	List			
ListKeyPhraseDetectionJobs	Get a list of key phrase detection jobs that you have submitted.	List			
ListSentimentDetectionJobs	Gets a list of sentiment detection jobs that you have submitted.	List			
ListTagsForResource	Lists tags for a resource	List	document-classifier (p. 883) document-classifier-endpoint (p. 883)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			entity-recognizer (p. 883)		
ListTopicsDetectionJobs	Gets a list of the topic detection jobs that you have submitted.	List			
StartDocumentClassificationJob	Starts an asynchronous document classification job.	Write	document-classifier* (p. 883)		
StartDominantLanguageDetectionJob	Starts an asynchronous dominant language detection job for a collection of documents.	Write			
StartEntitiesDetectionJob	Starts an asynchronous entity detection job for a collection of documents.	Write			
StartKeyPhrasesDetectionJob	Starts an asynchronous key phrase detection job for a collection of documents.	Write			
StartSentimentDetectionJob	Starts an asynchronous sentiment detection job for a collection of documents.	Write			
StartTopicsDetectionJob	Starts an asynchronous job to detect the most common topics in the collection of documents and the phrases associated with each topic.	Write			
StopDominantLanguageDetectionJob	Stops a dominant language detection job.	Write			
StopEntitiesDetectionJob	Stops an entity detection job.	Write			
StopKeyPhrasesDetectionJob	Stops a key phrase detection job.	Write			
StopSentimentDetectionJob	Stops a sentiment detection job.	Write			
StopTrainingDocumentClassifier	Stop a previously created document classifier training job.	Write	document-classifier* (p. 883)		
StopTrainingEntityRecognizer	Stop a previously created entity recognizer training job.	Write	entity-recognizer* (p. 883)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
TagResource	Tags a resource with given key value pairs	Tagging	document-classifier (p. 883)		
			document-classifier-endpoint (p. 883)		
			entity-recognizer (p. 883)		
				aws:RequestTag/\${TagKey} (p. 883)	aws:TagKeys (p. 883)
UntagResource	Untags a resource with given key	Tagging	document-classifier (p. 883)		
			document-classifier-endpoint (p. 883)		
			entity-recognizer (p. 883)		
				aws:TagKeys (p. 883)	
UpdateEndpoint	Updates information about the specified endpoint.	Write	document-classifier-endpoint* (p. 883)		

Resource Types Defined by Amazon Comprehend

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 877\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
document-classifier	arn:\${Partition}:comprehend:\${Region}: \${Account}:document-classifier/ \${DocumentClassifierName}	aws:ResourceTag/\${TagKey} (p. 883)
entity-recognizer	arn:\${Partition}:comprehend:\${Region}: \${Account}:entity-recognizer/ \${EntityRecognizerName}	aws:ResourceTag/\${TagKey} (p. 883)
document-classifier-endpoint	arn:\${Partition}:comprehend:\${Region}: \${Account}:document-classifier-endpoint/ \${DocumentClassifierEndpointName}	aws:ResourceTag/\${TagKey} (p. 883)

Condition Keys for Amazon Comprehend

Amazon Comprehend defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the [IAM Policy Reference](#).

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Filters create requests based on the allowed set of values for each of the mandatory tags.	String
aws:ResourceTag/\${TagKey}	Filters actions based on the tag value associated with the resource.	String
aws:TagKeys	Filters create requests based on the presence of mandatory tags in the request.	String

Actions, Resources, and Condition Keys for Comprehend Medical

Comprehend Medical (service prefix: `comprehendmedical`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Comprehend Medical \(p. 884\)](#)
- [Resource Types Defined by Comprehend Medical \(p. 884\)](#)
- [Condition Keys for Comprehend Medical \(p. 884\)](#)

Actions Defined by Comprehend Medical

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DetectEntities	Inspects the specified text for the specified type of entities and returns information about them.	Read			
DetectPHI	Inspects the specified text for PHI entities and returns information about them.	Read			

Resource Types Defined by Comprehend Medical

Comprehend Medical does not support specifying a resource ARN in the `Resource` element of an IAM policy statement. To allow access to Comprehend Medical, specify "Resource": "*" in your policy.

Condition Keys for Comprehend Medical

ComprehendMedical has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Compute Optimizer

Compute Optimizer (service prefix: `compute-optimizer`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Compute Optimizer \(p. 885\)](#)
- [Resource Types Defined by Compute Optimizer \(p. 885\)](#)

- [Condition Keys for Compute Optimizer \(p. 885\)](#)

Actions Defined by Compute Optimizer

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetAutoScalingGroupRecommendations	Grants permission to get recommendations for the provided autoscaling groups.	List			
GetEC2InstanceRecommendations	Grants permission to get recommendations for the provided EC2 instances.	List			
GetEC2RecommendationProjectedMetrics	Grants permission to get the recommendation projected metrics of the specified instance.	List			
GetEnrollmentStatus	Grants permission to get the enrollment status for the specified account.	List			
GetRecommendationSummaries	Grants permission to get the recommendation summaries for the specified account(s).	List			
UpdateEnrollmentStatus	Grants permission to update the enrollment status.	Write			

Resource Types Defined by Compute Optimizer

Compute Optimizer does not support specifying a resource ARN in the `Resource` element of an IAM policy statement. To allow access to Compute Optimizer, specify “`Resource`”: “*” in your policy.

Condition Keys for Compute Optimizer

Compute Optimizer has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for AWS Config

AWS Config (service prefix: `config`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Config \(p. 886\)](#)
- [Resource Types Defined by AWS Config \(p. 893\)](#)
- [Condition Keys for AWS Config \(p. 894\)](#)

Actions Defined by AWS Config

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
BatchGetAggregators	Returns the current configuration items for resources that are present in your AWS Config aggregator	Read	ConfigurationAggregator* (p. 893)		
BatchGetResources	Returns the current configuration for one or more requested resources	Read			
DeleteAggregationAuthorization	Deletes the authorization granted to the specified configuration aggregator account in a specified region	Write	AggregationAuthorization* (p. 893)		
DeleteConfigRule	Deletes the specified AWS Config rule and all of its evaluation results	Write	ConfigRule* (p. 893)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteConfigurationAggregator	Deletes the specified configuration aggregator and the aggregated data associated with the aggregator	Write	ConfigurationAggregator* (p. 893)		
DeleteConfigurationRecorder	Deletes the configuration recorder	Write			
DeleteConformancePack	Deletes the specified conformance pack and all the AWS Config rules and all evaluation results within that conformance pack.	Write			
DeleteDeliveryChannel	Deletes the delivery channel	Write			
DeleteEvaluationResults	Deletes the evaluation results for the specified Config rule	Write	ConfigRule* (p. 893)		
DeleteOrganizationConfigRule	Deletes the specified organization config rule and all of its evaluation results from all member accounts in that organization.	Write			
DeleteOrganizationConformancePack	Deletes the specified organization conformance pack and all of its evaluation results from all member accounts in that organization.	Write			
DeletePendingAuthorizationRequestsForRegion	Deletes pending authorization requests for a specified aggregator account in a specified region	Write			
DeleteRemediationConfiguration	Deletes the remediation configuration	Write	RemediationConfiguration* (p. 894)		
DeleteRemediationExceptions	Deletes one or more remediation exceptions for specific resource keys for a specific AWS Config Rule.	Write			
DeleteRetentionConfiguration	Deletes the retention configuration	Write			
DeliverConfigSnapshot	Schedules delivery of a configuration snapshot to the Amazon S3 bucket in the specified delivery channel	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeAggregates	Returns a list of compliant and noncompliant rules with the number of resources for compliant and noncompliant rules	List	ConfigurationAggregator* (p. 893)		
DescribeAggregations	Returns a list of authorizations granted to various aggregator accounts and regions	List			
DescribeCompliance	Indicates whether the specified AWS Config Rules are compliant	List	ConfigRule* (p. 893)		
DescribeCompliance	Indicates whether the specified AWS Resources are compliant	List			
DescribeConfigRules	Returns status information for each of your AWS-managed Config rules	List	ConfigRule* (p. 893)		
DescribeConfigRules	Returns details about your AWS Config rules	List	ConfigRule* (p. 893)		
DescribeConfigurations	Returns status information for sources within a aggregation	List	ConfigurationAggregator* (p. 893)		
DescribeConfigurations	Returns the details of one or more Configuration aggregators	List			
DescribeConfigurations	Returns the current status of the specified configuration recorder	List			
DescribeConfigurations	Returns the name of one or more specified configuration recorders	List			
DescribeConformancePacks	Returns compliance information for each Conformance pack.	Read			
DescribeConformancePacks	Provides one or more Conformance packs deployment status.	Read			
DescribeConformancePacks	Returns a list of one or more Conformance packs.	Read			
DescribeDeliveryChannels	Returns the current status of the specified delivery channel	List			
DescribeDeliveryChannels	Returns details about the specified delivery channel	List			
DescribeOrganizationDeploymentStatus	Provides organization config rule deployment status for an organization.	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeOrganizationConfigRules	Returns a list of organization configuration rules.	Read			
DescribeOrganizationConformancePackDeployment	Provides organization conformance pack deployment status for an organization.	Read			
DescribeOrganizationConformancePacks	Returns a list of organization conformance packs.	Read			
DescribePendingAggregationRequests	Returns a list of all pending aggregation requests.	List			
DescribeRemediationConfigurations	Returns the details of one or more remediations configurations.	List	RemediationConfiguration* (p. 894)		
DescribeRemediationExceptions	Returns the details of one or more remediation exceptions.	List			
DescribeRemediationExecutionLogs	Provides a detailed view of remediation execution for a set of resources including state, timestamps and any error messages for steps that have failed.	List	RemediationConfiguration* (p. 894)		
DescribeRetentionConfigurations	Returns the details of one or more retention configurations.	List			
GetAggregateComplianceForConfigRule	Returns the evaluation results for the specified AWS Config rule for a specific resource in a rule.	Read	ConfigurationAggregator* (p. 893)		
GetAggregateComplianceForRegions	Returns the number of compliant and non-compliant rules for one or more accounts and regions in an aggregator.	Read	ConfigurationAggregator* (p. 893)		
GetAggregateDiscrepancyCounts	Returns the resource counts across accounts and regions that are present in your AWS Config aggregator.	Read	ConfigurationAggregator* (p. 893)		
GetAggregateResourceConfigurations	Returns configuration item that is aggregated for your specific resource in a specific source account and region.	Read	ConfigurationAggregator* (p. 893)		
GetComplianceDetailsForConfigRule	Returns the evaluation results for the specified AWS Config rule.	Read	ConfigRule* (p. 893)		
GetComplianceDetailsForResource	Returns the evaluation results for the specified AWS resource.	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetComplianceSummaryForConfigRules	Returns the number of AWS Config rules that are compliant and noncompliant, up to a maximum of 25 for each.	Read			
GetComplianceSummaryForConfigRuleType	Returns the number of resources that are compliant and the number that are noncompliant.	Read			
GetConformancePacks	Returns compliance details of a conformance pack for all AWS resources that are monitored by conformance pack.	Read			
GetConformancePacksForCompliance	Provides compliance summary for one or more conformance packs.	Read			
GetDiscoveredResources	Returns the resource types, the number of each resource type, and the total number of resources that AWS Config is recording in this region for your AWS account.	Read			
GetOrganizationConfigRuleStatus	Returns detailed status for each member account within an organization for a given organization config rule.	Read			
GetOrganizationConfigRuleStatuses	Returns detailed status for each member account within an organization for a given organization conformance pack.	Read			
GetResourceConfigItems	Returns a list of configuration items for the specified resource.	Read			
ListAggregateDiscoveredResources	Accepts a resource type and returns a list of resource identifiers that are aggregated for a specific resource type across accounts and regions.	List	ConfigurationAggregator* (p. 893)		
ListDiscoveredResources	Accepts a resource type and returns a list of resource identifiers for the resources of that type.	List			
ListTagsForResource	List the tags for AWS Config resource	List	AggregationAuthorization (p. 893)	ConfigRule (p. 893)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			ConfigurationAggregator (p. 893)		
PutAggregationAccount	Authorizes the aggregator account and region to collect data from the source account and region	Write	AggregationAuthorization* (p. 893)		
				aws:RequestTag/\${TagKey} (p. 894)	
PutConfigRule	Adds or updates an AWS Config rule for evaluating whether your AWS resources comply with your desired configurations	Write	ConfigRule* (p. 893)		
				aws:RequestTag/\${TagKey} (p. 894)	
				aws:TagKeys (p. 894)	
PutConfigurationRecorder	Creates and updates the Configuration aggregator with the selected source accounts and regions	Write	ConfigurationAggregator* (p. 893)		
				aws:RequestTag/\${TagKey} (p. 894)	
PutConformancePack	Creates or updates a Conformance pack.	Write			
PutDeliveryChannel	Creates a delivery channel object to deliver configuration information to an Amazon S3 bucket and Amazon SNS topic	Write			
PutEvaluations	Used by an AWS Lambda function to deliver evaluation results to AWS Config	Write			
PutOrganizationConfigRule	Adds or updates organization ConfigRule for your entire organization evaluating whether your AWS resources comply with your desired configurations.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
PutOrganizationConformancePack	Adds or updates organization Conformance Pack for your entire organization evaluating whether your AWS resources comply with your desired configurations.	Write			
PutRemediationConfiguration	Adds or updates the remediation configuration with a specific AWS Config rule with the selected target or action	Write	RemediationConfiguration* (p. 894)		
PutRemediationException	Adds or updates remediation exception for specific resources for a specific AWS Config rule.	Write			
PutRetentionConfiguration	Creates and updates the retention configuration with details about retention period (number of days) that AWS Config stores your historical information	Write			
SelectResourceConfigurations	Accepts a structured query language (SQL) SELECT command, performs the corresponding search, and returns resource configurations matching the properties	Read			
StartConfigRulesEvaluation	Evaluates your resources against the specified Config rules	Write	ConfigRule* (p. 893)		
StartConfigurationRecording	Starts recording configurations of other AWS resources you have selected to record in your AWS account	Write			
StartRemediation	Runs an on-demand remediation for the specified AWS Config rules against the last known remediation configuration	Write	RemediationConfiguration* (p. 894)		
StopConfigurationRecording	Stops recording configurations of other AWS resources you have selected to record in your AWS account	Write			
TagResource	Associates the specified tags to a resource with the specified resourceArn	Tagging	AggregationAuthorization (p. 893)		
			ConfigRule (p. 893)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			ConfigurationAggregator (p. 893)		
			ConformancePack (p. 893)		
				aws:RequestTag/ \${TagKey} (p. 894)	
				aws:TagKeys (p. 894)	
UntagResource	Deletes specified tags from a resource	Tagging	AggregationAuthorization (p. 893)		
			ConfigRule (p. 893)		
			ConfigurationAggregator (p. 893)		
			ConformancePack (p. 893)		
				aws:TagKeys (p. 894)	

Resource Types Defined by AWS Config

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 886\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
AggregationAuthorization	arn:\${Partition}:config:\${Region}: \${Account}:aggregation-authorization/ \${AggregatorAccount}/\${AggregatorRegion}	aws:ResourceTag/ \${TagKey} (p. 894)
ConfigurationAggregator	arn:\${Partition}:config:\${Region}: \${Account}:config-aggregator/\${AggregatorId}	aws:ResourceTag/ \${TagKey} (p. 894)
ConfigRule	arn:\${Partition}:config:\${Region}: \${Account}:config-rule/\${ConfigRuleId}	aws:ResourceTag/ \${TagKey} (p. 894)
ConformancePack	arn:\${Partition}:config:\${Region}: \${Account}:conformance-pack/ \${ConformancePackName}/\${ConformancePackId}	aws:ResourceTag/ \${TagKey} (p. 894)

Resource Types	ARN	Condition Keys
OrganizationConfigRule	arn:\${Partition}:config:\${Region}: \${Account}:organization-config-rule/ \${OrganizationConfigRuleId}	
OrganizationConformancePack	arn:\${Partition}:config:\${Region}: \${Account}:organization-conformance-pack/ \${OrganizationConformancePackId}	
RemediationConfiguration	arn:\${Partition}:config:\${Region}: \${Account}:remediation-configuration/ \${RemediationConfigurationId}	

Condition Keys for AWS Config

AWS Config defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/ \${TagKey}	Filters actions based on the allowed set of values for each of the tags	String
aws:ResourceTag/ \${TagKey}	Filters actions based on tag-value associated with the resource	String
aws:TagKeys	Filters actions based on the presence of mandatory tags in the request	String

Actions, Resources, and Condition Keys for Amazon Connect

Amazon Connect (service prefix: connect) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Connect \(p. 895\)](#)
- [Resource Types Defined by Amazon Connect \(p. 900\)](#)
- [Condition Keys for Amazon Connect \(p. 901\)](#)

Actions Defined by Amazon Connect

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateInstance	Grants permissions to create a new Amazon Connect instance. The associated required actions grant permissions to configure instance settings.	Write			ds>CreateAlias ds>DeleteDirectory ds>DescribeDirectories firehose>DescribeDelivery firehose>ListDeliveryStreams iam>CreateServiceLinkedRole kinesis>DescribeStream kinesis>ListStreams kms>CreateGrant kms>DescribeKey kms>ListAliases kms>RetireGrant s3>CreateBucket s3>ListAllMyBuckets
CreateUser	Grants permission to create a user for the specified Amazon Connect instance.	Write	routing-profile* (p. 900)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			hierarchy-group (p. 900)		
			aws:RequestTag/ \${TagKey} (p. 901)	aws:TagKeys (p. 901)	
DeleteUser	Grants permissions to delete a user in an Amazon Connect instance.	Write	user* (p. 900)		
				aws:ResourceTag/ \${TagKey} (p. 901)	
DescribeInstance	Grants permissions to view details of an Amazon Connect instance. This is required to create an instance.	Read	instance* (p. 900)		firehose:DescribeDelivery firehose>ListDeliveryStrea kinesis:DescribeStream kinesis>ListStreams kms:DescribeKey kms>ListAliases s3>ListAllMyBuckets
DescribeUser	Grants permissions to describe a user in an Amazon Connect instance.	Read	user* (p. 900)		
				aws:ResourceTag/ \${TagKey} (p. 901)	
DescribeUserHierarchyGroup	Grants permissions to describe a hierarchy group for an Amazon Connect instance.	Read	hierarchy-group* (p. 900)		
DescribeUserHierarchyStructure	Grants permissions to describe the hierarchy structure for an Amazon Connect instance.	Read	instance* (p. 900)		
DestroyInstance	Grants permissions to delete an Amazon Connect instance. When you remove an instance, the link to an existing AWS directory is also removed.	Write	instance* (p. 900)		
GetContactAttribute	Grants permissions to retrieve the contact attributes for the specified contact.	Read	contact* (p. 900)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetCurrentMetricData	Grants permissions to retrieve current metric data for the queues in an Amazon Connect instance.	Read	queue* (p. 900)		
GetFederationToken	Allows federation into an instance when using SAML-based authentication for identity management.	Read	instance* (p. 900)		
GetFederationTokens	Grants permissions to federate to an Amazon Connect instance (Log in as administrator functionality in the AWS console).	Write	instance* (p. 900)		connect:DescribeInstances connect>ListInstances ds:DescribeDirectories
GetMetricData	Grants permissions to retrieve historical metric data for queues in an Amazon Connect instance.	Read	queue* (p. 900)		
ListContactFlows	Grants permissions to list contact flow resources in an Amazon Connect instance.	List	instance* (p. 900)		
ListHoursOfOperation	Grants permissions to list hours of operation resources in an Amazon Connect instance.	List	instance* (p. 900)		
ListInstances	Grants permissions to view the Amazon Connect instances associated with an AWS account.	List			
ListPhoneNumbers	Grants permissions to list phone number resources in an Amazon Connect instance.	List	instance* (p. 900)		
ListQueues	Grants permissions to list queue resources in an Amazon Connect instance.	List	instance* (p. 900)		
ListRoutingProfiles	Grants permissions to list routing profile resources in an Amazon Connect instance.	List	instance* (p. 900)		
ListSecurityProfiles	Grants permissions to list security profile resources in an Amazon Connect instance.	List	instance* (p. 900)		
ListTagsForResource	Grants permissions to list tags for an Amazon Connect resource.	Read	user (p. 900)		
					aws:ResourceTag/\${TagKey} (p. 901)

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListUserHierarchy	Grants permissions to list the Hierarchy group resources in an Amazon Connect instance.	List	instance* (p. 900)		
ListUsers	Grants permissions to list user resources in an Amazon Connect instance.	List	instance* (p. 900)		
ModifyInstance	Grants permissions to modify configuration settings for an existing Amazon Connect instance. The associated required actions grant permission modify the settings for the instance.	Write	instance* (p. 900)		firehose:DescribeDelivery firehose>ListDeliveryStream kinesis:DescribeStream kinesis>ListStreams kms>CreateGrant kms:DescribeKey kms>ListAliases kms:RetireGrant s3>CreateBucket s3>ListAllMyBuckets
StartOutboundVoice	Grants permissions to initiate outbound calls using the Amazon Connect API.	Write	contact* (p. 900)		
StopContact	Grants permissions to stop contacts that were initiated using the Amazon Connect API. If you use this operation on an active contact the contact ends, even if an agent is active on a call with a customer.	Write	contact* (p. 900)		
TagResource	Grants permissions to tag an Amazon Connect resource.	Tagging	user (p. 900)		
			aws:TagKeys (p. 901) aws:RequestTag/ {\$TagKey} (p. 901) aws:ResourceTag/ {\$TagKey} (p. 901)		
UntagResource	Grants permissions to untag an Amazon Connect resource.	Tagging	user (p. 900)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:TagKeys (p. 901) aws:ResourceTag/ {\$TagKey} (p. 901)	
UpdateContactAttribute	Grants permissions to create or update the contact attributes associated with the specified contact.	Write	contact* (p. 900)		
UpdateUserHierarchyGroup	Grants permissions to update a hierarchy group for a user in an Amazon Connect instance.	Write	user* (p. 900)		
			hierarchy-group (p. 900)		
				aws:ResourceTag/ {\$TagKey} (p. 901)	
UpdateUserIdentity	Grants permissions to update identity information for a user in an Amazon Connect instance.	Write	user* (p. 900)		
				aws:ResourceTag/ {\$TagKey} (p. 901)	
UpdateUserPhoneConfig	Grants permissions to update phone configuration settings for a user in an Amazon Connect instance.	Write	user* (p. 900)		
				aws:ResourceTag/ {\$TagKey} (p. 901)	
UpdateUserRoutingProfile	Grants permissions to update a routing profile for a user in an Amazon Connect instance.	Write	routing-profile* (p. 900)		
			user* (p. 900)		
				aws:ResourceTag/ {\$TagKey} (p. 901)	
UpdateUserSecurityProfile	Grants permissions to update security profiles for a user in an Amazon Connect instance.	Write	security-profile* (p. 900)		
			user* (p. 900)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:ResourceTag/ \${TagKey} (p. 901)	

Resource Types Defined by Amazon Connect

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 895\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
instance	arn:\${Partition}:connect:\${Region}: \${Account}:instance/\${InstanceId}	
contact	arn:\${Partition}:connect:\${Region}: \${Account}:instance/\${InstanceId}/contact/ \${ContactId}	
user	arn:\${Partition}:connect:\${Region}: \${Account}:instance/\${InstanceId}/agent/ \${UserId}	aws:ResourceTag/ \${TagKey} (p. 901)
routing-profile	arn:\${Partition}:connect:\${Region}: \${Account}:instance/\${InstanceId}/routing-profile/\${RoutingProfileId}	
security-profile	arn:\${Partition}:connect:\${Region}: \${Account}:instance/\${InstanceId}/security-profile/\${SecurityProfileId}	
hierarchy-group	arn:\${Partition}:connect:\${Region}: \${Account}:instance/\${InstanceId}/agent-group/\${HierarchyGroupId}	
queue	arn:\${Partition}:connect:\${Region}: \${Account}:instance/\${InstanceId}/queue/ \${QueueId}	
contact-flow	arn:\${Partition}:connect:\${Region}: \${Account}:instance/\${InstanceId}/contact-flow/\${ContactFlowId}	
hours-of-operation	arn:\${Partition}:connect:\${Region}: \${Account}:instance/\${InstanceId}/operating-hours/\${HoursOfOperationId}	
phone-number	arn:\${Partition}:connect:\${Region}: \${Account}:instance/\${InstanceId}/phone-numbers/\${PhoneNumberId}	

Condition Keys for Amazon Connect

Amazon Connect defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
<code>aws:RequestTag/\${TagKey}</code>	Filters actions based on the presence of tag key-value pairs in the request.	String
<code>aws:ResourceTag/\${TagKey}</code>	Filters actions based on tag key-value pairs attached to the resource.	String
<code>aws:TagKeys</code>	Filters actions based on the presence of tag keys in the request.	String

Actions, Resources, and Condition Keys for AWS Cost and Usage Report

AWS Cost and Usage Report (service prefix: `cur`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Cost and Usage Report \(p. 901\)](#)
- [Resource Types Defined by AWS Cost and Usage Report \(p. 902\)](#)
- [Condition Keys for AWS Cost and Usage Report \(p. 902\)](#)

Actions Defined by AWS Cost and Usage Report

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteReportDefinition	Delete Cost and Usage Report Definition	Write	cur* (p. 902)		
DescribeReportDefinitions	Get Cost and Usage Report Definitions	Read			
ModifyReportDefinition	Modify Cost and Usage Report Definition	Write	cur* (p. 902)		
PutReportDefinition	Write Cost and Usage Report Definition	Write	cur* (p. 902)		

Resource Types Defined by AWS Cost and Usage Report

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 901\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
cur	arn:\${Partition}:cur:\${Region}: \${Account}:definition/\${ReportName}	

Condition Keys for AWS Cost and Usage Report

Cost and Usage Report has no service-specific context keys that can be used in the Condition element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for AWS Cost Explorer Service

AWS Cost Explorer Service (service prefix: ce) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Cost Explorer Service \(p. 903\)](#)

- [Resource Types Defined by AWS Cost Explorer Service \(p. 904\)](#)
- [Condition Keys for AWS Cost Explorer Service \(p. 904\)](#)

Actions Defined by AWS Cost Explorer Service

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateCostCategory	Grants permission to create a new <code>CostCategory</code> with the requested name and rules.	Write			
DeleteCostCategory	Grants permission to delete a <code>CostCategory</code> .	Write			
DescribeCostCategories	Grants permission to retrieve <code>CostCategory</code> descriptions such as the name, ARN, rules, definition, and effective dates of a Cost Category.	Read			
GetCostAndUsage	Grants permission to retrieve the cost and usage metrics for your account.	Read			
GetCostAndUsageWithReservations	Grants permission to retrieve the cost and usage metrics with resources for your account.	Read			
GetCostForecast	Grants permission to retrieve a cost forecast for a forecast time period.	Read			
GetDimensionValues	Grants permission to retrieve all available filter values for a filter for a period of time.	Read			
GetReservationCoverage	Grants permission to retrieve the reservation coverage for your account.	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetReservationPurchaseRecommendations	Grants permission to retrieve the reservation recommendations for your account.	Read			
GetReservationUtilization	Grants permission to retrieve the reservation utilization for your account.	Read			
GetRightsizingRecommendations	Grants permission to retrieve the rightsizing recommendations for your account.	Read			
GetSavingsPlansCoverage	Grants permission to retrieve the Savings Plans coverage for your account.	Read			
GetSavingsPlansRecommendations	Grants permission to retrieve the Savings Plans recommendations for your account.	Read			
GetSavingsPlansUtilization	Grants permission to retrieve the Savings Plans utilization for your account.	Read			
GetSavingsPlansUtilizationDetails	Grants permission to retrieve the Savings Plans utilization details for your account.	Read			
GetTags	Grants permission to query tags for a specified time period.	Read			
GetUsageForecast	Grants permission to retrieve a usage forecast for a forecast time period.	Read			
ListCostCategoryDefinitions	Grants permission to retrieve Cost Category ARN, and effective dates for all Cost Categories.	List			
UpdateCostCategory	Grants permission to update an existing Cost Category.	Write			

Resource Types Defined by AWS Cost Explorer Service

AWS Cost Explorer Service does not support specifying a resource ARN in the `Resource` element of an IAM policy statement. To allow access to AWS Cost Explorer Service, specify “`Resource`”: “`*`” in your policy.

Condition Keys for AWS Cost Explorer Service

Cost Explorer Service has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for AWS Data Exchange

AWS Data Exchange (service prefix: `dataexchange`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Data Exchange \(p. 905\)](#)
- [Resource Types Defined by AWS Data Exchange \(p. 907\)](#)
- [Condition Keys for AWS Data Exchange \(p. 908\)](#)

Actions Defined by AWS Data Exchange

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CancelJob	Grants permissions to cancel a job.	Write	jobs* (p. 907)		
CreateDataSet	Grants permission to create a data set.	Write		aws:RequestTag/\${TagKey} (p. 908) aws:ResourceTag/\${TagKey} (p. 908) aws:TagKeys (p. 908)	
CreateJob	Grants permissions to create a job to import or export assets.	Write	jobs* (p. 907)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateRevision	Grants permission to create a revision.	Write		aws:RequestTag/ \${TagKey} (p. 908) aws:ResourceTag/ \${TagKey} (p. 908) aws:TagKeys (p. 908)	
DeleteAsset	Grants permissions to delete an asset.	Write	assets* (p. 907)		
DeleteDataSet	Grants permissions to delete a data set.	Write	data-sets* (p. 907)		
DeleteRevision	Grants permissions to delete a revision.	Write	revisions* (p. 907)		
GetAsset	Grants permissions to get information about an asset.	Read	assets* (p. 907)		
GetDataSet	Grants permission to get information about a data set.	Read	data-sets* (p. 907)		
GetJob	Grants permissions to get information about a job.	Write	jobs* (p. 907)		
GetRevision	Grants permission to get information about a revision.	Read	revisions* (p. 907)		
ListDataSetRevisions	Grants permissions to list the revisions of a data set.	List	revisions* (p. 907)		
ListDataSets	Grants permission to list data sets for the account.	List	data-sets* (p. 907)		
ListJobs	Grants permissions to list jobs for the account.	List	jobs* (p. 907)		
ListRevisionAssets	Grants permissions to get list the assets of a revision.	List	assets* (p. 907)		
ListTagsForResource	Grants permission to list the tags that you associated with the specified resource.	Read	data-sets (p. 907)		
			revisions (p. 907)		
StartJob	Grants permissions to start a job.	Write	jobs* (p. 907)		
TagResource	Grants permission to add one or more tags to a specified resource.	Tagging	data-sets (p. 907)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			revisions (p. 907)		
				aws:RequestTag/ \${TagKey} (p. 908)	
				aws:TagKeys (p. 908)	
UntagResource	Grants permission to remove one or more tags from a specified resource.	Tagging	data-sets (p. 907)		
			revisions (p. 907)		
				aws:TagKeys (p. 908)	
UpdateAsset	Grants permissions to get update information about an asset.	Write	assets* (p. 907)		
UpdateDataSet	Grants permissions to update information about a data set.	Write	data-sets* (p. 907)		
UpdateRevision	Grants permissions to update information about a revision.	Write	revisions* (p. 907)		

Resource Types Defined by AWS Data Exchange

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 905\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
jobs	arn:\${Partition}:dataexchange:\${Region}: \${Account}:jobs/\${JobId}	dataexchange:JobType (p. 908)
data-sets	arn:\${Partition}:dataexchange:\${Region}: \${Account}:data-sets/\${DataSetId}	
revisions	arn:\${Partition}:dataexchange:\${Region}: \${Account}:data-sets/\${DataSetId}/revisions/ \${RevisionId}	
assets	arn:\${Partition}:dataexchange:\${Region}: \${Account}:data-sets/\${DataSetId}/revisions/ \${RevisionId}/assets/\${AssetId}	

Condition Keys for AWS Data Exchange

AWS Data Exchange defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
<code>aws:RequestTag/\${TagKey}</code>	Filters create requests based on the allowed set of values for each of the mandatory tags.	String
<code>aws:ResourceTag/\${TagKey}</code>	Filters actions based on the tag value associated with the resource.	String
<code>aws:TagKeys</code>	Filters create requests based on the presence of mandatory tags in the request.	String
<code>dataexchange:JobType</code>	Indicates that the action can only be performed on the specified job type.	String

Actions, Resources, and Condition Keys for Amazon Data Lifecycle Manager

Amazon Data Lifecycle Manager (service prefix: `dlm`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Data Lifecycle Manager \(p. 908\)](#)
- [Resource Types Defined by Amazon Data Lifecycle Manager \(p. 909\)](#)
- [Condition Keys for Amazon Data Lifecycle Manager \(p. 910\)](#)

Actions Defined by Amazon Data Lifecycle Manager

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you

specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateLifecyclePolicy	Create a data lifecycle policy to manage the scheduled creation and retention of Amazon EBS snapshots. You may have up to 100 policies.	Write		aws:RequestTag/\${TagKey} (p. 910) aws:TagKeys (p. 910)	
DeleteLifecyclePolicy	Delete an existing data lifecycle policy. In addition, this action halts the creation and deletion of snapshots that the policy specified. Existing snapshots are not affected.	Write	policy* (p. 909)		
GetLifecyclePolicies	Returns a list of summary descriptions of data lifecycle policies.	List			
GetLifecyclePolicy	Returns a complete description of a single data lifecycle policy.	Read	policy* (p. 909)		
ListTagsForResource	Grants permission to list the tags associated with a resource.	Read	policy* (p. 909)		
TagResource	Grants permission to add or update tags of a resource.	Tagging	policy* (p. 909)		
UntagResource	Grants permission to remove associated with a resource.	Tagging	policy* (p. 909)		
UpdateLifecyclePolicy	Updates an existing data lifecycle policy.	Write	policy* (p. 909)		

Resource Types Defined by Amazon Data Lifecycle Manager

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 908\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
policy	arn:\${Partition}:dlm:\${Region}: \${Account}:policy/\${ResourceName}	aws:ResourceTag/\${TagKey} (p. 910)

Condition Keys for Amazon Data Lifecycle Manager

Amazon Data Lifecycle Manager defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
<code>aws:RequestTag/\${TagKey}</code>	Filters actions based on the presence of tag key-value pairs in the request	String
<code>aws:ResourceTag/\${TagKey}</code>	Filters actions based on tag key-value pairs attached to the resource	String
<code>aws:TagKeys</code>	Filters actions based on the presence of tag keys in the request	String

Actions, Resources, and Condition Keys for Data Pipeline

Data Pipeline (service prefix: `datapipeline`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Data Pipeline \(p. 910\)](#)
- [Resource Types Defined by Data Pipeline \(p. 913\)](#)
- [Condition Keys for Data Pipeline \(p. 913\)](#)

Actions Defined by Data Pipeline

You can specify the following actions in the Action element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the Resource element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ActivatePipeline	Validates the specified pipeline and starts processing pipeline tasks. If the pipeline does not pass validation, activation fails.	Write		datipeline:PipelineCreator (p. 913) datipeline:Tag (p. 913) datipeline:workerGroup (p. 913)	
AddTags	Adds or modifies tags for the specified pipeline.	Tagging		datipeline:PipelineCreator (p. 913) datipeline:Tag (p. 913)	
CreatePipeline	Creates a new, empty pipeline.	Write		datipline:Tag (p. 913)	
DeactivatePipeline	Deactivates the specified running pipeline.	Write		datipline:PipelineCreator (p. 913) datipline:Tag (p. 913) datipline:workerGroup (p. 913)	
DeletePipeline	Deletes a pipeline, its pipeline definition, and its run history.	Write		datipline:PipelineCreator (p. 913) datipline:Tag (p. 913)	
DescribeObjects	Gets the object definitions for a set of objects associated with the pipeline.	Read		datipline:PipelineCreator (p. 913) datipline:Tag (p. 913)	
DescribePipelines	Retrieves metadata about one or more pipelines.	List		datipline:PipelineCreator (p. 913) datipline:Tag (p. 913)	
EvaluateExpression	Task runners call <code>EvaluateExpression</code> to evaluate a string in the context of the specified object.	Read		datipline:PipelineCreator (p. 913) datipline:Tag (p. 913)	
GetAccountLimits	Description for <code>GetAccountLimits</code>	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetPipelineDefinition	Gets the definition of the specified pipeline.	Read		datipeline:PipelineCreator (p. 913) datipline:Tag (p. 913) datipline:workerGroup (p. 913)	
ListPipelines	Lists the pipeline identifiers for all active pipelines that you have permission to access.	List			
PollForTask	Task runners call PollForTask to receive a task to perform from AWS Data Pipeline.	Write		datipline:workerGroup (p. 913)	
PutAccountLimits	Description for PutAccountLimits	Write			
PutPipelineDefinition	Adds tasks, schedules, and preconditions to the specified pipeline.	Write		datipline:PipelineCreator (p. 913) datipline:Tag (p. 913) datipline:workerGroup (p. 913)	
QueryObjects	Queries the specified pipeline for the names of objects that match the specified set of conditions.	Read		datipline:PipelineCreator (p. 913) datipline:Tag (p. 913)	
RemoveTags	Removes existing tags from the specified pipeline.	Tagging		datipline:PipelineCreator (p. 913) datipline:Tag (p. 913)	
ReportTaskProgress	Task runners call ReportTaskProgress when assigned a task to acknowledge that it has the task.	Write			
ReportTaskRunnerHeartbeat	Task runners call ReportTaskRunnerHeartbeat every 15 minutes to indicate that they are operational.	Write			
SetStatus	Requests that the status of the specified physical or logical pipeline objects be updated in the specified pipeline.	Write		datipline:PipelineCreator (p. 913) datipline:Tag (p. 913)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
SetTaskStatus	Task runners call SetTaskStatus to notify AWS Data Pipeline that a task is completed and provide information about the final status.	Write			
ValidatePipelineDefinition	Validates the specified pipeline definition to ensure that it is well formed and can be run without error.	Read		datapipeline:PipelineCreator (p. 913) datapipeline:Tag (p. 913) datapipeline:workerGroup (p. 913)	

Resource Types Defined by Data Pipeline

Data Pipeline does not support specifying a resource ARN in the `Resource` element of an IAM policy statement. To allow access to Data Pipeline, specify “`Resource`”: “`*`” in your policy.

Condition Keys for Data Pipeline

Data Pipeline defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
datapipeline:PipelineCreator	The IAM user that created the pipeline.	ARN
datapipeline:Tag	A customer-specified key/value pair that can be attached to a resource.	ARN
datapipeline:workerGroup	The name of a worker group for which a Task Runner retrieves work.	ARN

Actions, Resources, and Condition Keys for AWS Database Migration Service

AWS Database Migration Service (service prefix: `dms`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Database Migration Service \(p. 914\)](#)
- [Resource Types Defined by AWS Database Migration Service \(p. 919\)](#)
- [Condition Keys for AWS Database Migration Service \(p. 920\)](#)

Actions Defined by AWS Database Migration Service

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AddTagsToResource	Adds metadata tags to a DMS resource, including replication instance, endpoint, security group, and migration task	Tagging	Certificate (p. 920) Endpoint (p. 920) EventSubscription (p. 920) ReplicationInstance (p. 919) ReplicationSubnetGroup (p. 920) ReplicationTask (p. 920) aws:RequestTag/ \${TagKey} (p. 920) aws:TagKeys (p. 920) dms:req- tag/ \${TagKey} (p. 921)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ApplyPendingMaintenanceActionToAResource	Applies a pending maintenance action to a resource (for example, to a replication instance).	Write	ReplicationInstance* (p. 919)		
CreateEndpoint	Creates an endpoint using the provided settings	Write		aws:RequestTag/ {\$TagKey} (p. 920) aws:TagKeys (p. 920) dms:req- tag/ {\$TagKey} (p. 921)	
CreateEventSubscription	Creates an AWS DMS event notification subscription.	Write		aws:RequestTag/ {\$TagKey} (p. 920) aws:TagKeys (p. 920) dms:req- tag/ {\$TagKey} (p. 921)	
CreateReplicationInstance	Creates the replication instance using the specified parameters	Write		aws:RequestTag/ {\$TagKey} (p. 920) aws:TagKeys (p. 920) dms:req- tag/ {\$TagKey} (p. 921)	
CreateReplicationSubnetGroup	Creates a replication subnet group given a list of the subnet IDs in a VPC	Write		aws:RequestTag/ {\$TagKey} (p. 920) aws:TagKeys (p. 920) dms:req- tag/ {\$TagKey} (p. 921)	
CreateReplicationTask	Creates a replication task using the specified parameters	Write	Endpoint* (p. 920)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			ReplicationInstance* (p. 919)		
				aws:RequestTag/ \${TagKey} (p. 920)	
				aws:TagKeys (p. 920)	
				dms:req- tag/ \${TagKey} (p. 921)	
DeleteCertificate	Deletes the specified certificate	Write	Certificate* (p. 920)		
DeleteEndpoint	Deletes the specified endpoint	Write	Endpoint* (p. 920)		
DeleteEventSubscription	Deletes an AWS DMS event subscription.	Write	EventSubscription* (p. 920)		
DeleteReplicationInstance	Deletes the specified replication instance	Write	ReplicationInstance* (p. 919)		
DeleteReplicationSubnetGroup	Deletes a subnet group	Write	ReplicationSubnetGroup* (p. 920)		
DeleteReplicationTask	Deletes the specified replication task	Write	ReplicationTask* (p. 920)		
DescribeAccountAttributes	Lists all of the AWS DMS attributes for a customer account	Read			
DescribeCertificate	Provides a description of the certificate.	Read			
DescribeConnections	Describes the status of the connections that have been made between the replication instance and an endpoint	Read			
DescribeEndpoints	Returns information about the types of endpoints available	Read			
DescribeEndpoints	Returns information about the endpoints for your account in the current region	Read			
DescribeEventCategories	Lists categories for all event source types, or, if specified, for a specified source type.	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeEventSubscriptionsForCustomer	Lists all the event subscriptions for a customer account.	Read			
DescribeEvents	Lists events for a given source identifier and source type.	Read			
DescribeOrderableReplicationInstances	Returns information about the replication instance types that can be created in the specified region	Read			
DescribeRefreshSchemasStatus	Returns the status of the RefreshSchemas operation	Read	Endpoint* (p. 920)		
DescribeReplicationTaskLogsForTask	Returns information about the task logs for the specified task.	Read	ReplicationInstance* (p. 919)	aws:ResourceTag/\${TagKey} (p. 920) aws:TagKeys (p. 920)	
DescribeReplicationInstances	Returns information about replication instances for your account in the current region	Read			
DescribeReplicationSubnets	Returns information about the replication subnet groups	Read			
DescribeReplicationTasksFromAmazonS3	Returns the task assessment results from Amazon S3. This action always returns the latest results.	Read	ReplicationTask (p. 920)		
DescribeReplicationTasks	Returns information about replication tasks for your account in the current region	Read			
DescribeSchemas	Returns information about the schema for the specified endpoint	Read	Endpoint* (p. 920)		
DescribeTableStatistics	Returns table statistics on the database migration task, including table name, rows inserted, rows updated, and rows deleted	Read	ReplicationTask* (p. 920)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ImportCertificate	Uploads the specified certificate.	Write		aws:RequestTag/ \${TagKey} (p. 920) aws:TagKeys (p. 920)	
ListTagsForResource	Lists all tags for an AWS DMS resource	List	Certificate (p. 920)		
			Endpoint (p. 920)		
			EventSubscription (p. 920)		
			ReplicationInstance (p. 919)		
			ReplicationSubnetGroup (p. 920)		
			ReplicationTask (p. 920)		
ModifyEndpoint	Modifies the specified endpoint	Write	Endpoint* (p. 920)		
			Certificate (p. 920)		
ModifyEventSubscription	Modifies an existing AWS DMS event notification subscription.	Write			
ModifyReplicationInstance	Modifies the replication instance to apply new settings	Write	ReplicationInstance* (p. 919)		
ModifyReplicationSubnetGroup	Modifies the settings for the specified replication subnet group	Write			
ModifyReplicationTask	Modifies the specified replication task.	Write	ReplicationTask* (p. 920)		
RebootReplicationInstance	Reboots a replication instance. Rebooting results in a momentary outage, until the replication instance becomes available again.	Write	ReplicationInstance* (p. 919)		
RefreshSchemas	Populates the schema for the specified endpoint	Write	Endpoint* (p. 920)		
			ReplicationInstance* (p. 919)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ReloadTables	Reloads the target database table with the source data.	Write	ReplicationTask* (p. 920)		
RemoveTagsFromDMSResource	Removes metadata tags from a DMS resource	Tagging	Certificate (p. 920)		
			Endpoint (p. 920)		
			EventSubscription (p. 920)		
			ReplicationInstance (p. 919)		
			ReplicationSubnetGroup (p. 920)		
			ReplicationTask (p. 920)		
				aws:TagKeys (p. 920)	
StartReplicationTask	Starts the replication task	Write	ReplicationTask* (p. 920)		
StartReplicationTaskAssessmentForUnsupportedDataTypes	Starts the replication task assessment for unsupported data types in the source database.	Write	ReplicationTask* (p. 920)		
StopReplicationTask	Stops the replication task	Write	ReplicationTask* (p. 920)		
TestConnection	Tests the connection between the replication instance and the endpoint	Read	Endpoint* (p. 920)		
				ReplicationInstance* (p. 919)	

Resource Types Defined by AWS Database Migration Service

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 914\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
ReplicationInstance	arn:\${Partition}:dms:\${Region}:\${Account}:rep:*	aws:ResourceTag/ {\$TagKey} (p. 920)

Resource Types	ARN	Condition Keys
		dms:rep-tag/\${TagKey} (p. 921)
ReplicationTask	arn:\${Partition}:dms:\${Region}: \${Account}:task:*	aws:ResourceTag/ \${TagKey} (p. 920) dms:task-tag/ \${TagKey} (p. 921)
Endpoint	arn:\${Partition}:dms:\${Region}: \${Account}:endpoint:*	aws:ResourceTag/ \${TagKey} (p. 920) dms:endpoint-tag/ \${TagKey} (p. 921)
Certificate	arn:\${Partition}:dms:\${Region}: \${Account}:cert:*	aws:ResourceTag/ \${TagKey} (p. 920) dms:cert-tag/ \${TagKey} (p. 921)
EventSubscription	arn:\${Partition}:dms:\${Region}: \${Account}:es:*	aws:ResourceTag/ \${TagKey} (p. 920) dms:es-tag/\${TagKey} (p. 921)
ReplicationSubnetGroup	arn:\${Partition}:dms:\${Region}: \${Account}:subgrp:*	aws:ResourceTag/ \${TagKey} (p. 920) dms:subgrp-tag/ \${TagKey} (p. 921)

Condition Keys for AWS Database Migration Service

AWS Database Migration Service defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/ \${TagKey}	Filters actions based on the presence of tag key-value pairs in the request	String
aws:ResourceTag/ \${TagKey}	Filters actions based on tag key-value pairs attached to the resource	String
aws:TagKeys	Filters actions based on the presence of tag keys in the request	String

Condition Keys	Description	Type
dms:cert-tag/\${TagKey}	Filters actions based on the presence of tag keys in the request for Certificate	String
dms:endpoint-tag/\${TagKey}	Filters actions based on the presence of tag keys in the request for Endpoint	String
dms:es-tag/\${TagKey}	Filters actions based on the presence of tag keys in the request for EventSubscription	String
dms:rep-tag/\${TagKey}	Filters actions based on the presence of tag keys in the request for ReplicationInstance	String
dms:req-tag/\${TagKey}	Filters actions based on the presence of tag key-value pairs in the request	String
dms:subgrp-tag/\${TagKey}	Filters actions based on the presence of tag keys in the request for ReplicationSubnetGroup	String
dms:task-tag/\${TagKey}	Filters actions based on the presence of tag keys in the request for ReplicationTask	String

Actions, Resources, and Condition Keys for Database Query Metadata Service

Database Query Metadata Service (service prefix: `dbqms`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Database Query Metadata Service \(p. 921\)](#)
- [Resource Types Defined by Database Query Metadata Service \(p. 922\)](#)
- [Condition Keys for Database Query Metadata Service \(p. 922\)](#)

Actions Defined by Database Query Metadata Service

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type.

Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateFavoriteQuery	Creates a new favorite query	Write			
CreateQueryHistory	Add a query to the history	Write			
DeleteFavoriteQueries	Delete saved queries	Write			
DeleteQueryHistory	Delete a historical query	Write			
DescribeFavoriteQueryMetadata	List saved queries and associated metadata	List			
DescribeQueryHistorySummary	List history of queries that were run	List			
GetQueryString	Retrieve favorite or history query string by id	Read			
UpdateFavoriteQueryDescription	Update saved query and description	Write			
UpdateQueryHistory	Update the query history	Write			

Resource Types Defined by Database Query Metadata Service

Database Query Metadata Service does not support specifying a resource ARN in the `Resource` element of an IAM policy statement. To allow access to Database Query Metadata Service, specify “`Resource`”: “`**`” in your policy.

Condition Keys for Database Query Metadata Service

DBQMS has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Note

`${ConceptsDocRoot}`

Actions, Resources, and Condition Keys for DataSync

DataSync (service prefix: `datasync`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by DataSync \(p. 923\)](#)
- [Resource Types Defined by DataSync \(p. 926\)](#)
- [Condition Keys for DataSync \(p. 926\)](#)

Actions Defined by DataSync

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CancelTaskExecution	Cancels execution of a sync task.	Write	taskexecution* (p. 926)		
CreateAgent	Activates an agent that you have deployed on your host.	Write		aws:RequestTag/\${TagKey} (p. 926) aws:TagKeys (p. 926)	
CreateLocationEfs	Creates an endpoint for an Amazon EFS file system.	Write		aws:RequestTag/\${TagKey} (p. 926) aws:TagKeys (p. 926)	
CreateLocationNfs	Creates an endpoint for a NFS file system.	Write		aws:RequestTag/\${TagKey} (p. 926) aws:TagKeys (p. 926)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateLocationS3	Creates an endpoint for an Amazon S3 bucket.	Write		aws:RequestTag/\${TagKey} (p. 926) aws:TagKeys (p. 926)	
CreateLocationSmb	Creates an endpoint for an SMB file system.	Write		aws:RequestTag/\${TagKey} (p. 926) aws:TagKeys (p. 926)	
CreateTask	Creates a sync task.	Write		aws:RequestTag/\${TagKey} (p. 926) aws:TagKeys (p. 926)	
DeleteAgent	Deletes an agent.	Write	agent* (p. 926)		
DeleteLocation	Deletes the configuration of a location used by AWS DataSync.	Write	location* (p. 926)		
DeleteTask	Deletes a sync task.	Write	task* (p. 926)		
DescribeAgent	Returns metadata such as name, network interfaces, and the status (that is, whether the agent is running or not) about a sync agent.	Read	agent* (p. 926)		
DescribeLocationEfs	Returns metadata, such as the path information about an Amazon EFS sync location.	Read	location* (p. 926)		
DescribeLocationNfs	Returns metadata, such as the path information, about a NFS sync location.	Read	location* (p. 926)		
DescribeLocationS3	Returns metadata, such as Bucket name, about an Amazon S3 bucket sync location.	Read	location* (p. 926)		
DescribeLocationSmb	Returns metadata, such as the path information, about an SMB sync location.	Read	location* (p. 926)		
DescribeTask	Returns metadata about a sync task.	Read	task* (p. 926)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions	
DescribeTaskExecution	Returns detailed metadata about a sync task that is being executed.	Read	taskexecution* (p. 926)			
ListAgents	Returns a list of agents owned by an AWS account in a region specified in the request.	List				
ListLocations	Returns a lists of source and destination sync locations.	List				
ListTagsForResource	This operation lists the tags that have been added to the specified resource.	Read	agent (p. 926)			
			location (p. 926)			
			task (p. 926)			
ListTaskExecutions	Returns a list of executed sync tasks.	List				
ListTasks	Returns a list of all the sync tasks.	List				
StartTaskExecution	Starts a specific invocation of a sync task.	Write	task* (p. 926)			
TagResource	Applies a key-value pair to an AWS resource.	Write	agent (p. 926)			
				location (p. 926)		
				task (p. 926)		
				aws:RequestTag / \${TagKey} (p. 926)		
UntagResource	This operation removes one or more tags from the specified resource.	Tagging	agent (p. 926)			
				location (p. 926)		
				task (p. 926)		
				aws:TagKeys (p. 926)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UpdateAgent	Updates the name of an agent.	Write	agent* (p. 926)		
UpdateTask	Updates the metadata associated with a sync task.	Write	task* (p. 926)		

Resource Types Defined by DataSync

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 923\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
agent	<code>arn:\${Partition}:datasync:\${Region}: \${AccountId}:agent/\${AgentId}</code>	<code>aws:ResourceTag/\${TagKey}</code> (p. 926)
location	<code>arn:\${Partition}:datasync:\${Region}: \${AccountId}:location/\${LocationId}</code>	<code>aws:ResourceTag/\${TagKey}</code> (p. 926)
task	<code>arn:\${Partition}:datasync:\${Region}: \${AccountId}:task/\${TaskId}</code>	<code>aws:ResourceTag/\${TagKey}</code> (p. 926)
taskexecution	<code>arn:\${Partition}:datasync:\${Region}: \${AccountId}:task/\${TaskId}/execution/ \${ExecutionId}</code>	

Condition Keys for DataSync

DataSync defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
<code>aws:RequestTag/\${TagKey}</code>	Filters create requests based on the allowed set of values for each of the tags.	String
<code>aws:ResourceTag/\${TagKey}</code>	Filters actions based on tag-value associated with the resource.	String
<code>aws:TagKeys</code>	Filters create requests based on the presence of mandatory tags in the request.	String

Actions, Resources, and Condition Keys for AWS DeepLens

AWS DeepLens (service prefix: `deeplens`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

Topics

- [Actions Defined by AWS DeepLens \(p. 927\)](#)
- [Resource Types Defined by AWS DeepLens \(p. 929\)](#)
- [Condition Keys for AWS DeepLens \(p. 929\)](#)

Actions Defined by AWS DeepLens

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AssociateServiceRoleToAccount	Associates the user's account <code>RoleToAccount</code> controlling various permissions needed by AWS DeepLens for proper functionality.	Permissions management			
BatchGetDevice	Retrieves a list of AWS DeepLens devices.	Read	device* (p. 929)		
BatchGetModel	Retrieves a list of AWS DeepLens Models.	Read	model* (p. 929)		
BatchGetProject	Retrieves a list of AWS DeepLens Projects.	Read	project* (p. 929)		
CreateDeviceCertificate	Creates a certificate package <code>certificate</code> used to successfully authenticate and Register an AWS DeepLens device.	Write			
CreateModel	Creates a new AWS DeepLens Model.	Write			
CreateProject	Creates a new AWS DeepLens Project.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteModel	Deletes an AWS DeepLens Model.	Write	model* (p. 929)		
DeleteProject	Deletes an AWS DeepLens Project.	Write	project* (p. 929)		
DeployProject	Deploys an AWS DeepLens project to a registered AWS DeepLens device.	Write	device* (p. 929)		
			project* (p. 929)		
DeregisterDevice	Begins a device de-registration workflow for a registered AWS DeepLens device.	Write	device* (p. 929)		
GetAssociatedResources	Retrieves the account level resources associated with the user's account.	Read			
GetDeploymentStatus	Retrieves the deployment status of a particular AWS DeepLens device, along with any associated metadata.	Read			
GetDevice	Retrieves information about an AWS DeepLens device.	Read	device* (p. 929)		
GetModel	Retrieves an AWS DeepLens Model.	Read	model* (p. 929)		
GetProject	Retrieves an AWS DeepLens Project.	Read	project* (p. 929)		
ImportProjectFromTemplate	Creates a new AWS DeepLens project from a sample project template.	Write			
ListDeployments	Retrieves a list of AWS DeepLens Deployment identifiers.	List			
ListDevices	Retrieves a list of AWS DeepLens device identifiers.	List			
ListModels	Retrieves a list of AWS DeepLens Model identifiers.	List			
ListProjects	Retrieves a list of AWS DeepLens Project identifiers.	List			
RegisterDevice	Begins a device registration workflow for an AWS DeepLens device.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
RemoveProject	Removes a deployed AWS DeepLens project from an AWS DeepLens device.	Write	device* (p. 929)		
UpdateProject	Updates an existing AWS DeepLens Project.	Write	project* (p. 929)		

Resource Types Defined by AWS DeepLens

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 927\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
device	arn:\${Partition}:deplens:\${Region}: \${Account}:device/\${DeviceName}	
project	arn:\${Partition}:deplens:\${Region}: \${Account}:project/\${ProjectName}	
model	arn:\${Partition}:deplens:\${Region}: \${Account}:model/\${ModelName}	

Condition Keys for AWS DeepLens

DeepLens has no service-specific context keys that can be used in the Condition element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for AWS DeepRacer

AWS DeepRacer (service prefix: deepracer) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS DeepRacer \(p. 930\)](#)
- [Resource Types Defined by AWS DeepRacer \(p. 932\)](#)
- [Condition Keys for AWS DeepRacer \(p. 933\)](#)

Actions Defined by AWS DeepRacer

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CloneReinforcementLearningModel [permission only]	Grants permission to clone existing DeepRacer models	Write	reinforcement_learning_model* (p. 932)		
	track* (p. 933)				
CreateAccountResource [permission only]	Grants permission to create resources needed by DeepRacer on behalf of the user	Write			
CreateLeaderboard [permission only]	Grants permission to submit DeepRacer models to be evaluated for leaderboards	Write	leaderboard* (p. 933)		
	reinforcement_learning_model* (p. 932)				
CreateReinforcementLearningModel [permission only]	Grants permission to create reinforcement learning models for DeepRacer	Write	track* (p. 933)		
DeleteAccountResource [permission only]	Grants permission to delete resources created by DeepRacer on behalf of the user	Write			
DeleteModel [permission only]	Grants permission to delete DeepRacer models	Write	reinforcement_learning_model* (p. 932)		
GetAccountResource [permission only]	Grants permission to retrieve the resources created by DeepRacer on behalf of the user	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetAlias [permission only]	Grants permission to retrieve the user's alias for submitting DeepRacer models to leaderboards	Read			
GetEvaluation [permission only]	Grants permission to retrieve information about existing DeepRacer models' evaluation jobs	Read	evaluation_job* (p. 933)		
GetLatestUserSubmission [permission only]	Grants permission to retrieve information about how the latest submitted DeepRacer model for a user performed on a leaderboard	Read	leaderboard* (p. 933)		
GetLeaderboard [permission only]	Grants permission to retrieve information about leaderboards	Read	leaderboard* (p. 933)		
GetModel [permission only]	Grants permission to retrieve information about existing DeepRacer models	Read	reinforcement_learning_model* (p. 932)		
GetRankedUserSubmission [permission only]	Grants permission to retrieve information about the performance of a user's DeepRacer model that got placed on a leaderboard	Read	leaderboard* (p. 933)		
GetTrack [permission only]	Grants permission to retrieve information about DeepRacer tracks	Read	track* (p. 933)		
GetTrainingJob [permission only]	Grants permission to retrieve information about existing DeepRacer models' training job	Read	training_job* (p. 933)		
ListEvaluations [permission only]	Grants permission to list DeepRacer models' evaluation jobs	List	reinforcement_learning_model* (p. 932)		
ListLeaderboardSubmissions [permission only]	Grants permission to list all the submissions of DeepRacer models of a user on a leaderboard	List	leaderboard* (p. 933)		
ListLeaderboards [permission only]	Grants permission to list all the available leaderboards	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListModels [permission only]	Grants permission to list all existing DeepRacer models	List			
ListTracks [permission only]	Grants permission to list all DeepRacer tracks	List			
ListTrainingJobs [permission only]	Grants permission to list DeepRacer models' training jobs	List	reinforcement_learning_model* (p. 932)		
SetAlias [permission only]	Grants permission to set the user's alias for submitting DeepRacer models to leaderboards	Write			
StartEvaluation [permission only]	Grants permission to evaluate DeepRacer models in a simulated environment	Write	reinforcement_learning_model* (p. 932)		
StopEvaluation [permission only]	Grants permission to stop DeepRacer model evaluations		track* (p. 933)		
StopTrainingReinforcementLearningModel [permission only]	Grants permission to stop training reinforcement learning models	Write	evaluation_job* (p. 933)		
TestRewardFunction [permission only]	Grants permission to test reward functions for correctness	Write	reinforcement_learning_model* (p. 932)		

Resource Types Defined by AWS DeepRacer

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 930\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
reinforcement_learning_model	arn:\${Partition}:deepracer:\${Region}: \${Account}:model/reinforcement_learning/ \${ResourceId}	

Resource Types	ARN	Condition Keys
training_job	arn:\${Partition}:deepracer:\${Region}: \${Account}:training_job/\${ResourceId}	
evaluation_job	arn:\${Partition}:deepracer:\${Region}: \${Account}: evaluation_job/\${ResourceId}	
leaderboard_evaluation_job	arn:\${Partition}:deepracer:\${Region}: \${Account}:leaderboard_evaluation_job/ \${ResourceId}	
track	arn:\${Partition}:deepracer:\${Region}::track/ \${ResourceId}	
leaderboard	arn:\${Partition}:deepracer: \${Region}::leaderboard/\${ResourceId}	

Condition Keys for AWS DeepRacer

DeepRacer has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Amazon Detective

Amazon Detective (service prefix: `detective`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Detective \(p. 933\)](#)
- [Resource Types Defined by Amazon Detective \(p. 935\)](#)
- [Condition Keys for Amazon Detective \(p. 935\)](#)

Actions Defined by Amazon Detective

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you

specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AcceptInvitation	Grants permission to accept an invitation to become a member of a behavior graph	Write	Graph* (p. 935)		
CreateGraph	Grants permission to create a behavior graph and begin to aggregate security information	Write			
CreateMembers	Grants permission to request the membership of one or more accounts in a behavior graph managed by this account	Write	Graph* (p. 935)		
DeleteGraph	Grants permission to delete a behavior graph and stop aggregating security information	Write	Graph* (p. 935)		
DeleteMembers	Grants permission to remove member accounts from a behavior graph managed by this account	Write	Graph* (p. 935)		
DisassociateMembers	Grants permission to remove the association of this account with a behavior graph	Write	Graph* (p. 935)		
GetFreeTrialEligibility [permission only]	Grants permission to retrieve a behavior graph's eligibility for a free trial period	Read	Graph* (p. 935)		
GetGraphIngestState [permission only]	Grants permission to retrieve the data ingestion state of a behavior graph	Read	Graph* (p. 935)		
GetMembers	Grants permission to retrieve details on specified members of a behavior graph	Read	Graph* (p. 935)		
GetPricingInformation [permission only]	Grants permission to retrieve information about Amazon Detective's pricing	Read			
GetUsageInformation [permission only]	Grants permission to list usage information of a behavior graph	Read	Graph* (p. 935)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListGraphs	Grants permission to list behavior graphs managed by this account	List			
ListInvitations	Grants permission to retrieve details on the behavior graphs to which this account has been invited to join	List			
ListMembers	Grants permission to retrieve details on all members of a behavior graph	List	Graph* (p. 935)		
RejectInvitation	Grants permission to reject an invitation to become a member of a behavior graph	Write	Graph* (p. 935)		
SearchGraph [permission only]	Grants permission to search the data stored in a behavior graph	Read	Graph* (p. 935)		

Resource Types Defined by Amazon Detective

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 933\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
Graph	arn:\${Partition}:detective:\${Region}:\${Account}:graph:\${ResourceId}	

Condition Keys for Amazon Detective

Detective has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the [IAM Policy Reference](#).

Actions, Resources, and Condition Keys for AWS Device Farm

AWS Device Farm (service prefix: `devicefarm`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Device Farm \(p. 936\)](#)
- [Resource Types Defined by AWS Device Farm \(p. 943\)](#)
- [Condition Keys for AWS Device Farm \(p. 944\)](#)

Actions Defined by AWS Device Farm

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateDevicePool	Grants permission to create a device pool within a project	Write	project* (p. 943)		
CreateInstanceProfile	Grants permission to create a device instance profile	Write			
CreateNetworkProfile	Grants permission to create a network profile within a project	Write	project* (p. 943)		
CreateProject	Grants permission to create a project	Tagging		aws:RequestTag/\${TagKey} (p. 944) aws:TagKeys (p. 944)	
CreateRemoteAccessSession	Grants permission to start a remote access session to a device instance	Write	device* (p. 944)		
			project* (p. 943)		
			deviceinstance (p. 943)		
			upload (p. 943)		
CreateUpload	Grants permission to upload a new file or app within a project	Write	project* (p. 943)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateVPCEConfig	Grants permission to create an Amazon Virtual Private Cloud (VPC) endpoint configuration	Write			
DeleteDevicePool	Grants permission to delete a user-generated device pool	Write	devicepool* (p. 943)		
DeleteInstanceProfile	Grants permission to delete a user-generated instance profile	Write	instanceprofile* (p. 944)		
DeleteNetworkProfile	Grants permission to delete a user-generated network profile	Write	networkprofile* (p. 943)		
DeleteProject	Grants permission to delete a project	Write	project* (p. 943)		
DeleteRemoteAccessSession	Grants permission to delete a completed remote access session and its results	Write	session* (p. 943)		
DeleteRun	Grants permission to delete a run	Write	run* (p. 943)		
DeleteUpload	Grants permission to delete a user-uploaded file	Write	upload* (p. 943)		
DeleteVPCEConfig	Grants permission to delete an Amazon Virtual Private Cloud (VPC) endpoint configuration	Write	vpceconfiguration* (p. 944)		
GetAccountSetting	Grants permission to retrieve the number of unmetered iOS and/or unmetered Android devices purchased by the account	Read			
GetDevice	Grants permission to retrieve the information of a unique device type	Read	device* (p. 944)		
GetDeviceInstance	Grants permission to retrieve the information of a device instance	Read	deviceinstance* (p. 943)		
GetDevicePool	Grants permission to retrieve the information of a device pool	Read	devicepool* (p. 943)		
GetDevicePoolCompatibility	Grants permission to retrieve information about the compatibility of a test and/or app with a device pool	Read	devicepool* (p. 943)		
			upload (p. 943)		
GetInstanceProfile	Grants permission to retrieve the information of an instance profile	Read	instanceprofile* (p. 944)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetJob	Grants permission to retrieve the information of a job	Read	job* (p. 943)		
GetNetworkProfile	Grants permission to retrieve the information of a network profile	Read	networkprofile* (p. 943)		
GetOfferingStatus	Grants permission to retrieve the current status and future status of all offerings purchased by an AWS account	Read			
GetProject	Grants permission to retrieve information about a project	Read	project* (p. 943)		
GetRemoteAccessSession	Grants permission to retrieve the link to a currently running remote access session	Read	session* (p. 943)		
GetRun	Grants permission to retrieve the information of a run	Read	run* (p. 943)		
GetSuite	Grants permission to retrieve the information of a testing suite	Read	suite* (p. 943)		
GetTest	Grants permission to retrieve the information of a test case	Read	test* (p. 943)		
GetUpload	Grants permission to retrieve the information of an uploaded file	Read	upload* (p. 943)		
GetVPCEConfiguration	Grants permission to retrieve the information of an Amazon Virtual Private Cloud (VPC) endpoint configuration	Read	vpceconfiguration* (p. 944)		
InstallToRemoteApplication	Grants permission to install an application to a device in a remote access session	Write	session* (p. 943)		
			upload* (p. 943)		
ListArtifacts	Grants permission to list the artifacts in a project	List	job (p. 943)		
			run (p. 943)		
			suite (p. 943)		
			test (p. 943)		
ListDeviceInstances	Grants permission to list the information of device instances	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListDevicePools	Grants permission to list the information of device pools	List	project* (p. 943)		
ListDevices	Grants permission to list the information of unique device types	List			
ListInstanceProfiles	Grants permission to list the information of device instance profiles	List			
ListJobs	Grants permission to list the information of jobs within a run	List	run* (p. 943)		
ListNetworkProfiles	Grants permission to list the information of network profiles within a project	List	project* (p. 943)		
ListOfferingPromotions	Grants permission to list the offerings promotions	List			
ListOfferingTransactions	Grants permission to list all historical purchases, renewals, and system renewal transactions for an AWS account	List			
ListOfferings	Grants permission to list the products or offerings that the user can manage through the API	List			
ListProjects	Grants permission to list the information of projects for an AWS account	List			
ListRemoteAccessSessions	Grants permission to list the information of currently running remote access sessions	List	project* (p. 943)		
ListRuns	Grants permission to list the information of runs within a project	List	project* (p. 943)		
ListSamples	Grants permission to list the information of samples within a project	List	job* (p. 943)		
ListSuites	Grants permission to list the information of testing suites within a job	List	job* (p. 943)		
ListTagsForResource	Grants permission to list the tags of a resource	List	device (p. 944) deviceinstance (p. 943)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			devicepool (p. 943)		
			instanceprofile (p. 944)		
			networkprofile (p. 943)		
			project (p. 943)		
			run (p. 943)		
			session (p. 943)		
			vpceconfiguration (p. 944)		
ListTests	Grants permission to list the information of tests within a testing suite	List	suite* (p. 943)		
ListUniqueProblems	Grants permission to list the information of unique problems within a run	List	run* (p. 943)		
ListUploads	Grants permission to list the information of uploads within a project	List	project* (p. 943)		
ListVPCEConfigurations	Grants permission to list the information of Amazon Virtual Private Cloud (VPC) endpoint configurations	List			
PurchaseOffering	Grants permission to purchase offerings for an AWS account	Write			
RenewOffering	Grants permission to set the quantity of devices to renew for an offering	Write			
ScheduleRun	Grants permission to schedule a run	Write	project* (p. 943)		
			devicepool (p. 943)		
			upload (p. 943)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
	SCENARIO: Device Pool as filter		devicepool* (p. 943) project* (p. 943) upload (p. 943)		
	SCENARIO: Device Selection Configuration as filter		project* (p. 943) upload (p. 943)		
StopJob	Grants permission to terminate a running job	Write	job* (p. 943)		
StopRemoteAccessSession	Grants permission to terminate a running remote access session	Write	session* (p. 943)		
StopRun	Grants permission to terminate a running test run	Write	run* (p. 943)		
TagResource	Grants permission to add tags to a resource	Tagging	device (p. 944)		
deviceinstance (p. 943)					
devicepool (p. 943)					
instanceprofile (p. 944)					
networkprofile (p. 943)					
project (p. 943)					
run (p. 943)					
session (p. 943)					
vpceconfiguration (p. 944)					

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:RequestTag/ \${TagKey} (p. 944)	
UntagResource	Grants permission to remove tags from a resource	Tagging	device (p. 944)		
			deviceinstance (p. 943)		
			devicepool (p. 943)		
			instanceprofile (p. 944)		
			networkprofile (p. 943)		
			project (p. 943)		
			run (p. 943)		
			session (p. 943)		
			vpceconfiguration (p. 944)		
				aws:TagKeys (p. 944)	
UpdateDeviceInstance	Grants permission to modify an existing device instance	Write	deviceinstance*(p. 943)		
			instanceprofile (p. 944)		
UpdateDevicePool	Grants permission to modify an existing device pool	Write	devicepool*(p. 943)		
UpdateInstanceProfile	Grants permission to modify an existing instance profile	Write	instanceprofile*(p. 944)		
UpdateNetworkProfile	Grants permission to modify an existing network profile	Write	networkprofile*(p. 943)		
UpdateProject	Grants permission to modify an existing project	Write	project*(p. 943)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UpdateUpload	Grants permission to modify an existing upload	Write	upload* (p. 943)		
UpdateVPCEConfiguration	Grants permission to modify an existing Amazon Virtual Private Cloud (VPC) endpoint configuration	Write	vpceconfiguration* (p. 944)		

Resource Types Defined by AWS Device Farm

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 936\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
project	arn:\${Partition}:devicefarm:\${Region}: \${Account}:project:\${ResourceId}	aws:ResourceTag/\${TagKey} (p. 944)
run	arn:\${Partition}:devicefarm:\${Region}: \${Account}:run:\${ResourceId}	aws:ResourceTag/\${TagKey} (p. 944)
job	arn:\${Partition}:devicefarm:\${Region}: \${Account}:job:\${ResourceId}	
suite	arn:\${Partition}:devicefarm:\${Region}: \${Account}:suite:\${ResourceId}	
test	arn:\${Partition}:devicefarm:\${Region}: \${Account}:test:\${ResourceId}	
upload	arn:\${Partition}:devicefarm:\${Region}: \${Account}:upload:\${ResourceId}	
artifact	arn:\${Partition}:devicefarm:\${Region}: \${Account}:artifact:\${ResourceId}	
sample	arn:\${Partition}:devicefarm:\${Region}: \${Account}:sample:\${ResourceId}	
networkprofile	arn:\${Partition}:devicefarm:\${Region}: \${Account}:networkprofile:\${ResourceId}	aws:ResourceTag/\${TagKey} (p. 944)
deviceinstance	arn:\${Partition}:devicefarm: \${Region}::deviceinstance:\${ResourceId}	aws:ResourceTag/\${TagKey} (p. 944)
session	arn:\${Partition}:devicefarm:\${Region}: \${Account}:session:\${ResourceId}	aws:ResourceTag/\${TagKey} (p. 944)
devicepool	arn:\${Partition}:devicefarm:\${Region}: \${Account}:devicepool:\${ResourceId}	aws:ResourceTag/\${TagKey} (p. 944)

Resource Types	ARN	Condition Keys
device	arn:\${Partition}:devicefarm:\${Region}::device:\${ResourceId}	aws:ResourceTag/\${TagKey} (p. 944)
instanceprofile	arn:\${Partition}:devicefarm:\${Region}:\${Account}:instanceprofile:\${ResourceId}	aws:ResourceTag/\${TagKey} (p. 944)
vpceconfiguration	arn:\${Partition}:devicefarm:\${Region}:\${Account}:vpceconfiguration:\${ResourceId}	aws:ResourceTag/\${TagKey} (p. 944)

Condition Keys for AWS Device Farm

AWS Device Farm defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Filters actions based on the allowed set of values for each of the tags	String
aws:ResourceTag/\${TagKey}	Filters actions based on tag-value associated with the resource	String
aws:TagKeys	Filters actions based on the presence of mandatory tags in the request	String

Actions, Resources, and Condition Keys for AWS Direct Connect

AWS Direct Connect (service prefix: `directconnect`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Direct Connect \(p. 944\)](#)
- [Resource Types Defined by AWS Direct Connect \(p. 951\)](#)
- [Condition Keys for AWS Direct Connect \(p. 951\)](#)

Actions Defined by AWS Direct Connect

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually

allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the Resource element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AcceptDirectConnectAttachment	Accepts a proposal request to attach a virtual private gateway to a Direct Connect gateway.	Write	dx-gateway* (p. 951)		
AllocateConnectionOnInterconnect	Creates a hosted connection on an Interconnect.	Write	dxcon* (p. 951)		
AllocateHostedConnection	Creates a new hosted connection between a AWS Direct Connect partner's network and a specific AWS Direct Connect location.	Write	dxcon (p. 951) dxlag (p. 951)	aws:RequestTag/\${TagKey} (p. 952) aws:TagKeys (p. 952)	
AllocatePrivateVirtualInterface	Provisions a private virtual interface to be owned by a different customer.	Write	dxcon (p. 951) dxlag (p. 951)	aws:RequestTag/\${TagKey} (p. 952) aws:TagKeys (p. 952)	
AllocatePublicVirtualInterface	Provisions a public virtual interface to be owned by a different customer.	Write	dxcon (p. 951) dxlag (p. 951)	aws:RequestTag/\${TagKey} (p. 952)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:TagKeys (p. 952)	
AllocateTransitVirtualInterface	Provisions a transit virtual interface to be owned by a different customer.	Write	dxcon (p. 951)		
			dxlag (p. 951)		
				aws:RequestTag/\${TagKey} (p. 952)	aws:TagKeys (p. 952)
AssociateConnectionWithLag	Associates a connection with a LAG.	Write	dxcon* (p. 951)		
			dxlag* (p. 951)		
AssociateHostedConnectionWithVirtualInterface	Associates a hosted connection with virtual interfaces with a link aggregation group (LAG) or interconnect.	Write	dxcon* (p. 951)		
			dxcon (p. 951)		
			dxlag (p. 951)		
AssociateVirtualInterfaceWithConnection	Associates a virtual interface with a specified link aggregation group (LAG) or connection.	Write	dxvif* (p. 951)		
			dxcon (p. 951)		
			dxlag (p. 951)		
ConfirmConnection	Confirm the creation of a hosted connection on an interconnect.	Write	dxcon* (p. 951)		
ConfirmPrivateVirtualInterface	Accept ownership of a private virtual interface created by another customer.	Write	dxvif* (p. 951)		
ConfirmPublicVirtualInterface	Accept ownership of a public virtual interface created by another customer.	Write	dxvif* (p. 951)		
ConfirmTransitVirtualInterface	Accept ownership of a transit virtual interface created by another customer.	Write	dxvif* (p. 951)		
CreateBGPPeer	Creates a BGP peer on the specified virtual interface.	Write	dxvif* (p. 951)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateConnection	Creates a new connection between the customer network and a specific AWS Direct Connect location.	Write	dxlag (p. 951)		
	aws:RequestTag/ {\$TagKey} (p. 952)		aws:TagKeys (p. 952)		
CreateDirectConnectGateway	Creates a Direct Connect gateway which is an intermediate object that enables you to connect a set of virtual interfaces and virtual private gateways.	Write			
CreateDirectConnectAssociation	Creates an association between a Direct Connect gateway and a virtual private gateway.	Write	dx-gateway* (p. 951)		
CreateDirectConnectProposal	Creates a proposal to associate the specified virtual private proposal gateway with the specified Direct Connect gateway.	Write	dx-gateway* (p. 951)		
CreateInterconnect	Creates a new interconnect between a AWS Direct Connect partner's network and a specific AWS Direct Connect location.	Write	dxlag (p. 951)		
aws:RequestTag/ {\$TagKey} (p. 952)	aws:TagKeys (p. 952)				
CreateLag	Creates a link aggregation group (LAG) with the specified number of bundled physical connections between the customer network and a specific AWS Direct Connect location.	Write	dxcon (p. 951)		
aws:RequestTag/ {\$TagKey} (p. 952)	aws:TagKeys (p. 952)				
CreatePrivateVirtualInterface	Creates a new private virtual interface	Write	dxcon (p. 951)		
	dxlag (p. 951)				

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:RequestTag/ \${TagKey} (p. 952) aws:TagKeys (p. 952)	
CreatePublicVirtualInterface	Creates a new public virtual interface	Write	dxcon (p. 951)		
			dxlag (p. 951)		
				aws:RequestTag/ \${TagKey} (p. 952) aws:TagKeys (p. 952)	
CreateTransitVirtualInterface	Creates a new transit virtual interface	Write	dxcon (p. 951)		
			dxlag (p. 951)		
				aws:RequestTag/ \${TagKey} (p. 952) aws:TagKeys (p. 952)	
DeleteBGPPeer	Deletes the specified BGP peer on the specified virtual interface with the specified customer address and ASN.	Write	dxvif* (p. 951)		
DeleteConnection	Deletes the connection.	Write	dxcon* (p. 951)		
DeleteDirectConnectGateway	Deletes the specified Direct Connect gateway.	Write	dx-gateway* (p. 951)		
DeleteDirectConnection	Deletes the association between the specified Direct Connect gateway and virtual private gateway.	Write	dx-gateway* (p. 951)		
DeleteDirectConnectionProposal	Deletes the association proposal request between the specified Direct Connect gateway and virtual private gateway.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteInterconnect	Deletes the specified interconnect.	Write	dxcon* (p. 951)		
DeleteLag	Deletes the specified link aggregation group (LAG).	Write	dxlag* (p. 951)		
DeleteVirtualInterface	Deletes a virtual interface.	Write	dxvif* (p. 951)		
DescribeConnection	Returns the LOA-CFA for a connection.	Read	dxcon* (p. 951)		
DescribeConnections	Displays all connections in this region.	Read	dxcon (p. 951)		
DescribeConnections	Return a list of connections that have been provisioned on the given interconnect.	Read	dxcon* (p. 951)		
DescribeDirectConnectAssociationProposals	Describes one or more connection between a virtual private gateway and a Direct Connect gateway.	Read	dx-gateway (p. 951)		
DescribeDirectConnectAssociations	Lists the associations between your Direct Connect gateways and virtual private gateways.	Read	dx-gateway (p. 951)		
DescribeDirectConnectAttachments	Lists the attachments between your Direct Connect gateways and virtual interfaces.	Read	dx-gateway (p. 951)		
DescribeDirectConnectGateways	Lists all your Direct Connect gateways or only the specified Direct Connect gateway.	Read	dx-gateway (p. 951)		
DescribeHostedConnections	Lists the hosted connections that have been provisioned on the specified interconnect or link aggregation group (LAG).	Read	dxcon (p. 951)		
DescribeInterconnects	Returns the LOA-CFA for an interconnect.	Read	dxcon* (p. 951)		
DescribeInterconnects	Returns a list of interconnects owned by the AWS account.	Read	dxcon (p. 951)		
DescribeLags	Describes all your link aggregation groups (LAG) or the specified LAG.	Read	dxlag (p. 951)		
DescribeLoa	Gets the LOA-CFA for a connection, interconnect, or link aggregation group (LAG).	Read	dxcon (p. 951)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			dxlag (p. 951)		
DescribeLocations	Returns the list of AWS Direct Connect locations in the current AWS region.	List			
DescribeTags	Describes the tags associated with the specified AWS Direct Connect resources.	Read	dxcon (p. 951)		
			dxlag (p. 951)		
			dxvif (p. 951)		
DescribeVirtualGateways	Returns a list of virtual private gateways owned by the AWS account.	Read			
DescribeVirtualInterfaces	Displays all virtual interfaces for an AWS account.	Read	dxcon (p. 951)		
			dxlag (p. 951)		
			dxvif (p. 951)		
DisassociateConnectionAggregates	Disassociates a connection from a link aggregation group (LAG).	Write	dxcon* (p. 951)		
			dxlag* (p. 951)		
TagResource	Adds the specified tags to the specified AWS Direct Connect resource. Each resource can have a maximum of 50 tags.	Tagging	dxcon (p. 951)		
			dxlag (p. 951)		
			dxvif (p. 951)		
			aws:RequestTag/\${TagKey} (p. 952)	aws:TagKeys (p. 952)	
UntagResource	Removes one or more tags from the specified AWS Direct Connect resource.	Tagging	dxcon (p. 951)		
			dxlag (p. 951)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			dxvif (p. 951)		
				aws:TagKeys (p. 952)	
UpdateDirectConnectGateway	Updates the specified attributes of the Direct Connect gateway association.	Write			
UpdateLag	Updates the attributes of the specified link aggregation group (LAG).	Write	dxlag* (p. 951)		
UpdateVirtualInterface	Updates the specified attributes of the specified virtual private interface.	Write	dxvif* (p. 951)		

Resource Types Defined by AWS Direct Connect

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 944\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
dxcon	arn:\${Partition}:directconnect:\${Region}: \${Account}:dxcon/\${ConnectionId}	aws:ResourceTag/ {\$TagKey} (p. 952)
dxlag	arn:\${Partition}:directconnect:\${Region}: \${Account}:dxlag/\${LagId}	aws:ResourceTag/ {\$TagKey} (p. 952)
dxvif	arn:\${Partition}:directconnect:\${Region}: \${Account}:dxvif/\${VirtualInterfaceId}	aws:ResourceTag/ {\$TagKey} (p. 952)
dx-gateway	arn:\${Partition}:directconnect:: \${Account}:dx-gateway/ \${DirectConnectGatewayId}	

Condition Keys for AWS Direct Connect

AWS Direct Connect defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
<code>aws:RequestTag/\${TagKey}</code>	Filters actions based on the presence of tag key-value pairs in the request	String
<code>aws:ResourceTag/\${TagKey}</code>	Filters actions based on tag key-value pairs attached to the resource	String
<code>aws:TagKeys</code>	Filters actions based on the presence of tag keys in the request	String

Actions, Resources, and Condition Keys for AWS Directory Service

AWS Directory Service (service prefix: `ds`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Directory Service \(p. 952\)](#)
- [Resource Types Defined by AWS Directory Service \(p. 958\)](#)
- [Condition Keys for AWS Directory Service \(p. 958\)](#)

Actions Defined by AWS Directory Service

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AcceptSharedDirectoryRequest	Accepts a directory sharing request that was sent from the directory owner account.	Write	directory* (p. 958)		
AddIpRoutes	Adds a CIDR address block to correctly route traffic to and from your Microsoft AD on Amazon Web Services	Write	directory* (p. 958)		ec2:AuthorizeSecurityGroup ec2:AuthorizeSecurityGroup ec2:DescribeSecurityGroups
AddTagsToResource	Adds or overwrites one or more tags for the specified Amazon Directory Services directory.	Tagging	directory* (p. 958)		ec2:CreateTags
			aws:RequestTag/ \${TagKey} (p. 959)		
AuthorizeApplication [permission only]	Authorizes an application for your AWS Directory.	Write	directory* (p. 958)		
			aws:TagKeys (p. 959)		
CancelSchemaExtension	Cancels an in-progress schema extension to a Microsoft AD directory.	Write	directory* (p. 958)		
CheckAlias [permission only]	Verifies that the alias is available for use.	Read			
ConnectDirectory	Creates an AD Connector to connect to an on-premises directory.	Tagging		aws:RequestTag/ \${TagKey} (p. 959)	ec2:AuthorizeSecurityGroup ec2:AuthorizeSecurityGroup
			aws:TagKeys (p. 959)		ec2>CreateNetworkInterface ec2:CreateSecurityGroup ec2:CreateTags ec2:DescribeNetworkInterfaces ec2:DescribeSubnets ec2:DescribeVpcs
CreateAlias	Creates an alias for a directory and assigns the alias to the directory.	Write	directory* (p. 958)		
CreateComputer	Creates a computer account in the specified directory, and joins the computer to the directory.	Write	directory* (p. 958)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateConditionalForwarder	Creates a conditional forwarder associated with your AWS directory.	Write	directory* (p. 958)		
.CreateDirectory	Creates a Simple AD directory.	Tagging		aws:RequestTag/ \${TagKey} (p. 959) aws:TagKeys (p. 959)	ec2:AuthorizeSecurityGroups ec2:CreateNetworkInterfaces ec2:CreateSecurityGroups ec2:CreateTags ec2:DescribeNetworkInterfaces ec2:DescribeSubnets ec2:DescribeVpcs
CreateIdentityPoolDirectory	Creates a IdentityPool Directory in the AWS cloud. [permission only]	Tagging		aws:RequestTag/ \${TagKey} (p. 959) aws:TagKeys (p. 959)	
CreateLogSubscription	Creates a subscription to forward real time Directory Service domain controller security logs to the specified CloudWatch log group in your AWS account.	Write	directory* (p. 958)		
CreateMicrosoftAD	Creates a Microsoft AD in the AWS cloud.	Tagging		aws:RequestTag/ \${TagKey} (p. 959) aws:TagKeys (p. 959)	ec2:AuthorizeSecurityGroups ec2:CreateNetworkInterfaces ec2:CreateSecurityGroups ec2:CreateTags ec2:DescribeNetworkInterfaces ec2:DescribeSubnets ec2:DescribeVpcs
CreateSnapshot	Creates a snapshot of a Simple AD or Microsoft AD directory in the AWS cloud.	Write	directory* (p. 958)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateTrust	Initiates the creation of the AWS side of a trust relationship between a Microsoft AD in the AWS cloud and an external domain.	Write	directory* (p. 958)		
DeleteConditionalForwarder	Deletes a conditional forwarder that has been set up for your AWS directory.	Write	directory* (p. 958)		
DeleteDirectory	Deletes an AWS Directory Service directory.	Write	directory* (p. 958)		ec2:DeleteNetworkInterface ec2:DeleteSecurityGroup ec2:DescribeNetworkInterface ec2:RevokeSecurityGroup ec2:RevokeSecurityGroup
DeleteLogSubscription	Deletes the specified log subscription.	Write	directory* (p. 958)		
DeleteSnapshot	Deletes a directory snapshot.	Write	directory* (p. 958)		
DeleteTrust	Deletes an existing trust relationship between your Microsoft AD in the AWS cloud and an external domain.	Write	directory* (p. 958)		
DeregisterEventTopic	Removes the specified directory as a publisher to the specified SNS topic.	Write	directory* (p. 958)		
DescribeConditionalForwarder	Obtains information about the conditional forwarders for this account.	Read	directory* (p. 958)		
DescribeDirectory	Obtains information about the directories that belong to this account.	List			
DescribeDomainController	Provides information about any domain controllers in your directory.	Read	directory* (p. 958)		
DescribeEventTopic	Obtains information about which SNS topics receive status messages from the specified directory.	Read	directory* (p. 958)		
DescribeSharedDirectory	Returns the shared directories in your account.	Read	directory* (p. 958)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeSnapshot	Obtains information about the directory snapshots that belong to this account.	Read			
DescribeTrusts	Obtains information about the trust relationships for this account.	Read			
DisableRadius	Disables multi-factor authentication (MFA) with the Remote Authentication Dial In User Service (RADIUS) server for an AD Connector directory.	Write	directory* (p. 958)		
DisableSso	Disables single-sign on for a directory.	Write	directory* (p. 958)		
EnableRadius	Enables multi-factor authentication (MFA) with the Remote Authentication Dial In User Service (RADIUS) server for an AD Connector directory.	Write	directory* (p. 958)		
EnableSso	Enables single-sign on for a directory.	Write	directory* (p. 958)		
GetAuthorizedApplicationDetails [permission only]		Read	directory* (p. 958)		
GetDirectoryLimits	Obtains directory limit information for the current region.	Read			
GetSnapshotLimits	Obtains the manual snapshot limits for a directory.	Read	directory* (p. 958)		
ListAuthorizedApplications [permission only]	Obtains the aws applications authorized for a directory.	Read	directory* (p. 958)		
ListIpRoutes	Lists the address blocks that you have added to a directory.	Read	directory* (p. 958)		
ListLogSubscriptions	Lists the active log subscriptions for the AWS account.	Read			
ListSchemaExtensions	Lists all schema extensions applied to a Microsoft AD Directory.	List	directory* (p. 958)		
ListTagsForResource	Lists all tags on an Amazon Directory Services directory.	Read	directory* (p. 958)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
RegisterEventTopic	Associates a directory with an SNS topic.	Write	directory* (p. 958)		sns:GetTopicAttributes
RejectSharedDirectoryRequest	Rejects a directory sharing request that was sent from the directory owner account.	Write	directory* (p. 958)		
RemoveIpRoutes	Removes IP address blocks from a directory.	Write	directory* (p. 958)		
RemoveTagsFromDirectory	Removes tags from an Amazon Directory Services directory.	Tagging	directory* (p. 958)		ec2:DeleteTags
				aws:RequestTag/\${TagKey} (p. 959)	aws:TagKeys (p. 959)
ResetUserPassword	Resets the password for any user in your AWS Managed Microsoft AD or Simple AD directory.	Write	directory* (p. 958)		
RestoreFromSnapshot	Restores a directory using an existing directory snapshot.	Write	directory* (p. 958)		
ShareDirectory	Shares a specified directory in your AWS account (directory owner) with another AWS account (directory consumer). With this operation you can use your directory from any AWS account and from any Amazon VPC within an AWS Region.	Write	directory* (p. 958)		
StartSchemaExtension	Applies a schema extension to a Microsoft AD directory.	Write	directory* (p. 958)		
UnauthorizeApplication [permission only]	Unauthorized application [permission only] to your AWS Directory.	Write	directory* (p. 958)		
UnshareDirectory	Stops the directory sharing between the directory owner and consumer accounts.	Write	directory* (p. 958)		
UpdateConditionalForwarder [that has been set up for your AWS directory]	Updates a conditional forwarder that has been set up for your AWS directory.	Write	directory* (p. 958)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UpdateNumberOfDomainControllers	Adds or removes domain controllers to or from the directory. Based on the difference between current value and new value (provided through this API call), domain controllers will be added or removed. It may take up to 45 minutes for any new domain controllers to become fully active once the requested number of domain controllers is updated. During this time, you cannot make another update request.	Write	directory* (p. 958)		
UpdateRadius	Updates the Remote Authentication Dial In User Service (RADIUS) server information for an AD Connector directory.	Write	directory* (p. 958)		
VerifyTrust	Verifies a trust relationship between your Microsoft AD in the AWS cloud and an external domain.	Read	directory* (p. 958)		

Resource Types Defined by AWS Directory Service

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 952\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
directory	<code>arn:\${Partition}:ds:\${Region}: \${Account}:directory/\${DirectoryId}</code>	aws:ResourceTag/ \${TagKey} (p. 959)

Condition Keys for AWS Directory Service

AWS Directory Service defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/ \${TagKey}		String
aws:ResourceTag/ \${TagKey}		String
aws:TagKeys		String

Actions, Resources, and Condition Keys for Amazon DynamoDB

Amazon DynamoDB (service prefix: dynamodb) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon DynamoDB \(p. 959\)](#)
- [Resource Types Defined by Amazon DynamoDB \(p. 965\)](#)
- [Condition Keys for Amazon DynamoDB \(p. 966\)](#)

Actions Defined by Amazon DynamoDB

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
BatchGetItem	Returns the attributes of one or more items from one or more tables	Read	table* (p. 966)		dynamodb:Attributes (p. 966)

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				dynamodb:LeadingKeys (p. 966) dynamodb:ReturnConsumedCapacity (p. 966) dynamodb:Select (p. 966)	
BatchWriteItem	Puts or deletes multiple items in one or more tables	Write	table* (p. 966)		
				dynamodb:Attributes (p. 966) dynamodb:LeadingKeys (p. 966) dynamodb:ReturnConsumedCapacity (p. 966)	
ConditionCheckItem	The ConditionCheckItem operation checks the existence of a set of attributes for the item with the given primary key	Read	table* (p. 966)		
				dynamodb:Attributes (p. 966) dynamodb:LeadingKeys (p. 966) dynamodb:ReturnConsumedCapacity (p. 966) dynamodb:ReturnValues (p. 966)	
CreateBackup	Creates a backup for an existing table	Write	table* (p. 966)		
CreateGlobalTable	Enables the user to create a global table from an existing table	Write	global-table* (p. 966)		
			table* (p. 966)		
 CreateTable	The CreateTable operation adds a new table to your account	Write	table* (p. 966)		
 CreateTableReplica	Adds a new replica table	Write	table* (p. 966)		
DeleteBackup	Deletes an existing backup of a table	Write	backup* (p. 966)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteItem	Deletes a single item in a table by primary key	Write	table* (p. 966)		dynamodb:Attributes (p. 966) dynamodb:EnclosingOperation (p. 966) dynamodb:LeadingKeys (p. 966) dynamodb:ReturnConsumedCapacity (p. 966) dynamodb:ReturnValues (p. 966)
DeleteTable	The DeleteTable operation deletes a table and all of its items	Write	table* (p. 966)		
DeleteTableReplica	Deletes a replica table and all of its items	Write	table* (p. 966)		
DescribeBackup	Describes an existing backup of a table	Read	backup* (p. 966)		
DescribeContinuousBackup	Checks the status of the backup restore settings on the specified table	Read	table* (p. 966)		
DescribeContributorInsightsTable	Describes the contributor insights status and related details for a given table or global secondary index	Read	table* (p. 966)		
			index (p. 965)		
DescribeGlobalTable	Returns information about the specified global table	Read	global-table* (p. 966)		
DescribeGlobalTableAboutThe	Returns settings information about the specified global table	Read	global-table* (p. 966)		
DescribeLimits	Returns the current provisioned-capacity limits for your AWS account in a region, both for the region as a whole and for any one DynamoDB table that you create there	Read			
DescribeReservedCapacity	Describes one or more of the reserved Capacity purchased	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeReservedCapacityOfferings	Describes Reserved Capacity Offerings available for purchase	Read			
DescribeStream	Returns information about a stream, including the current status of the stream, its Amazon Resource Name (ARN), the composition of its shards, and its corresponding DynamoDB table	Read	stream* (p. 965)		
DescribeTable	Returns information about the table	Read	table* (p. 966)		
DescribeTableReplicaSettings	Describes the auto scaling settings across all replicas of the global table	Read	table* (p. 966)		
DescribeTimeToLive	Gives a description of the Time to Live (TTL) status on the specified table.	Read	table* (p. 966)		
GetItem	The GetItem operation returns a set of attributes for the item with the given primary key	Read	table* (p. 966)		
				dynamodb:Attributes (p. 966)	
				dynamodb:EnclosingOperation (p. 966)	
				dynamodb:LeadingKeys (p. 966)	
				dynamodb:ReturnConsumedCapacity (p. 966)	
				dynamodb:Select (p. 966)	
GetRecords	Retrieves the stream records from a given shard	Read	stream* (p. 965)		
GetShardIterator	Returns a shard iterator	Read	stream* (p. 965)		
ListBackups	List backups associated with the account and endpoint	List			
ListContributorInsights	Lists the ContributorInsightsSummary for all tables and global secondary indexes associated with the current account and endpoint	List	table (p. 966)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListGlobalTables	Lists all global tables that have a replica in the specified region	List			
ListStreams	Returns an array of stream ARNs associated with the current account and endpoint	Read			
ListTables	Returns an array of table names associated with the current account and endpoint	List			
ListTagsOfResource	List all tags on an Amazon DynamoDB resource	Read	table* (p. 966)		
PurchaseReservedCapacityOfferings	Purchases Reserved Capacity for a specific capacity offering	Write			
PutItem	Creates a new item, or replaces an old item with a new item	Write	table* (p. 966)		
				dynamodb:Attributes (p. 966)	
				dynamodb:EnclosingOperation (p. 966)	
				dynamodb:LeadingKeys (p. 966)	
				dynamodb:ReturnConsumedCapacity (p. 966)	
Query	Uses the primary key of a table or a secondary index to directly access items from that table or index	Read	table* (p. 966)		
				index (p. 965)	
				dynamodb:Attributes (p. 966)	
				dynamodb:LeadingKeys (p. 966)	
				dynamodb:ReturnConsumedCapacity (p. 966)	
				dynamodb:ReturnValues (p. 966)	
				dynamodb:Select (p. 966)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
RestoreTableFromExistingBackup	Creates a new table from an existing backup	Write	backup* (p. 966)		
	table* (p. 966)				
RestoreTableToPointInTime	Restores a table to a point in time	Write	table* (p. 966)		
Scan	Returns one or more items and item attributes by accessing every item in a table or a secondary index	Read	table* (p. 966)		
	index (p. 965)				
				dynamodb:Attributes (p. 966)	
				dynamodb:LeadingKeys (p. 966)	
				dynamodb:ReturnConsumedCapacity (p. 966)	
				dynamodb:ReturnValues (p. 966)	
TagResource	Associate a set of tags with an Amazon DynamoDB resource	Tagging	table* (p. 966)		
UntagResource	Removes the association of tags from an Amazon DynamoDB resource.	Tagging	table* (p. 966)		
UpdateContinuousBackups	Enables or disables continuous backups	Write	table* (p. 966)		
UpdateContributorInsights	Updates the status for contributor insights for a specific table or global secondary index	Write	table* (p. 966)		
	index (p. 965)				
UpdateGlobalTableReplicas	Enables the user to add or remove replicas in the specified global table	Write	global-table* (p. 966)		
	table* (p. 966)				
UpdateGlobalTableSettings	Enables the user to update settings of the specified global table	Write	global-table* (p. 966)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			table* (p. 966)		
UpdateItem	Edits an existing item's attributes, or adds a new item to the table if it does not already exist	Write	table* (p. 966)	dynamodb:Attributes (p. 966) dynamodb:EnclosingOperation (p. 966) dynamodb:LeadingKeys (p. 966) dynamodb:ReturnConsumedCapacity (p. 966) dynamodb:ReturnValues (p. 966)	
UpdateTable	Modifies the provisioned throughput settings, global secondary indexes, or DynamoDB Streams settings for a given table	Write	table* (p. 966)		
UpdateTableReplica	Updates auto scaling settings on your replicated table	Write	table* (p. 966)		
UpdateTimeToLive	Enables or disables TTL for the specified table	Write	table* (p. 966)		

Resource Types Defined by Amazon DynamoDB

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 959\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
<code>index</code>	<code>arn:\${Partition}:dynamodb:\${Region}: \${Account}:table/\${TableName}/index/\${IndexName}</code>	
<code>stream</code>	<code>arn:\${Partition}:dynamodb:\${Region}: \${Account}:table/\${TableName}/stream/\${StreamLabel}</code>	

Resource Types	ARN	Condition Keys
table	arn:\${Partition}:dynamodb:\${Region}: \${Account}:table/\${TableName}	
backup	arn:\${Partition}:dynamodb:\${Region}: \${Account}:table/\${TableName}/backup/ \${BackupName}	
global-table	arn:\${Partition}:dynamodb: \${Account}:global-table/\${GlobalTableName}	

Condition Keys for Amazon DynamoDB

Amazon DynamoDB defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Note

For information about how to use context keys to refine DynamoDB access using an IAM policy, see [Using IAM Policy Conditions for Fine-Grained Access Control](#) in the *Amazon DynamoDB Developer Guide*.

Condition Keys	Description	Type
dynamodb:AttributeNames	Filter based on the attribute (field or column) names of the table.	String
dynamodb:EnclosingTransaction	Used to block Transactions APIs calls and allow the non-Transaction APIs calls and vice-versa.	String
dynamodb:LeadingKeys	Filters based on the partition key of the table.	String
dynamodb:ReturnConsumedCapacity	Filter based on the <code>ReturnConsumedCapacity</code> parameter of a request. Contains either "TOTAL" or "NONE".	String
dynamodb:ReturnValues	Filter based on the <code>ReturnValues</code> parameter of a request. Contains one of the following: "ALL_OLD", "UPDATED_OLD", "ALL_NEW", "UPDATED_NEW", or "NONE".	String
dynamodb:Select	Filter based on the <code>Select</code> parameter of a Query or Scan request.	String

Actions, Resources, and Condition Keys for Amazon DynamoDB Accelerator (DAX)

Amazon DynamoDB Accelerator (DAX) (service prefix: `dax`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon DynamoDB Accelerator \(DAX\) \(p. 967\)](#)
- [Resource Types Defined by Amazon DynamoDB Accelerator \(DAX\) \(p. 970\)](#)
- [Condition Keys for Amazon DynamoDB Accelerator \(DAX\) \(p. 970\)](#)

Actions Defined by Amazon DynamoDB Accelerator (DAX)

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
BatchGetItem	The <code>BatchGetItem</code> action returns the attributes of one or more items from one or more tables.	Read	application* (p. 970)		
BatchWriteItem	The <code>BatchWriteItem</code> action operation puts or deletes multiple items in one or more tables.	Write	application* (p. 970)		
ConditionCheckItem	The <code>ConditionCheckItem</code> operation checks the existence of a set of attributes for the item with the given primary key	Read	application* (p. 970)		
CreateCluster	The <code>CreateCluster</code> action creates a DAX cluster.	Write	application* (p. 970)		dax: dax: ec2: ec2: ec2: ec2:

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
					ec2:DescribeSubnets ec2:DescribeVpcs iam:GetRole iam:PassRole
CreateParameterGroup	The CreateParameterGroup action creates collection of parameters that you apply to all of the nodes in a DAX cluster.	Write			
CreateSubnetGroup	The CreateSubnetGroup action creates a new subnet group.	Write			
DecreaseReplicationFactor	The DecreaseReplicationFactor action removes one or more nodes from a DAX cluster.	Write	application* (p. 970)		
DeleteCluster	The DeleteCluster action deletes a previously provisioned DAX cluster.	Write	application* (p. 970)		
DeleteItem	The DeleteItem action deletes a single item in a table by primary key.	Write	application* (p. 970)		dax:EnclosingOperation (p. 970)
DeleteParameterGroup	The DeleteParameterGroup action deletes the specified parameter group.	Write			
DeleteSubnetGroup	The DeleteSubnetGroup action deletes a subnet group.	Write			
DescribeClusters	The DescribeClusters action returns information about all provisioned DAX clusters.	List	application (p. 970)		
DescribeDefaultParameters	The DescribeDefaultParameters action returns the default system parameter information for DAX.	List			
DescribeEvents	The DescribeEvents action returns events related to DAX clusters and parameter groups.	List			
DescribeParameterGroups	The DescribeParameterGroups action returns a list of parameter group descriptions.	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeParameters	The <code>DescribeParameters</code> action returns the detailed parameter list for a particular parameter group.	Read			
DescribeSubnetGroups	The <code>DescribeSubnetGroups</code> action returns a list of subnet group descriptions.	List			
GetItem	The <code>GetItem</code> action returns a set of attributes for the item with the given primary key.	Read	application* (p. 970)		
					dax:EnclosingOperation (p. 970)
IncreaseReplicationFactor	The <code>IncreaseReplicationFactor</code> action adds one or more nodes to a DAX cluster.	Write	application* (p. 970)		
ListTags	The <code>ListTags</code> action returns a list all of the tags for a DAX cluster.	Read	application* (p. 970)		
PutItem	The <code>PutItem</code> action creates a new item, or replaces an old item with a new item.	Write	application* (p. 970)		
					dax:EnclosingOperation (p. 970)
Query	The <code>Query</code> action finds items based on primary key values. You can query any table or secondary index that has a composite primary key (a partition key and a sort key).	Read	application* (p. 970)		
RebootNode	The <code>RebootNode</code> action reboots a single node of a DAX cluster.	Write	application* (p. 970)		
Scan	The <code>Scan</code> action returns one or more items and item attributes by accessing every item in a table or a secondary index.	Read	application* (p. 970)		
TagResource	The <code>TagResource</code> action associates a set of tags with a DAX resource.	Tagging	application* (p. 970)		
UntagResource	The <code>UntagResource</code> action removes the association of tags from a DAX resource.	Tagging	application* (p. 970)		
UpdateCluster	The <code>UpdateCluster</code> action modifies the settings for a DAX cluster.	Write	application* (p. 970)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UpdateItem	The UpdateItem action edits an existing item's attributes, or adds a new item to the table if it does not already exist.	Write	application* (p. 970)		
				dax:EnclosingOperation (p. 970)	
UpdateParameterGroup	The UpdateParameterGroup action modifies the parameters of a parameter group.	Write			
UpdateSubnetGroup	The UpdateSubnetGroup action modifies an existing subnet group.	Write			

Resource Types Defined by Amazon DynamoDB Accelerator (DAX)

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 967\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
application	arn:\${Partition}:dax:\${Region}: \${Account}:cache/\${ClusterName}	

Condition Keys for Amazon DynamoDB Accelerator (DAX)

Amazon DynamoDB Accelerator (DAX) defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
dax:EnclosingOperation	Used to block Transactions APIs calls and allow the non-transaction APIs calls and vice-versa.	String

Actions, Resources, and Condition Keys for Amazon EC2

Amazon EC2 (service prefix: ec2) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon EC2 \(p. 971\)](#)
- [Resource Types Defined by Amazon EC2 \(p. 1047\)](#)
- [Condition Keys for Amazon EC2 \(p. 1056\)](#)

Actions Defined by Amazon EC2

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AcceptReservedInstancesExchangeQuote	Accepts the Convertible Reserved Instances Exchange quote described in the <code>GetReservedInstancesExchangeQuote</code> call.	Write			
AcceptTransitGatewayAttachment	Accepts a request to attach a VPC to a transit gateway	Write	transit-gateway-attachment* (p. 1053)	ec2:Region (p. 1058) ec2:ResourceTag/{TagKey} (p. 1058)	
AcceptVpcEndpointConnections	Accepts one or more interface VPC endpoint connection requests to your VPC endpoint service.	Write			
AcceptVpcPeeringConnection	Accept a VPC peering connection request	Write	vpc* (p. 1054)	ec2:Region (p. 1058) ec2:ResourceTag/{TagKey} (p. 1058)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				ec2:Tenancy (p. 1058)	
			vpc-peering-connection* (p. 1054)	ec2:AccepterVpc (p. 1056) ec2:Region (p. 1058) ec2:RequesterVpc (p. 1058) ec2:ResourceTag/ <code> \${TagKey}</code> (p. 1058)	
AdvertiseByoipCidr	Advertises an IPv4 address range that is provisioned for use with your AWS resources through bring your own IP addresses (BYOIP)	Write			
AllocateAddress	Acquires an Elastic IP address.	Write			
AllocateHosts	Allocates a Dedicated Host to your account.	Write			
ApplySecurityGroup	Applies a security group to the association between the target network and the Client VPN endpoint.	Write	client-vpn-endpoint* (p. 1047)	ec2:Region (p. 1058) ec2:ResourceTag/ <code> \${TagKey}</code> (p. 1058)	
			security-group* (p. 1051)	ec2:Region (p. 1058) ec2:ResourceTag/ <code> \${TagKey}</code> (p. 1058)	
			vpc* (p. 1054)	ec2:Region (p. 1058) ec2:ResourceTag/ <code> \${TagKey}</code> (p. 1058)	
AssignIpv6Addresses	Assigns one or more IPv6 addresses to the specified network interface.	Write			
AssignPrivateIpAddresses	Assigns one or more secondary private IP addresses to the specified network interface.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AssociateAddress	Associates an Elastic IP address with an instance or a network interface.	Write			
AssociateClientVpnEndpoint	Associates a target network with a Client VPN endpoint .	Write	client-vpn-endpoint* (p. 1047)	ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058)	
	subnet* (p. 1052)		ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058)		
AssociateDhcpOptions	Associates a set of DHCP options (that you've previously created) with the specified VPC, or associates no DHCP options with the VPC.	Write			
AssociateIamInstanceProfile	Associates an IAM instance profile with a running or stopped instance.	Write	instance* (p. 1049)	ec2:AvailabilityZonePassRole (p. 1056) ec2:EbsOptimized (p. 1056) ec2:InstanceProfile (p. 1057) ec2:InstanceType (p. 1057) ec2:PlacementGroup (p. 1058) ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058) ec2:RootDeviceType (p. 1058) ec2:Tenancy (p. 1058)	
AssociateRouteTable	Associates a subnet with a route table.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AssociateSubnetCidrBlockWithYourSubnet	Associates a CIDR block with your subnet.	Write			
AssociateTransitGatewayAttachmentWithTheSpecifiedTransitGatewayRouteTable	Associates the specified transit gateway attachment with the specified transit gateway route table	Write	transit-gateway-attachment* (p. 1053)	ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058)	
			transit-gateway-route-table* (p. 1053)	ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058)	
AssociateVpcCidrBlockWithYourVpc	Associates a CIDR block with your VPC.	Write			
AttachClassicLinkInterface	Links an EC2-Classic instance to a ClassicLink-enabled VPC through one or more of the VPC's security groups.	Write	instance* (p. 1049)	ec2:AvailabilityZone (p. 1056) ec2:EbsOptimized (p. 1056) ec2:InstanceProfile (p. 1057) ec2:InstanceType (p. 1057) ec2:PlacementGroup (p. 1058) ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058) ec2:RootDeviceType (p. 1058) ec2:Tenancy (p. 1058)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			security-group* (p. 1051)	ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058) ec2:Vpc (p. 1059)	
	vpc* (p. 1054)		ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058) ec2:Tenancy (p. 1058)		
AttachInternetGateway	Attaches an Internet gateway to a VPC, enabling connectivity between the Internet and the VPC.	Write			
AttachNetworkInterface	Attaches a network interface to an instance.	Write			
AttachVolume	Attaches an EBS volume to a running or stopped instance and exposes it to the instance with the specified device name.	Write	instance* (p. 1049)	ec2:AvailabilityZone (p. 1056) ec2:EbsOptimized (p. 1056) ec2:InstanceProfile (p. 1057) ec2:InstanceType (p. 1057) ec2:PlacementGroup (p. 1058) ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058) ec2:RootDeviceType (p. 1058) ec2:Tenancy (p. 1058)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			volume* (p. 1053)	ec2:AvailabilityZone (p. 1056) ec2:Encrypted (p. 1056) ec2:ParentSnapshot (p. 1057) ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058) ec2:VolumeIops (p. 1058) ec2:VolumeSize (p. 1058) ec2:VolumeType (p. 1058)	
AttachVpnGateway	Attaches a virtual private gateway to a VPC.	Write			
AuthorizeClientVpnRules	Adds an ingress authorization rule to a Client VPN endpoint.	Write	client-vpn-endpoint* (p. 1047)	ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058)	
AuthorizeSecurityGroupIngress	[EC2-VPC only] Adds one or more ingress rules to a security group for use with a VPC.	Write	security-group* (p. 1051)	ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058) ec2:Vpc (p. 1059)	
AuthorizeSecurityGroupEgress	Adds one or more ingress rules to a security group.	Write	security-group* (p. 1051)	ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058) ec2:Vpc (p. 1059)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
BundleInstance	Bundles an Amazon instance store-backed Windows instance.	Write			
CancelBundleTask	Cancels a bundling operation for an instance store-backed Windows instance.	Write			
CancelCapacityReservation	Cancels the specified Capacity Reservation, releases the reserved capacity, and changes the Capacity Reservation's state to cancelled.	Write	capacity-reservation* (p. 1047)	ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)	
CancelConversionTask	Cancels an active conversion task.	Write			
CancelExportTask	Cancels an active export task.	Write			
CancelImportTask	Cancels an in-process import virtual machine or import snapshot task.	Write			
CancelReservedInstancesListing	Cancels the specified Reserved Instance listing in the Reserved Instance Marketplace.	Write			
CancelSpotFleetRequests	Cancels the specified Spot fleet requests.	Write			
CancelSpotInstanceRequests	Cancels one or more Spot instance requests.	Write			
ConfirmProductCodes	Determines whether a product code is associated with an instance.	Write			
CopyFpgaImage	Initiates the copy of an Amazon FPGA Image (AFI) from the specified source region to the current region.	Write			
CopyImage	Initiates the copy of an AMI from the specified source region to the current region.	Write			
CopySnapshot	Copies a point-in-time snapshot of an EBS volume and stores it in Amazon S3.	Write	snapshot* (p. 1051)	aws:TagKeys (p. 1056) aws:RequestTag/\${TagKey} (p. 1056) ec2:Region (p. 1058)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateCapacityReservation	Creates a new Capacity Reservation with the specified attributes.	Write			
CreateClientVpnEndpoint	Creates a Client VPN endpoint.	Write	client-vpn-endpoint* (p. 1047)	aws:RequestTag/\${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058)	
CreateClientVpnRoute	Adds a route to a network to a Client VPN endpoint.	Write	client-vpn-endpoint* (p. 1047)	ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)	
			subnet* (p. 1052)	ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)	
CreateCustomerGateway	Provides information to AWS about your VPN customer gateway device.	Write			
CreateDefaultSubnet	Creates a default subnet with a size /20 IPv4 CIDR block in the specified Availability Zone in your default VPC.	Write			
CreateDefaultVpc	Creates a default VPC with a size /16 IPv4 CIDR block and a default subnet in each Availability Zone.	Write			
CreateDhcpOptions	Creates a set of DHCP options for your VPC.	Write			
CreateEgressOnlyInternetGateway	Creates an egress-only Internet gateway for your VPC.	Write			
CreateFleet	Launches an EC2 Fleet.	Write			
CreateFlowLogs	Creates one or more flow logs to capture IP traffic for a specific network interface, subnet, or VPC.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateFpgaImage	Creates an Amazon FPGA Image (AFI) from the specified design checkpoint (DCP).	Write			
CreateImage	Creates an Amazon EBS-backed AMI from an Amazon EBS-backed instance that is either running or stopped.	Write			
CreateInstanceExportTask	Exports a running or stopped instance to an S3 bucket.	Write			
CreateInternetGateway	Creates an Internet gateway for use with a VPC.	Write			
CreateKeyPair	Creates a 2048-bit RSA key pair with the specified name.	Write			
CreateLaunchTemplate	Creates a new launch template.	Write			
CreateLaunchTemplateVersion	Creates a new version for the specified launch template .	Write	launch-template* (p. 1049)	ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058)	
CreateNatGateway	Creates a NAT gateway in the specified subnet .	Write			
CreateNetworkAcl	Creates a network ACL in a VPC.	Write			
CreateNetworkAclEntry	Creates an entry (a rule) in a network ACL with the specified rule number.	Write			
CreateNetworkInterface	Creates a network interface in the specified subnet .	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateNetworkInterface	Creates a permission for a network interface that grants certain operations to another authorized user.	Permissions management	network-interface* (p. 1050)	ec2:AuthorizedUser (p. 1056) ec2:AvailabilityZone (p. 1056) ec2:Permission (p. 1057) ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058) ec2:Subnet (p. 1058) ec2:Vpc (p. 1059) ec2:AuthorizedService (p. 1056)	
CreatePlacementGroup	Creates a placement group that you launch cluster instances into.	Write			
CreateReservedInstancesOffering	Creates a listing for Amazon EC2 Standard Reserved Instances to be sold in the Reserved Instance Marketplace.	Write			
CreateRoute	Creates a route in a route table within a VPC.	Write	route-table* (p. 1051)	ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058) ec2:Vpc (p. 1059)	
CreateRouteTable	Creates a route table for the specified VPC.	Write			
CreateSecurityGroup	Creates a security group.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateSnapshot	Creates a snapshot of an EBS volume and stores it in Amazon S3.	Write	snapshot* (p. 1051) volume* (p. 1053)	aws:TagKeys (p. 1056) aws:RequestTag/\${TagKey} (p. 1056) ec2:ParentVolume (p. 1057) ec2:Region (p. 1058)	ec2:Encrypted (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058) ec2:VolumeIops (p. 1058) ec2:VolumeSize (p. 1058) ec2:VolumeType (p. 1058)

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateSnapshots	Creates a snapshots of an EBS volumes which attached to an EC2 instance and stores them in Amazon S3.	Write	instance* (p. 1049)	ec2:AvailabilityZone (p. 1056) ec2:EbsOptimized (p. 1056) ec2:InstanceProfile (p. 1057) ec2:InstanceType (p. 1057) ec2:PlacementGroup (p. 1058) ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058) ec2:RootDeviceType (p. 1058) ec2:Tenancy (p. 1058)	
			snapshot* (p. 1051)	aws:TagKeys (p. 1056) aws:RequestTag/ {\$TagKey} (p. 1056) ec2:ParentVolume (p. 1057) ec2:Region (p. 1058)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			volume* (p. 1053)	ec2:Encrypted (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058) ec2:VolumeIops (p. 1058) ec2:VolumeSize (p. 1058) ec2:VolumeType (p. 1058)	
CreateSpotDatafeedInstances	Creates a data feed for Spot instances, enabling you to view Spot instance usage logs. You can create one data feed per AWS account.	Write			
CreateSubnet	Creates a subnet in an existing VPC.	Write			
CreateTags	Adds or overwrites one or more tags for the specified Amazon EC2 resource or resources.	Tagging	capacity-reservation (p. 1047)	aws:RequestTag/ {\$TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058)	
			client-vpn-endpoint (p. 1047)	aws:RequestTag/ {\$TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			dhcp-options (p. 1047)	aws:RequestTag/\${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)	
			fpga-image (p. 1048)	aws:RequestTag/\${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Owner (p. 1057) ec2:Public (p. 1058) ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			image (p. 1048)	aws:RequestTag/ \${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:ImageType (p. 1057) ec2:Owner (p. 1057) ec2:Public (p. 1058) ec2:Region (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058) ec2:RootDeviceType (p. 1058)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			instance (p. 1049)	aws:RequestTag/ \${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:AvailabilityZone (p. 1056) ec2:EbsOptimized (p. 1056) ec2:InstanceProfile (p. 1057) ec2:InstanceType (p. 1057) ec2:PlacementGroup (p. 1058) ec2:Region (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058) ec2:RootDeviceType (p. 1058) ec2:Tenancy (p. 1058)	
		internet-gateway (p. 1049)	aws:RequestTag/ \${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058)	aws:RequestTag/ \${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			network-acl (p. 1050)	aws:RequestTag/ \${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058) ec2:Vpc (p. 1059)	
			network-interface (p. 1050)	aws:RequestTag/ \${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:AvailabilityZone (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058) ec2:Subnet (p. 1058) ec2:Vpc (p. 1059)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			reserved-instances (p. 1050)	aws:RequestTag/ {\$TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:AvailabilityZone (p. 1056) ec2:InstanceType (p. 1057) ec2:Region (p. 1058) ec2:ReservedInstancesOfferingType (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058) ec2:Tenancy (p. 1058)	
		route-table (p. 1051)		aws:RequestTag/ {\$TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058) ec2:Vpc (p. 1059)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			security-group (p. 1051)	aws:RequestTag/ \${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058) ec2:Vpc (p. 1059)	
			snapshot (p. 1051)	aws:RequestTag/ \${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Owner (p. 1057) ec2:ParentVolume (p. 1057) ec2:Region (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058) ec2:SnapshotTime (p. 1058) ec2:VolumeSize (p. 1058)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			spot-instance-request (p. 1051)	aws:RequestTag/ \${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058)	
			subnet (p. 1052)	aws:RequestTag/ \${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:AvailabilityZone (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058) ec2:Vpc (p. 1059)	
			traffic-mirror-filter (p. 1052)	aws:RequestTag/ \${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			traffic-mirror-session (p. 1052)	aws:RequestTag/\${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)	
			traffic-mirror-target (p. 1052)	aws:RequestTag/\${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)	
			transit-gateway (p. 1053)	aws:RequestTag/\${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)	
			transit-gateway-attachment (p. 1053)	aws:RequestTag/\${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			transit-gateway-route-table (p. 1053)	aws:RequestTag/\${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)	
			volume (p. 1053)	aws:RequestTag/\${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:AvailabilityZone (p. 1056) ec2:Encrypted (p. 1056) ec2:ParentSnapshot (p. 1057) ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058) ec2:VolumeIops (p. 1058) ec2:VolumeSize (p. 1058) ec2:VolumeType (p. 1058)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			vpc (p. 1054)	aws:RequestTag/ \${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058) ec2:Tenancy (p. 1058)	
			vpn-connection (p. 1055)	aws:RequestTag/ \${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058)	
			vpn-gateway (p. 1056)	aws:RequestTag/ \${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058)	
				ec2>CreateAction (p. 1056)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
	Creates a Traffic Mirror filter. CreateTrafficMirrorFilter	Write	traffic-mirror-filter* (p. 1052)	aws:RequestTag/ {\$TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058)	
	Creates a Traffic Mirror filter rule. CreateTrafficMirrorFilterRule	Write	traffic-mirror-filter* (p. 1052)	ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058)	
			traffic-mirror-filter-rule* (p. 1052)	ec2:Region (p. 1058)	
	Creates a Traffic Mirror session. CreateTrafficMirrorSession	Write	network-interface* (p. 1050)	ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058)	
			traffic-mirror-filter* (p. 1052)	ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058)	
			traffic-mirror-session* (p. 1052)	aws:RequestTag/ {\$TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058)	
			traffic-mirror-target* (p. 1052)	ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateTrafficMirrorTarget	Creates a Traffic Mirror target.	Write	traffic-mirror-target* (p. 1052)	aws:RequestTag/\${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058)	
			network-interface (p. 1050)	ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)	
CreateTransitGateway	Creates a transit gateway.	Write	transit-gateway* (p. 1053)	aws:RequestTag/\${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058)	
CreateTransitGatewayRouteTable	Creates a static route for the specified transit gateway route table.	Write	transit-gateway-route-table* (p. 1053)	ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)	
	transit-gateway-attachment (p. 1053)		ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)		
CreateTransitGatewayRouteTable	Creates a route table for the specified transit gateway.	Write	transit-gateway* (p. 1053)	ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			transit-gateway-route-table* (p. 1053)	aws:RequestTag/\${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058)	
CreateTransitGatewayAttachment	Attaches the specified VPC to the specified transit gateway.	Write	transit-gateway* (p. 1053)	ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)	
			transit-gateway-attachment* (p. 1053)	aws:RequestTag/\${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058)	
			vpc* (p. 1054)	ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)	
			subnet (p. 1052)	ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateVolume	Creates an EBS volume that can be attached to an instance in the same Availability Zone.	Write	volume* (p. 1053)	aws:RequestTag/ {\$TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:AvailabilityZone (p. 1056) ec2:Encrypted (p. 1056) ec2:ParentSnapshot (p. 1057) ec2:Region (p. 1058) ec2:VolumeIops (p. 1058) ec2:VolumeSize (p. 1058) ec2:VolumeType (p. 1058)	
CreateVpc	Creates a VPC with the specified CIDR block.	Write			
CreateVpcEndpoint	Creates a VPC endpoint for a specified AWS service.	Write			route53:AssociateVPCWithHostedZone
CreateVpcEndpointConfiguration	Creates a connection notification for a specified VPC endpoint or VPC endpoint service.	Write			
CreateVpcEndpointConnection	Creates a VPC endpoint service configuration for which service consumers (AWS accounts, IAM users, and IAM roles) can connect.	Write			
CreateVpcPeeringConnection	Requests a VPC peering connection between two VPCs: a requester VPC that you own and a peer VPC with which to create the connection.	Write	vpc* (p. 1054)	ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058) ec2:Tenancy (p. 1058)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			vpc-peering-connection* (p. 1054)	ec2:AcceptorVpc (p. 1056) ec2:Region (p. 1058) ec2:RequesterVpc (p. 1058)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateVpnConnection	Creates a VPN connection between an existing virtual private gateway and a VPN customer gateway.	Write	vpn-connection* (p. 1055)	ec2:Region (p. 1058) ec2:AuthenticationType (p. 1056) ec2:DPDTimeoutSeconds (p. 1056) ec2:GatewayType (p. 1056) ec2:IKEVersions (p. 1056) ec2:InsideTunnelCidr (p. 1057) ec2:Phase1DHGroupNumbers (p. 1057) ec2:Phase2DHGroupNumbers (p. 1057) ec2:Phase1EncryptionAlgorithms (p. 1057) ec2:Phase2EncryptionAlgorithms (p. 1057) ec2:Phase1IntegrityAlgorithms (p. 1057) ec2:Phase2IntegrityAlgorithms (p. 1057) ec2:Phase1LifetimeSeconds (p. 1057) ec2:Phase2LifetimeSeconds (p. 1058) ec2:PresharedKeys (p. 1058) ec2:RekeyFuzzPercentage (p. 1058) ec2:RekeyMarginTimeSeconds (p. 1058) ec2:RoutingType (p. 1058)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateVpnConnectionWithPvPN	Creates a static route associated with a VPN connection between an existing virtual private gateway and a VPN customer gateway.	Write			
CreateVpnGateway	Creates a virtual private gateway.	Write			
DeleteClientVpnEndpoint	Deletes the specified Client VPN endpoint.	Write	client-vpn-endpoint* (p. 1047)	ec2:Region (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058)	
DeleteClientVpnRoute	Deletes a route from a Client VPN endpoint.	Write	client-vpn-endpoint* (p. 1047)	ec2:Region (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058)	
	subnet (p. 1052)		ec2:Region (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058)		
DeleteCustomerGateway	Deletes the specified customer gateway.	Write	customer-gateway* (p. 1047)	ec2:Region (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058)	
DeleteDhcpOptions	Deletes the specified set of DHCP options.	Write	dhcp-options* (p. 1047)	ec2:Region (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058)	
DeleteEgressOnlyInternetGateway	Deletes the specified egress-only Internet gateway.	Write			
DeleteFleets	Deletes the specified EC2 Fleet.	Write			
DeleteFlowLogs	Deletes one or more flow logs.	Write			
DeleteFpgaImage	Deletes the specified Amazon FPGA Image (AFI).	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteInternetGateway	Deletes the specified Internet gateway.	Write	internet-gateway* (p. 1049)	ec2:Region (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058)	
DeleteKeyPair	Deletes the specified key pair, by removing the public key from Amazon EC2.	Write			
DeleteLaunchTemplate	Deletes the specified launch template and all associated versions.	Write	launch-template* (p. 1049)	ec2:Region (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058)	
DeleteLaunchTemplateVersion	Deletes the specified versions for the specified launch template.	Write	launch-template* (p. 1049)	ec2:Region (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058)	
DeleteNatGateway	Deletes the specified NAT gateway.	Write			
DeleteNetworkACLEntry	Deletes the specified network ACL entry (rule) from the specified network ACL.	Write	network-acl* (p. 1050)	ec2:Region (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058) ec2:Vpc (p. 1059)	
DeleteNetworkACLEntry	Deletes the specified ingress or egress entry (rule) from the specified network ACL.	Write	network-acl* (p. 1050)	ec2:Region (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058) ec2:Vpc (p. 1059)	
DeleteNetworkInterface	Deletes the specified network interface. You must detach the network interface before you can delete it.	Write			
DeleteNetworkInterfacePermission	Deletes a permission associated with a network interface.	Permissions management			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeletePlacementGroup	Deletes the specified placement group.	Write			
DeleteRoute	Deletes the specified route from the specified route table.	Write	route-table* (p. 1051)	ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058) ec2:Vpc (p. 1059)	
DeleteRouteTable	Deletes the specified route table.	Write	route-table* (p. 1051)	ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058) ec2:Vpc (p. 1059)	
DeleteSecurityGroup	Deletes a security group.	Write	security-group* (p. 1051)	ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058) ec2:Vpc (p. 1059)	
DeleteSnapshot	Deletes the specified snapshot.	Write	snapshot* (p. 1051)	ec2:Owner (p. 1057) ec2:ParentVolume (p. 1057) ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058) ec2:SnapshotTime (p. 1058) ec2:VolumeSize (p. 1058)	
DeleteSpotDatafeedInstances	Deletes the data feed for Spot instances.	Write			
DeleteSubnet	Deletes the specified subnet.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteTags	Deletes the specified set of tags from the specified set of resources.	Tagging	capacity-reservation (p. 1047)	aws:RequestTag/\${TagKey} (p. 1056)	
	aws:TagKeys (p. 1056)				
	ec2:Region (p. 1058)				
	ec2:ResourceTag/\${TagKey} (p. 1058)				
			client-vpn-endpoint (p. 1047)	aws:RequestTag/\${TagKey} (p. 1056)	
				aws:TagKeys (p. 1056)	
				ec2:Region (p. 1058)	
				ec2:ResourceTag/\${TagKey} (p. 1058)	
			dhcp-options (p. 1047)	aws:RequestTag/\${TagKey} (p. 1056)	
				aws:TagKeys (p. 1056)	
				ec2:Region (p. 1058)	
				ec2:ResourceTag/\${TagKey} (p. 1058)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			image (p. 1048)	aws:RequestTag/ {\$TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058)	
			instance (p. 1049)	aws:RequestTag/ {\$TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058)	
			internet-gateway (p. 1049)	aws:RequestTag/ {\$TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058)	
			network-acl (p. 1050)	aws:RequestTag/ {\$TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			network-interface (p. 1050)	aws:RequestTag/ \${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058)	
			reserved-instances (p. 1050)	aws:RequestTag/ \${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058)	
			route-table (p. 1051)	aws:RequestTag/ \${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058)	
			security-group (p. 1051)	aws:RequestTag/ \${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			snapshot (p. 1051) aws:RequestTag/ \${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058)		
			spot-instance-request (p. 1051) aws:RequestTag/ \${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058)		
			subnet (p. 1052) aws:RequestTag/ \${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058)		
			transit-gateway (p. 1053) aws:RequestTag/ \${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			transit-gateway-attachment (p. 1053)	aws:RequestTag/\${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)	
			transit-gateway-route-table (p. 1053)	aws:RequestTag/\${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)	
			volume (p. 1053)	aws:RequestTag/\${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)	
			vpc (p. 1054)	aws:RequestTag/\${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			vpn-connection (p. 1055)	aws:RequestTag/\${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)	
	vpn-gateway (p. 1056)		aws:RequestTag/\${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)		
DeleteTrafficMirrorFilter	Deletes the specified Traffic Mirror filter.	Write	traffic-mirror-filter* (p. 1052)	ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)	
DeleteTrafficMirrorFilterRule	Deletes the specified Traffic Mirror rule.	Write	traffic-mirror-filter* (p. 1052)	ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)	
			traffic-mirror-filter-rule* (p. 1052)	ec2:Region (p. 1058)	
DeleteTrafficMirrorSession	Deletes the specified Traffic Mirror session.	Write	traffic-mirror-session* (p. 1052)	ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteTrafficMirrorTarget	Deletes the specified Traffic Mirror target.	Write	traffic-mirror-target* (p. 1052)	ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)	
DeleteTransitGateway	Deletes the specified transit gateway.	Write	transit-gateway* (p. 1053)	ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)	
DeleteTransitGatewayRouteTable	Deletes the specified route from the specified transit gateway route table.	Write	transit-gateway-route-table* (p. 1053)	ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)	
DeleteTransitGatewayRouteTableAttachment	Deletes the specified transit gateway route table attachment.	Write	transit-gateway-route-table* (p. 1053)	ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)	
DeleteTransitGatewayAttachment	Deletes the specified VPC attachment.	Write	transit-gateway-attachment* (p. 1053)	ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteVolume	Deletes the specified EBS volume.	Write	volume* (p. 1053)	ec2:AvailabilityZone (p. 1056) ec2:Encrypted (p. 1056) ec2:ParentSnapshot (p. 1057) ec2:Region (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058) ec2:VolumeIops (p. 1058) ec2:VolumeSize (p. 1058) ec2:VolumeType (p. 1058)	
DeleteVpc	Deletes the specified VPC. You must detach or delete all gateways and resources that are associated with the VPC before you can delete it.	Write			
DeleteVpcEndpointConnectionNotifications	Deletes one or more VPC endpoint connection notifications.	Write			
DeleteVpcEndpointServiceConfigurations	Deletes one or more VPC endpoint service configurations in your account.	Write			
DeleteVpcEndpoints	Deletes one or more specified VPC endpoints.	Write			
DeleteVpcPeeringConnection	Deletes a VPC peering connection.	Write	vpc-peering-connection* (p. 1054)	ec2:AcceptorVpc (p. 1056) ec2:Region (p. 1058) ec2:RequesterVpc (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteVpnConnection	Deletes a VPC peering connection.	Write			
DeleteVpnConnectionRoute	Deletes the specified static route associated with a VPN connection between an existing virtual private gateway and a VPN customer gateway.	Write			
DeleteVpnGateway	Deletes the specified virtual private gateway.	Write			
DeprovisionByoipCidr	Releases the specified address range that you provisioned for use with your AWS resources through bring your own IP addresses (BYOIP) and deletes the corresponding address pool.	Write			
DeregisterImage	Deregisters the specified AMI.	Write			
DescribeAccountAttributes	Describes attributes of your AWS accounts.	List			
DescribeAddresses	Describes one or more of your Elastic IP addresses.	List			
DescribeAggregateSettings	Describes the longer ID format settings for all resource types in a specific region.	List			
DescribeAvailabilityZones	Describes one or more of the Availability Zones that are available to you.	List			
DescribeBundleTasks	Describes one or more of your bundling tasks.	List			
DescribeByoipCidr	Describes the IP address ranges that were specified in calls to ProvisionByoipCidr.	List			
DescribeCapacityReservations	Describes one or more of your Capacity Reservations.	List			
DescribeClassicLinkInstances	Describes one or more of your linked EC2-Classic instances.	List			
DescribeClientVpnRules	Describes the authorization rules for a specified Client VPN endpoint.	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeClientVpnConnections	Describes active client connections and connections that have been terminated within the last 60 minutes for the specified Client VPN endpoint.	List			
DescribeClientVpnEndpoints	Describes one or more Client VPN endpoints in the account.	List			
DescribeClientVpnRoutes	Describes the routes for the specified Client VPN endpoint.	List			
DescribeClientVpnTargetNetworks	Describes the target networks associated with the specified Client VPN endpoint.	List			
DescribeConversionTasks	Describes one or more of your conversion tasks.	List			
DescribeCustomerGateways	Describes one or more of your customer gateways.	List			
DescribeDhcpOptions	Describes one or more of your DHCP options sets.	List			
DescribeEgressOnlyInternetGateways	Describes one or more of your egress-only Internet gateways.	List			
DescribeElasticGpus	Describes the Elastic GPUs associated with your instances.	Read			
DescribeExportImageTasks	Describes the specified export image tasks or all your export image tasks.	List			
DescribeExportTasks	Describes one or more of your export tasks.	List			
DescribeFastSnapshotRestores	Describes the state of fast snapshot restores for your snapshots	Read			
DescribeFleetHistory	Describes the events for the specified EC2 Fleet during the specified time.	List			
DescribeFleetInstances	Describes the running instances for the specified EC2 Fleet.	List			
DescribeFleets	Describes one or more of your EC2 Fleets.	List			
DescribeFlowLogs	Describes one or more flow logs.	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeFpgaImage	Describes the specified attribute of the specified Amazon FPGA Images (AFI).	List			
DescribeFpgaImageAttribute	Describes one or more of the Amazon FPGA Images (AFIs) available to you.	List			
DescribeHostReservations	Describes the Dedicated Host Reservations that are available to purchase.	List			
DescribeHostReservations	Describes Dedicated Host Reservations which are associated with Dedicated Hosts in your account.	List			
DescribeHosts	Describes one or more of your Dedicated Hosts.	List			
DescribeIamInstanceProfileAssociations	Describes your IAM instance profile associations.	List			
DescribeIdFormat	Describes the ID format settings for your resources on a per-region basis, for example, to view which resource types are enabled for longer IDs.	List			
DescribeIdentityForResources	Describes the ID format settings for resources for the specified IAM user, IAM role, or root user.	List			
DescribeImageAttribute	Describes the specified attribute of the specified AMI.	List			
DescribeImages	Describes one or more of the images (AMIs, AKIs, and ARIs) available to you.	List			
DescribeImportVirtualMachineTasks	Displays details about an import virtual machine or import snapshot tasks that are already created.	List			
DescribeImportSnapshotTasks	Describes your import snapshot tasks.	List			
DescribeInstanceAttribute	Describes the specified attribute of the specified instance.	List			
DescribeInstancesCPUUsageInformation	Describes the credit option for CPU usage information for one or more of your instances.	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeInstances	Describes the status of one or more instances.	List			
DescribeInstanceTypes	Describes all instance types offered in an AWS Region.	List			
DescribeInstances	Describes one or more of your instances.	List			
DescribeInternetGateways	Describes one or more of your Internet gateways.	List			
DescribeKeyPairs	Describes one or more of your key pairs.	List			
DescribeLaunchTemplates	Describes one or more of your launch template versions.	List			
DescribeLaunchTemplates	Describes one or more of your launch templates.	List			
DescribeMovingAddresses	Describes your Elastic IP addresses that are being moved to the EC2-VPC platform, or that are being restored to the EC2-Classic platform.	List			
DescribeNatGateways	Describes one or more of your NAT gateways.	List			
DescribeNetworkACLs	Describes one or more of your Network ACLs.	List			
DescribeNetworkInterfaces	Describes a network interface attribute. You can specify only one attribute at a time.	List			
DescribeNetworkInterfacePermissions	Describes the permissions associated with a network interface.	List			
DescribeNetworkInterfaces	Describes one or more of your network interfaces.	List			
DescribePlacementGroups	Describes one or more of your placement groups.	List			
DescribePrefixLists	Describes available AWS services in a prefix list format, which includes the prefix list name and prefix list ID of the service and the IP address range for the service.	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribePrincipalIdFormat	Describes the ID format settings for the root user and all IAM roles and IAM users that have explicitly specified a longer ID (17-character ID) preference.	List			
DescribePublicIpv4AddressPools	Describes the specified IPv4 address pools.	List			
DescribeRegions	Describes one or more regions that are currently available to you.	List			
DescribeReservedInstances	Describes one or more of the Reserved Instances that you purchased.	List			
DescribeReservedInstancesListings	Describes your account's Reserved Instance listings in the Reserved Instance Marketplace.	List			
DescribeReservedInstancesModifications	Describes the modifications made to your Reserved Instances.	List			
DescribeReservedInstancesOfferings	Describes Reserved Instance offerings that are available for purchase.	List			
DescribeRouteTables	Describes one or more of your route tables.	List			
DescribeScheduledActions	Finds available schedules that meet the specified criteria.	Read			
DescribeScheduledInstances	Describes one or more of your Scheduled Instances.	Read			
DescribeSecurityGroups	[EC2-VPC only] Describes the security groups on the other side of a VPC peering connection that are referencing the security groups you've specified in this request.	List			
DescribeSecurityGroups	Describes one or more of your security groups.	List			
DescribeSnapshotAttribute	Describes the specified attribute of the specified snapshot.	List			
DescribeSnapshots	Describes one or more of the EBS snapshots available to you.	List			
DescribeSpotDatafeedSubscription	Describes the data feed for Spot instances.	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeSpotFleetForTheSpecifiedSpotFleet	Describes the running instances for the specified Spot fleet.	List			
DescribeSpotFleetEventsForTheSpecifiedSpotFleetRequest	Describes the events for the specified Spot fleet request during the specified time.	List			
DescribeSpotFleetRequests	Describes your Spot fleet requests.	List			
DescribeSpotInstanceRequestsThatBelongToYourAccount	Describes the Spot instance requests that belong to your account.	List			
DescribeSpotPriceHistory	Describes the Spot price history.	List			
DescribeStaleSecurityGroupRules	[EC2-VPC only] Describes the stale security group rules for security groups in a specified VPC.	List			
DescribeSubnets	Describes one or more of your subnets.	List			
DescribeTags	Describes one or more of the tags for your EC2 resources.	Read			
DescribeTrafficMirrorFilters	Describes one or more Traffic Mirror filters.	List			
DescribeTrafficMirrorSessions	Describes one or more Traffic Mirror sessions.	List			
DescribeTrafficMirrorTargets	Describes one or more Traffic Mirror targets.	List			
DescribeTransitGatewayAttachments	Describes one or more attachments between resources and transit gateways.	List			
DescribeTransitGatewayRouteTables	Describes one or more transit gateway route tables.	List			
DescribeTransitGatewayAttachments	Describes one or more VPC attachments.	List			
DescribeTransitGateways	Describes one or more transit gateways.	List			
DescribeVolumeAttributes	Describes the specified attribute of the specified volume.	List			
DescribeVolumeStatus	Describes the status of the specified volumes.	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeVolumes	Describes the specified EBS volumes.	List			
DescribeVolumesStatus	Reports the current modification status of EBS volumes.	Read			
DescribeVpcAttribute	Describes the specified attribute of the specified VPC.	List			
DescribeVpcClassicLinkStatus	Describes the ClassicLink status of one or more VPCs.	List			
DescribeVpcClassicLinkSupportStatus	Describes the ClassicLink DNS support status of one or more VPCs.	List			
DescribeVpcEndpointNotifications	Describes the connection notifications for VPC endpoints and VPC endpoint services.	List			
DescribeVpcEndpointConnections	Describes the VPC endpoint connections to your VPC endpoint services, including any endpoints that are pending your acceptance.	List			
DescribeVpcEndpointServiceConfigurations	Describes the VPC endpoint service configurations in your account (your services).	List			
DescribeVpcEndpointConsumers	Describes the principals (service consumers) that are permitted to discover your VPC endpoint service.	List			
DescribeVpcEndpointServices	Describes all supported AWS services that can be specified when creating a VPC endpoint.	List			
DescribeVpcEndpoints	Describes one or more of your VPC endpoints.	List			
DescribeVpcPeerings	Describes one or more of your VPC peerings connections.	List			
DescribeVpcs	Describes one or more of your VPCs.	List			
DescribeVpnConnections	Describes one or more of your VPN connections.	Read			
DescribeVpnGateways	Describes one or more of your virtual private gateways.	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DetachClassicLink	Unlinks (detaches) a linked EC2-Classic instance from a VPC.	Write	instance* (p. 1049)	ec2:AvailabilityZone (p. 1056) ec2:EbsOptimized (p. 1056) ec2:InstanceProfile (p. 1057) ec2:InstanceType (p. 1057) ec2:PlacementGroup (p. 1058) ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058) ec2:RootDeviceType (p. 1058) ec2:Tenancy (p. 1058)	
	vpc* (p. 1054)		ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058) ec2:Tenancy (p. 1058)		
DetachInternetGateway	Detaches an Internet gateway from a VPC, disabling connectivity between the Internet and the VPC.	Write			
DetachNetworkInterface	Detaches a network interface from an instance.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DetachVolume	Detaches an EBS volume from an instance.	Write	instance* (p. 1049)	ec2:AvailabilityZone (p. 1056) ec2:EbsOptimized (p. 1056) ec2:InstanceProfile (p. 1057) ec2:InstanceType (p. 1057) ec2:PlacementGroup (p. 1058) ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058) ec2:RootDeviceType (p. 1058) ec2:Tenancy (p. 1058)	
			volume* (p. 1053)	ec2:AvailabilityZone (p. 1056) ec2:Encrypted (p. 1056) ec2:ParentSnapshot (p. 1057) ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058) ec2:VolumeIops (p. 1058) ec2:VolumeSize (p. 1058) ec2:VolumeType (p. 1058)	
DetachVpnGateway	Detaches a virtual private gateway from a VPC.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DisableEbsEncryptionByDefault	Disable the default EBS encryption enabled for your account in the current region	Write			
DisableFastSnapshotRestores	Disables fast snapshot restores for the specified snapshots in the specified Availability Zones	Write	snapshot* (p. 1051)	ec2:Owner (p. 1057) ec2:ParentVolume (p. 1057) ec2:Region (p. 1058) ec2:AvailabilityZone (p. 1056) ec2:SnapshotTime (p. 1058) ec2:Encrypted (p. 1056) ec2:VolumeSize (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058)	
DisableTransitGatewayAttachmentFromPropagation	Disables the specified resource attachment from propagating routes to the specified propagation route table.	Write	transit-gateway-attachment* (p. 1053)	ec2:Region (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058)	
			transit-gateway-route-table* (p. 1053)	ec2:Region (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058)	
DisableVgwRoutePropagation	Disables a virtual private gateway (VGW) from propagating routes to a specified route table of a VPC.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DisableVpcClassicLink	Disables ClassicLink for a VPC.	Write	vpc* (p. 1054)	ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058) ec2:Tenancy (p. 1058)	
DisableVpcClassicLinkSupportForAmp	Disables ClassicLink DNS support for a VPC.	Write			
DisassociateAddress	Disassociates an Elastic IP address from the instance or network interface it's associated with.	Write			
DisassociateClientVpnEndpoint	Disassociates a target network from the specified Client VPN endpoint.	Write	client-vpn-endpoint* (p. 1047)	ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058)	
DisassociateIamInstanceProfile	Disassociates an IAM instance profile from a running or stopped instance.	Write	instance* (p. 1049)	ec2:AvailabilityZone (p. 1056) ec2:EbsOptimized (p. 1056) ec2:InstanceProfile (p. 1057) ec2:InstanceType (p. 1057) ec2:PlacementGroup (p. 1058) ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058) ec2:RootDeviceType (p. 1058) ec2:Tenancy (p. 1058)	
DisassociateRouteTable	Disassociates a subnet from a route table.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DisassociateSubnetCidrBlock	Disassociates a CIDR block from a subnet.	Write			
DisassociateTransitGatewayAttachmentFromTable	Disassociates a resource attachment from a transit gateway route table.	Write	transit-gateway-attachment* (p. 1053)	ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058)	
			transit-gateway-route-table* (p. 1053)	ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058)	
DisassociateVpcCidrBlock	Disassociates a CIDR block from a VPC.	Write			
EnableEbsEncryptionByDefault	Enables EBS encryption by default for your account in the current Region.	Write			
EnableFastSnapshotRestore	Enables fast snapshot restores for the specified snapshots in the specified Availability Zones.	Write	snapshot* (p. 1051)	ec2:Owner (p. 1057) ec2:ParentVolume (p. 1057) ec2:Region (p. 1058) ec2:AvailabilityZone (p. 1056) ec2:SnapshotTime (p. 1058) ec2:Encrypted (p. 1056) ec2:VolumeSize (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058)	
EnableTransitGatewayAttachmentToPropagationRouteTable	Enables the specified attachment to propagate routes to the specified propagation route table.	Write	transit-gateway-attachment* (p. 1053)	ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			transit-gateway-route-table* (p. 1053)	ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)	
EnableVgwRoutePropagation	Enables a virtual private gateway (VGW) to propagate routes to the specified route table of a VPC.	Write			
EnableVolumeIO	Enables I/O operations for a volume that had I/O operations disabled because the data on the volume was potentially inconsistent.	Write			
EnableVpcClassicLink	Enables a VPC for ClassicLink.	Write	vpc* (p. 1054)	ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058) ec2:Tenancy (p. 1058)	
EnableVpcClassicLinkDnsSupport	Enables a VPC to support DNS host name resolution for ClassicLink.	Write			
ExportClientVpnClientConfiguration	Downloads the client certificate revocation list for the specified Client VPN endpoint.	List			
ExportClientVpnConfig	Downloads the contents of the Client VPN endpoint configuration file for the specified Client VPN endpoint.	List			
ExportImage	Exports an Amazon Machine Image (AMI) to a VM file.	Write			
ExportTransitGatewayRoutes	Exports routes from the specified transit gateway route table to the specified S3 bucket.	Write			
GetCapacityReservationUsage	Gets usage information about a Capacity Reservation.	Read			
GetConsoleOutput	Gets the console output for the specified instance.	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetConsoleScreenshot	Retrieve a JPG-format screenshot of a running instance to help with troubleshooting.	Read	instance* (p. 1049)	ec2:AvailabilityZone (p. 1056) ec2:EbsOptimized (p. 1056) ec2:InstanceProfile (p. 1057) ec2:InstanceType (p. 1057) ec2:PlacementGroup (p. 1058) ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058) ec2:RootDeviceType (p. 1058) ec2:Tenancy (p. 1058)	
GetEbsDefaultKmsKeyId	Get EBS Default Kms Key Id	Read			
GetEbsEncryptionByDefault	Describes whether EBS Encryption by default is enabled for your account in the current Region	Read			
GetHostReservations	Preview a reservation purchase with configurations that match those of your Dedicated Host.	Read			
GetLaunchTemplate	Retrieves the configuration data of the specified instance.	Read			
GetPasswordData	Retrieves the encrypted administrator password for an instance running Windows.	Read			
GetReservedInstancesOfferings	Returns details about the values and terms of your specified Convertible Reserved Instances.	Read			
GetTransitGatewayAttachments	Lists the route tables to which the specified resource attachment propagates routes.	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetTransitGatewayAssociationsForRouteTable	Gets information about the associations for the specified transit gateway route table.	List			
GetTransitGatewayRouteTablePropagation	Gets information about the route table propagations for the specified transit gateway route table.	List			
ImportClientVpnClientRevocationList	Uploads a client certificate revocation list to the specified Client VPN endpoint.	Write	client-vpn-endpoint* (p. 1047)	ec2:Region (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058)	
ImportImage	Import single or multi-volume disk images or EBS snapshots into an Amazon Machine Image (AMI).	Write			
ImportInstance	Creates an import instance task using metadata from the specified disk image.	Write			
ImportKeyPair	Imports the public key from an RSA key pair that you created with a third-party tool.	Write			
ImportSnapshot	Imports a disk into an EBS snapshot.	Write			
ImportVolume	Creates an import volume task using metadata from the specified disk image.	Write			
ModifyCapacityReservation	Modifies a Capacity Reservation's capacity and the conditions under which it is to be released.	Write	capacity-reservation* (p. 1047)	ec2:Region (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058)	
ModifyClientVpnEndpoint	Modifies the specified Client VPN endpoint.	Write	client-vpn-endpoint* (p. 1047)	ec2:Region (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058)	
ModifyEbsDefaultKmsMasterKey	Changes the default customer master key (CMK) for EBS encryption by default for your account in this Region	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ModifyFleet	Modifies the specified EC2 Fleet.	Write			
ModifyFpgaImageAttribute	Modifies the specified attribute of the specified Amazon FPGA Image (AFI).	Write			
ModifyHosts	Modify the auto-placement setting of a Dedicated Host.	Write			
ModifyIdFormat	Modifies the ID format for the specified resource on a per-region basis.	Write			
ModifyIdentityIdFormat	Modifies the ID format of a resource for a specified IAM user, IAM role, or the root user for an account; or all IAM users, IAM roles, and the root user for an account.	Write			
ModifyImageAttribute	Modifies the specified attribute of the specified AMI.	Write			
ModifyInstanceAttribute	Modifies the specified attribute of the specified instance.	Write			
ModifyInstanceCapacityReservationSettingsForStoppedInstances	Modifies the Capacity Reservation settings for a stopped instance.	Write			
ModifyInstanceCreditSpecification	Modifies the credit option for CPU usage on an instance.	Write			
ModifyInstanceEventScheduledTime	Modifies the start time for a scheduled EC2 instance event.	Write	instance* (p. 1049)	ec2:Region (p. 1058)	
ModifyInstanceMetadataOptionsForRunningInstances	Modifies the metadata options for an instance.	Write			
ModifyInstancePlacement	Set the instance affinity value for a specific stopped instance and modify the instance tenancy setting.	Write			
ModifyLaunchTemplate	Modifies the specified launch template.	Write	launch-template* (p. 1049)	ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058)	
ModifyNetworkInterfaceAttribute	Modifies the specified network interface attribute. You can specify only one attribute at a time.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ModifyReservedInstances	Modifies the Availability Zone, instance count, instance type, or network platform (EC2-Classic or EC2-VPC) of your Standard Reserved Instances.	Write			
ModifySnapshotPermissions	Adds or removes permission settings for the specified snapshot.	Permissions management	snapshot* (p. 1051)	ec2:Owner (p. 1057) ec2:ParentVolume (p. 1057) ec2:Region (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058) ec2:SnapshotTime (p. 1058) ec2:VolumeSize (p. 1058)	
ModifySpotFleetRequest	Modifies the specified Spot fleet request.	Write			
ModifySubnetAttribute	Modifies a subnet attribute.	Write			
ModifyTrafficMirrorNetworkServices	Allows or restricts mirroring network services.	Write	traffic-mirror-filter* (p. 1052)	ec2:Region (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058)	
ModifyTrafficMirrorRule	Modifies the specified Traffic Mirror rule.	Write	traffic-mirror-filter* (p. 1052)	ec2:Region (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058)	
			traffic-mirror-filter-rule* (p. 1052)	ec2:Region (p. 1058)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ModifyTrafficMirrorSession	Modifies a Traffic Mirror session.	Write	traffic-mirror-session* (p. 1052)	ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)	
	traffic-mirror-filter (p. 1052)		ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)		
	traffic-mirror-target (p. 1052)		ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)		
ModifyTransitGatewayAttachment	Modifies the specified VPC attachment.	Write	transit-gateway-attachment* (p. 1053)	ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)	
	subnet (p. 1052)		ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)		
ModifyVolume	You can modify several parameters of an existing EBS volume, including volume size, volume type, and IOPS capacity.	Write			
ModifyVolumeAttribute	Modifies a volume attribute.	Write			
ModifyVpcAttribute	Modifies the specified attribute of the specified VPC.	Write			
ModifyVpcEndpoint	Modifies attributes of a specified VPC endpoint.	Write			
ModifyVpcEndpointNotificationConfiguration	Modifies a connection notification for a VPC endpoint or VPC endpoint service.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ModifyVpcEndpoint	Modifies the attributes of your VPC endpoint service configuration.	Write			
ModifyVpcEndpoint	Modifies the permissions for your VPC endpoint service.	Permissions management			
ModifyVpcPeeringConnectionOptions	Modifies the VPC peering connection options on one side of a VPC peering connection.	Write			
ModifyVpcTenancy	Modifies the instance tenancy attribute of the specified VPC.	Write			
ModifyVpnConnection	Modifies the target gateway of an AWS Site-to-Site VPN connection	Write	vpn-connection* (p. 1055)	ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058) ec2:GatewayType (p. 1056)	
ModifyVpnTunnel	Modifies the certificate for an AWS Site-to-Site VPN connection.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ModifyVpnTunnelSiteToSite	Modifies the options for an AWS Site-to-Site VPN connection.	Write	vpn-connection* (p. 1055)	ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058) ec2:AuthenticationType (p. 1056) ec2:DPDTimeoutSeconds (p. 1056) ec2:IKEVersions (p. 1056) ec2:InsideTunnelCidr (p. 1057) ec2:Phase1DHGroupNumbers (p. 1057) ec2:Phase2DHGroupNumbers (p. 1057) ec2:Phase1EncryptionAlgorithms (p. 1057) ec2:Phase2EncryptionAlgorithms (p. 1057) ec2:Phase1IntegrityAlgorithms (p. 1057) ec2:Phase2IntegrityAlgorithms (p. 1057) ec2:Phase1LifetimeSeconds (p. 1057) ec2:Phase2LifetimeSeconds (p. 1058) ec2:PresharedKeys (p. 1058) ec2:RekeyFuzzPercentage (p. 1058) ec2:RekeyMarginTimeSeconds (p. 1058) ec2:RoutingType (p. 1058)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
MonitorInstances	Enables detailed monitoring for a running instance.	Write			
MoveAddressToVpc	Moves an Elastic IP address from the EC2-Classic platform to the EC2-VPC platform.	Write			
ProvisionByoipCidr	Provisions an address range for use with your AWS resources through bring your own IP addresses (BYOIP) and creates a corresponding address pool.	Write			
PurchaseHostReservations	Purchase a reservation with configurations that match those of your Dedicated Host.	Write			
PurchaseReservedInstancesOfferings	Purchases a Reserved Instance for use with your account.	Write			
PurchaseScheduledInstances	Purchases one or more Scheduled Instances with the specified schedule.	Write			
RebootInstances	Requests a reboot of one or more instances.	Write	instance* (p. 1049)	ec2:AvailabilityZone (p. 1056) ec2:EbsOptimized (p. 1056) ec2:InstanceProfile (p. 1057) ec2:InstanceType (p. 1057) ec2:PlacementGroup (p. 1058) ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058) ec2:RootDeviceType (p. 1058) ec2:Tenancy (p. 1058)	
RegisterImage	Registers an AMI.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
RejectTransitGatewayAttachment	Rejects a request to attach a VPC to a Transit gateway .	Write	transit-gateway-attachment* (p. 1053)	ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058)	
RejectVpcEndpointConnection	Rejects one or more VPC endpoint connection requests to your VPC endpoint service.	Write			
RejectVpcPeeringConnection	Rejects a VPC peering connection request.	Write	vpc-peering-connection* (p. 1054)	ec2:AcceptorVpc (p. 1056) ec2:Region (p. 1058) ec2:RequesterVpc (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058)	
ReleaseAddress	Releases the specified Elastic IP address.	Write			
ReleaseHosts	When you no longer want to use an On-Demand Dedicated Host it can be released	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ReplaceIamInstanceProfile	Replaces an IAM instance profile for the specified instance.	Write	instance* (p. 1049)	ec2:AvailabilityZone (p. 1056) ec2:EbsOptimized (p. 1056) ec2:InstanceProfile (p. 1057) ec2:InstanceType (p. 1057) ec2:PlacementGroup (p. 1058) ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058) ec2:RootDeviceType (p. 1058) ec2:Tenancy (p. 1058)	iam:PassRole (p. 1056)
ReplaceNetworkAcl	Changes which network ACL a subnet is associated with.	Write			
ReplaceNetworkAclEntry	Replaces an entry (rule) in a network ACL.	Write			
ReplaceRoute	Replaces an existing route within a route table in a VPC.	Write	route-table* (p. 1051)	ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058) ec2:Vpc (p. 1059)	
ReplaceRouteTableAssociation	Changes the route table associated with a given subnet in a VPC.	Write			
ReplaceTransitGatewayRouteTable	Replaces the specified route in the specified transit gateway route table.	Write	transit-gateway-route-table* (p. 1053)	ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			transit-gateway-attachment (p. 1053)	ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)	
ReportInstanceStatus	Submits feedback about the status of an instance	Write			
RequestSpotFleet	Creates a Spot fleet request	Write			
RequestSpotInstances	Creates a Spot instance request	Write			
ResetEbsDefaultKmsMasterKey	Resets the default customer master key (CMK) for EBS encryption for your account in this Region to the AWS managed CMK for EBS	Write			
ResetFpgaImageAttribute	Resets an attribute of an Amazon FPGA Image (AFI) to its default value.	Write			
ResetImageAttribute	Resets an attribute of an AMI to its default value	Write			
ResetInstanceAttribute	Resets an attribute of an instance to its default value	Write			
ResetNetworkInterfaceAttribute	Resets a network interface attribute. You can specify only one attribute at a time.	Write			
ResetSnapshotAttribute	Resets permission settings for the specified snapshot.	Permissions management			
RestoreAddressToClassic	Restores an Elastic IP address that was previously moved to the EC2-VPC platform back to the EC2-Classic platform.	Write			
RevokeClientVpnAuthorization	Removes an ingress authorization rule from a Client VPN endpoint.	Write	client-vpn-endpoint* (p. 1047)	ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
RevokeSecurityGroupIngress	[EC2-VPC only] Removes one ingress rule from a security group for EC2-VPC. This action doesn't apply to security groups for use in EC2-Classic.	Write	security-group* (p. 1051)	ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058) ec2:Vpc (p. 1059)	
RevokeSecurityGroupEgress	Removes one or more egress rules from a security group.	Write	security-group* (p. 1051)	ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058) ec2:Vpc (p. 1059)	
RunInstances	Launches the specified number of instances using an AMI for which you have permissions.	Write	image* (p. 1048)	ec2:ImageType (p. 1057) ec2:IsLaunchTemplateResource (p. 1057) ec2:LaunchTemplate (p. 1057) ec2:Owner (p. 1057) ec2:Public (p. 1058) ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058) ec2:RootDeviceType (p. 1058)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			instance* (p. 1049)	aws:RequestTag/ \${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:AvailabilityZone (p. 1056) ec2:EbsOptimized (p. 1056) ec2:InstanceProfile (p. 1057) ec2:InstanceType (p. 1057) ec2:IsLaunchTemplateResource (p. 1057) ec2:LaunchTemplate (p. 1057) ec2:PlacementGroup (p. 1058) ec2:Region (p. 1058) ec2:RootDeviceType (p. 1058) ec2:Tenancy (p. 1058) ec2:MetadataHttpEndpoint (p. 1057) ec2:MetadataHttpTokens (p. 1057) ec2:MetadataHttpPutResponseHopLimit (p. 1057)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			network-interface* (p. 1050)	ec2:AvailabilityZone (p. 1056) ec2:IsLaunchTemplateResource (p. 1057) ec2:LaunchTemplate (p. 1057) ec2:Region (p. 1058) ec2:ResourceTag/ (p. 1058) ec2:Subnet (p. 1058) ec2:Vpc (p. 1059)	
	security-group* (p. 1051)	ec2:IsLaunchTemplateResource (p. 1057) ec2:LaunchTemplate (p. 1057) ec2:Region (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058) ec2:Vpc (p. 1059)	ec2:IsLaunchTemplateResource (p. 1057) ec2:LaunchTemplate (p. 1057) ec2:Region (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058) ec2:Vpc (p. 1059)		
	subnet* (p. 1052)	ec2:AvailabilityZone (p. 1056) ec2:IsLaunchTemplateResource (p. 1057) ec2:LaunchTemplate (p. 1057) ec2:Region (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058) ec2:Vpc (p. 1059)	ec2:AvailabilityZone (p. 1056) ec2:IsLaunchTemplateResource (p. 1057) ec2:LaunchTemplate (p. 1057) ec2:Region (p. 1058) ec2:ResourceTag/ \${TagKey} (p. 1058) ec2:Vpc (p. 1059)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			volume* (p. 1053) aws:RequestTag/\${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:AvailabilityZone (p. 1056) ec2:Encrypted (p. 1056) ec2:IsLaunchTemplateResource (p. 1057) ec2:LaunchTemplate (p. 1057) ec2:ParentSnapshot (p. 1057) ec2:Region (p. 1058) ec2:VolumeLops (p. 1058) ec2:VolumeSize (p. 1058) ec2:VolumeType (p. 1058)		
			key-pair (p. 1049) ec2:IsLaunchTemplateResource (p. 1057) ec2:LaunchTemplate (p. 1057) ec2:Region (p. 1058)		ec2:IsLaunchTemplateResource (p. 1057) ec2:LaunchTemplate (p. 1057) ec2:Region (p. 1058)
			launch-template (p. 1049) ec2:IsLaunchTemplateResource (p. 1057) ec2:LaunchTemplate (p. 1057) ec2:Region (p. 1058)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			placement-group (p. 1050)	ec2:LaunchTemplate (p. 1057) ec2:PlacementGroupStrategy (p. 1058) ec2:Region (p. 1058)	ec2:IsLaunchTemplateResource (p. 1057) ec2:LaunchTemplate (p. 1057) ec2:PlacementGroupStrategy (p. 1058) ec2:Region (p. 1058)
			snapshot (p. 1051)	ec2:LaunchTemplate (p. 1057) ec2:Owner (p. 1057) ec2:ParentVolume (p. 1057) ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058) ec2:SnapshotTime (p. 1058) ec2:VolumeSize (p. 1058)	ec2:IsLaunchTemplateResource (p. 1057) ec2:LaunchTemplate (p. 1057) ec2:Owner (p. 1057) ec2:ParentVolume (p. 1057) ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058) ec2:SnapshotTime (p. 1058) ec2:VolumeSize (p. 1058)

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
	SCENARIO: EC2-Classic-EBS		image* (p. 1048) instance* (p. 1049) security-group* (p. 1051) volume* (p. 1053) key-pair (p. 1049) placement-group (p. 1050) snapshot (p. 1051)		
	SCENARIO: EC2-Classic-InstanceStore		image* (p. 1048) instance* (p. 1049) security-group* (p. 1051) key-pair (p. 1049) placement-group (p. 1050) snapshot (p. 1051)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
	SCENARIO: EC2-VPC-EBS		image* (p. 1048) instance* (p. 1049) network-interface* (p. 1050) security-group* (p. 1051) volume* (p. 1053) key-pair (p. 1049) placement-group (p. 1050) snapshot (p. 1051)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
	SCENARIO: EC2-VPC-EBS-Subnet		image* (p. 1048) instance* (p. 1049) network-interface* (p. 1050) security-group* (p. 1051) subnet* (p. 1052) volume* (p. 1053) key-pair (p. 1049) placement-group (p. 1050) snapshot (p. 1051)		
	SCENARIO: EC2-VPC-InstanceStore		image* (p. 1048) instance* (p. 1049) network-interface* (p. 1050) security-group* (p. 1051) key-pair (p. 1049) placement-group (p. 1050) snapshot (p. 1051)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
	SCENARIO: EC2-VPC-InstanceStore-Subnet		image* (p. 1048) instance* (p. 1049) network-interface* (p. 1050) security-group* (p. 1051) subnet* (p. 1052) key-pair (p. 1049) placement-group (p. 1050) snapshot (p. 1051)		
RunScheduledInstances	Launches the specified Scheduled Instances.	Write			
SearchTransitGatewayRoutes	Searches for routes in the specified transit gateway route table.	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
SendDiagnosticInterrupt	Sends a diagnostic interrupt to the specified Amazon EC2 instance.	Write	instance* (p. 1049)	ec2:AvailabilityZone (p. 1056) ec2:EbsOptimized (p. 1056) ec2:InstanceProfile (p. 1057) ec2:InstanceType (p. 1057) ec2:PlacementGroup (p. 1058) ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058) ec2:RootDeviceType (p. 1058) ec2:Tenancy (p. 1058)	
StartInstances	Starts an Amazon EBS-backed AMI that you've previously stopped.	Write	instance* (p. 1049)	ec2:AvailabilityZone (p. 1056) ec2:EbsOptimized (p. 1056) ec2:InstanceProfile (p. 1057) ec2:InstanceType (p. 1057) ec2:PlacementGroup (p. 1058) ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058) ec2:RootDeviceType (p. 1058) ec2:Tenancy (p. 1058)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
StopInstances	Stops an Amazon EBS-backed instance.	Write	instance* (p. 1049)	ec2:AvailabilityZone (p. 1056) ec2:EbsOptimized (p. 1056) ec2:InstanceProfile (p. 1057) ec2:InstanceType (p. 1057) ec2:PlacementGroup (p. 1058) ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058) ec2:RootDeviceType (p. 1058) ec2:Tenancy (p. 1058)	
TerminateClientVpnConnections	Terminates active Client VPN endpoint connections.	Write	client-vpn-endpoint* (p. 1047)	ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
TerminateInstances	Shuts down one or more instances.	Write	instance* (p. 1049)	ec2:AvailabilityZone (p. 1056) ec2:EbsOptimized (p. 1056) ec2:InstanceProfile (p. 1057) ec2:InstanceType (p. 1057) ec2:PlacementGroup (p. 1058) ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058) ec2:RootDeviceType (p. 1058) ec2:Tenancy (p. 1058)	
UnassignIpv6Addresses	Unassigns one or more IPv6 addresses from the specified network interface.	Write			
UnassignPrivateIpAddresses	Unassigns one or more secondary private IP addresses from a network interface.	Write			
UnmonitorInstances	Disables detailed monitoring for a running instance.	Write			
UpdateSecurityGroupEgress	[EC2-VPC only] Update egress rules of a security group. This action doesn't apply to security groups for use in EC2-Classic.	Write	security-group* (p. 1051)	ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058) ec2:Vpc (p. 1059)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UpdateSecurityGroupDescription	Update descriptions for one or more ingress rules of a security group.	Write	security-group* (p. 1051)	ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058) ec2:Vpc (p. 1059)	
WithdrawByoipCidr	Stops advertising an IPv4 address range that is provisioned as an address pool.	Write			

Resource Types Defined by Amazon EC2

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 971\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
capacity-reservation	arn:\${Partition}:ec2:\${Region}: \${Account}:capacity-reservation/ \${CapacityReservationId}	aws:RequestTag/ {\$TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058)
client-vpn-endpoint	arn:\${Partition}:ec2:\${Region}: \${Account}:client-vpn-endpoint/ \${ClientVpnEndpointId}	aws:RequestTag/ {\$TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058)
customer-gateway	arn:\${Partition}:ec2:\${Region}: \${Account}:customer-gateway/ \${CustomerGatewayId}	ec2:Region (p. 1058) ec2:ResourceTag/ {\$TagKey} (p. 1058)
dhcp-options	arn:\${Partition}:ec2:\${Region}: \${Account}:dhcp-options/\${DhcpOptionsId}	aws:RequestTag/ {\$TagKey} (p. 1056) aws:TagKeys (p. 1056)

Resource Types	ARN	Condition Keys
		ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)
elastic-gpu	<code>arn:\${Partition}:ec2:\${Region}: \${Account}:elasticGpu/\${ElasticGpuId}</code>	
fpga-image	<code>arn:\${Partition}:ec2:\${Region}::fpga-image/ \${FpgaImageId}</code>	aws:RequestTag/\${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Owner (p. 1057) ec2:Public (p. 1058) ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)
image	<code>arn:\${Partition}:ec2:\${Region}::image/ \${ImageId}</code>	aws:RequestTag/\${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:ImageType (p. 1057) ec2:Owner (p. 1057) ec2:Public (p. 1058) ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058) ec2:RootDeviceType (p. 1058)

Resource Types	ARN	Condition Keys
instance	arn:\${Partition}:ec2:\${Region}: \${Account}:instance/\${InstanceId}	aws:RequestTag/\${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:AvailabilityZone (p. 1056) ec2:EbsOptimized (p. 1056) ec2:InstanceProfile (p. 1057) ec2:InstanceType (p. 1057) ec2:PlacementGroup (p. 1058) ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058) ec2:RootDeviceType (p. 1058) ec2:Tenancy (p. 1058)
internet-gateway	arn:\${Partition}:ec2:\${Region}: \${Account}:internet-gateway/ \${InternetGatewayId}	aws:RequestTag/\${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)
key-pair	arn:\${Partition}:ec2:\${Region}: \${Account}:key-pair/\${KeyName}	ec2:Region (p. 1058)
launch-template	arn:\${Partition}:ec2:\${Region}: \${Account}:launch-template/ \${LaunchTemplateId}	ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)

Resource Types	ARN	Condition Keys
network-acl	<code>arn:\${Partition}:ec2:\${Region}:\${Account}:network-acl/\${NaclId}</code>	aws:RequestTag/\${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058) ec2:Vpc (p. 1059)
network-interface	<code>arn:\${Partition}:ec2:\${Region}:\${Account}:network-interface/\${NetworkInterfaceId}</code>	aws:RequestTag/\${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:AuthorizedService (p. 1056) ec2:AvailabilityZone (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058) ec2:Subnet (p. 1058) ec2:Vpc (p. 1059)
placement-group	<code>arn:\${Partition}:ec2:\${Region}:\${Account}:placement-group/\${PlacementGroupName}</code>	ec2:PlacementGroupStrategy (p. 1058) ec2:Region (p. 1058)
reserved-instances	<code>arn:\${Partition}:ec2:\${Region}:\${Account}:reserved-instances/\${ReservationId}</code>	aws:RequestTag/\${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:AvailabilityZone (p. 1056) ec2:InstanceType (p. 1057) ec2:Region (p. 1058) ec2:ReservedInstancesOfferingType (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058) ec2:Tenancy (p. 1058)

Resource Types	ARN	Condition Keys
route-table	<code>arn:\${Partition}:ec2:\${Region}:\${Account}:route-table/\${RouteTableId}</code>	aws:RequestTag/\${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058) ec2:Vpc (p. 1059)
security-group	<code>arn:\${Partition}:ec2:\${Region}:\${Account}:security-group/\${SecurityGroupId}</code>	aws:RequestTag/\${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058) ec2:Vpc (p. 1059)
snapshot	<code>arn:\${Partition}:ec2:\${Region}::snapshot/\${SnapshotId}</code>	aws:RequestTag/\${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Owner (p. 1057) ec2:ParentVolume (p. 1057) ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058) ec2:SnapshotTime (p. 1058) ec2:VolumeSize (p. 1058)
spot-instance-request	<code>arn:\${Partition}:ec2:\${Region}::spot-instances-request/\${SpotInstanceRequestId}</code>	aws:RequestTag/\${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)

Resource Types	ARN	Condition Keys
subnet	arn:\${Partition}:ec2:\${Region}:\${Account}:subnet/\${SubnetId}	aws:RequestTag/\${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:AvailabilityZone (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058) ec2:Vpc (p. 1059)
traffic-mirror-session	arn:\${Partition}:ec2:\${Region}:\${Account}:traffic-mirror-session/\${TrafficMirrorSessionId}	aws:RequestTag/\${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)
traffic-mirror-target	arn:\${Partition}:ec2:\${Region}:\${Account}:traffic-mirror-target/\${TrafficMirrorTargetId}	aws:RequestTag/\${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)
traffic-mirror-filter	arn:\${Partition}:ec2:\${Region}:\${Account}:traffic-mirror-filter/\${TrafficMirrorFilterId}	aws:RequestTag/\${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)
traffic-mirror-filter-rule	arn:\${Partition}:ec2:\${Region}:\${Account}:traffic-mirror-filter-rule/\${TrafficMirrorFilterRuleId}	ec2:Region (p. 1058)

Resource Types	ARN	Condition Keys
transit-gateway-attachment	arn:\${Partition}:ec2:\${Region}:\${Account}:transit-gateway-attachment/\${TransitGatewayAttachmentId}	aws:RequestTag/\${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)
transit-gateway-route-table	arn:\${Partition}:ec2:\${Region}:\${Account}:transit-gateway-route-table/\${TransitGatewayRouteTableId}	aws:RequestTag/\${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)
transit-gateway	arn:\${Partition}:ec2:\${Region}:\${Account}:transit-gateway/\${TransitGatewayId}	aws:RequestTag/\${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)
volume	arn:\${Partition}:ec2:\${Region}:\${Account}:volume/\${VolumeId}	aws:RequestTag/\${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:AvailabilityZone (p. 1056) ec2:Encrypted (p. 1056) ec2:ParentSnapshot (p. 1057) ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058) ec2:Volumelops (p. 1058) ec2:VolumeSize (p. 1058) ec2:VolumeType (p. 1058)

Resource Types	ARN	Condition Keys
vpc	<code>arn:\${Partition}:ec2:\${Region}:\${Account}:vpc/\${VpcId}</code>	aws:RequestTag/\${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:Region (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058) ec2:Tenancy (p. 1058)
vpc-peering-connection	<code>arn:\${Partition}:ec2:\${Region}:\${Account}:vpc-peering-connection/\${VpcPeeringConnectionId}</code>	ec2:AcceptorVpc (p. 1056) ec2:Region (p. 1058) ec2:RequesterVpc (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)

Resource Types	ARN	Condition Keys
vpn-connection	<code>arn:\${Partition}:ec2:\${Region}:\${Account}:vpn-connection/\${VpnConnectionId}</code>	aws:RequestTag/\${TagKey} (p. 1056) aws:TagKeys (p. 1056) ec2:AuthenticationType (p. 1056) ec2:DPDTimeoutSeconds (p. 1056) ec2:GatewayType (p. 1056) ec2:IKEVersions (p. 1056) ec2:InsideTunnelCidr (p. 1057) ec2:Phase1DHGroupNumbers (p. 1057) ec2:Phase1EncryptionAlgorithms (p. 1057) ec2:Phase1IntegrityAlgorithms (p. 1057) ec2:Phase1LifetimeSeconds (p. 1057) ec2:Phase2DHGroupNumbers (p. 1057) ec2:Phase2EncryptionAlgorithms (p. 1057) ec2:Phase2IntegrityAlgorithms (p. 1057) ec2:Phase2LifetimeSeconds (p. 1058) ec2:PresharedKeys (p. 1058) ec2:Region (p. 1058) ec2:RekeyFuzzPercentage (p. 1058) ec2:RekeyMarginTimeSeconds (p. 1058) ec2:ResourceTag/\${TagKey} (p. 1058)

Resource Types	ARN	Condition Keys
		ec2:RoutingType (p. 1058)
vpn-gateway	<code>arn:\${Partition}:ec2:\${Region}:\${Account}:vpn-gateway/\${VpnGatewayId}</code>	

Condition Keys for Amazon EC2

Amazon EC2 defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	A key that is present in the request the user makes to the EC2 service.	String
aws:TagKeys	The list of all the tag key names associated with the resource in the request.	String
ec2:AcceptorVpc	The ARN of an accepter VPC in a VPC peering connection.	ARN
ec2:AuthenticationType	The authentication type for the VPN tunnel endpoints.	String
ec2:AuthorizedService	The AWS service that has permission to use a resource.	String
ec2:AuthorizedUser	The IAM principal that has permission to use a resource.	String
ec2:AvailabilityZone	The name of an Availability Zone in a region.	String
ec2>CreateAction	The name of a resource-creating API action.	String
ec2:DPDTimeoutSeconds	The duration after which DPD timeout occur.	Numeric
ec2:EbsOptimized	Whether the instance is enabled for EBS-optimization.	Bool
ec2:ElasticGpuType	The name of the type of ElasticGpu.	String
ec2:Encrypted	Whether the volume is encrypted.	Bool
ec2:GatewayType	The gateway type for the VPN endpoint on the AWS side of the VPN connection.	String
ec2:IKEVersions	The internet key exchange (IKE) versions that are permitted for the VPN tunnel.	String

Condition Keys	Description	Type
ec2:ImageType	The name of the type of image.	String
ec2:InsideTunnelCidr	The range of inside IP addresses for the VPN tunnel.	String
ec2:InstanceMarketType	The name of the market type.	String
ec2:InstanceProfile	The ARN of the instance profile.	ARN
ec2:InstanceType	The name of the instance type.	String
ec2:IsLaunchTemplateResource	Launch template resource flag.	Bool
ec2:LaunchTemplate	The ARN of the launch template.	ARN
ec2:MetadataHttpEndpointService	Whether the http endpoint is enabled in instance metadata	String
ec2:MetadataHttpPutServiceHopLimit	Allowed number of hops when calling instance metadata	Numeric
ec2:MetadataHttpTokenService	Whether tokens are required when calling instance metadata	String
ec2:Owner	The name or account ID of the owner.	String
ec2:ParentSnapshot	The ARN of the parent snapshot.	ARN
ec2:ParentVolume	The ARN of the parent volume.	ARN
ec2:Permission	The type of permission for a resource.	String
ec2:Phase1DHGroups	The Diffie-Hellman groups that are permitted for the VPN Tunnel for the phase 1 IKE negotiations.	Numeric
ec2:Phase1EncryptionAlgorithms	The encryption algorithms that are permitted for the VPN Tunnel for the phase 1 IKE negotiations.	String
ec2:Phase1IntegrityAlgorithms	The integrity algorithms that are permitted for the VPN Tunnel for the phase 1 IKE negotiations.	String
ec2:Phase1LifetimeSeconds	The lifetime in seconds for phase 1 of the IKE negotiation.	Numeric
ec2:Phase2DHGroups	The Diffie-Hellman groups that are permitted for the VPN Tunnel for the phase 2 IKE negotiations.	Numeric
ec2:Phase2EncryptionAlgorithms	The encryption algorithms that are permitted for the VPN Tunnel for the phase 2 IKE negotiations.	String
ec2:Phase2IntegrityAlgorithms	The integrity algorithms that are permitted for the VPN Tunnel for the phase 2 IKE negotiations.	String

Condition Keys	Description	Type
ec2:Phase2LifetimeSeconds	The lifetime in seconds for phase 2 of the IKE negotiation.	Numeric
ec2:PlacementGroup	The ARN of the placement group.	ARN
ec2:PlacementGroupStrategy	The name of the placement group strategy.	String
ec2:PreSharedKeys	The pre-shared key (PSK) to establish the initial IKE security association between the virtual private gateway and customer gateway.	String
ec2:ProductCode	The product code of the product.	String
ec2:Public	Whether the image is public.	Bool
ec2:Region	The name of the region.	String
ec2:RekeyFuzzPercentage	The percentage of the rekey window (determined by the rekey margin time) within which the rekey time is randomly selected.	Numeric
ec2:RekeyMarginTime	The margin time before the phase 2 lifetime expires, during which AWS performs an IKE rekey.	Numeric
ec2:RequesterVpc	The ARN of a requester VPC in a VPC peering connection.	ARN
ec2:ReservedInstancesOfferingType	The payment option for a Reserved Instance.	String
ec2:ResourceTag/ \${TagKey}	The preface string for a tag key and value pair attached to a resource.	String
ec2:ResourceTag/ \${TagKey}	A tag key and value pair.	String
ec2:RoleDelivery	The IMDS version of IAM role credential for EC2.	Numeric
ec2:RootDeviceType	The root device type: ebs or instance-store.	String
ec2:RoutingType	The routing type for the VPN connection.	String
ec2:SnapshotTime	The snapshot creation time.	String
ec2:SourceInstanceARN	The ARN of the instance from which the request originated.	ARN
ec2:Subnet	The ARN of the subnet.	ARN
ec2:Tenancy	The tenancy of the instance or VPC.	String
ec2:Volumelops	The number of input/output operations per second.	Numeric
ec2:VolumeSize	The size of the volume, in GiB.	Numeric
ec2:VolumeType	The name of the type of volume.	String

Condition Keys	Description	Type
ec2:Vpc	The ARN of the VPC.	ARN

Actions, Resources, and Condition Keys for Amazon EC2 Auto Scaling

Amazon EC2 Auto Scaling (service prefix: `autoscaling`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon EC2 Auto Scaling \(p. 1059\)](#)
- [Resource Types Defined by Amazon EC2 Auto Scaling \(p. 1067\)](#)
- [Condition Keys for Amazon EC2 Auto Scaling \(p. 1068\)](#)

Actions Defined by Amazon EC2 Auto Scaling

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AttachInstances	Attaches one or more EC2 instances to the specified Auto Scaling group.	Write	autoScaling (p. 1068)	autoScaling:\${TagKey} (p. 1068) aws:ResourceTag/\${TagKey} (p. 1069)	ResourceTag/
AttachLoadBalancerTargetGroupsToAutoScalingGroup	Attaches one or more target groups to the specified Auto Scaling group.	Write	autoScaling (p. 1068)	autoScaling:\${TagKey} (p. 1068)	ResourceTag/

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:ResourceTag/ \${TagKey} (p. 1069)	
	autoscaling:TargetGroupARNs (p. 1068)				
AttachLoadBalancerToAutoScalingGroup	Attaches one or more load balancers to the specified Auto Scaling group.	Write	autoScaling:GroupARNs (p. 1068)	aws:ResourceTag/ \${TagKey} (p. 1068)	
	aws:ResourceTag/ \${TagKey} (p. 1069)				
	autoscaling:LoadBalancerNames (p. 1068)				
BatchDeleteScheduledScalingActions	Deletes the specified scheduled scaling actions.	Write	autoScaling:GroupARNs (p. 1068)	aws:ResourceTag/ \${TagKey} (p. 1068)	
BatchPutScheduledScalingActions	Creates or updates multiple scheduled scaling actions for an Auto Scaling group.	Write	autoScaling:GroupARNs (p. 1068)	aws:ResourceTag/ \${TagKey} (p. 1068)	
CompleteLifecycleTransition	Completes the lifecycle action for the specified token or instance with the specified result.	Write	autoScaling:GroupARNs (p. 1068)	aws:ResourceTag/ \${TagKey} (p. 1068)	
CreateAutoScalingGroup	Creates an Auto Scaling group with the specified name and attributes.	Tagging	autoScaling:GroupARNs (p. 1068)	aws:ResourceTag/ \${TagKey} (p. 1068)	
				aws:ResourceTag/ \${TagKey} (p. 1069)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
					autoscaling:InstanceTypes (p. 1068) autoscaling:LaunchConfigurationName (p. 1068) autoscaling:LoadBalancerNames (p. 1068) autoscaling:MaxSize (p. 1068) autoscaling:MinSize (p. 1068) autoscaling:TargetGroupARNs (p. 1068) autoscaling:VPCZoneIdentifiers (p. 1069) aws:RequestTag/\${TagKey} (p. 1069) aws:TagKeys (p. 1069)
CreateLaunchConfiguration	Creates a launch configuration.	Write	launchConfiguration*		
				autoscaling:ImageId (p. 1068) autoscaling:InstanceType (p. 1068) autoscaling:SpotPrice (p. 1068)	
CreateOrUpdateTags	Creates or updates tags for the specified Auto Scaling group.	Tagging	autoScalingGroup*	autoScaling:ResourceTag/\${TagKey} (p. 1068) aws:ResourceTag/\${TagKey} (p. 1069)	
				aws:RequestTag/\${TagKey} (p. 1069) aws:TagKeys (p. 1069)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteAutoScalingGroup	Deletes the specified Auto Scaling group.	Write	autoScaling (p. 1068)	autoScaling:ResourceTag/\${TagKey} (p. 1068) aws:ResourceTag/\${TagKey} (p. 1069)	
DeleteLaunchConfiguration	Deletes the specified launch configuration.	Write	launchConfiguration* (p. 1068)		
DeleteLifecycleHook	Deletes the specified lifecycle hook.	Write	autoScaling (p. 1068)	autoScaling:ResourceTag/\${TagKey} (p. 1068) aws:ResourceTag/\${TagKey} (p. 1069)	
DeleteNotification	Deletes the specified notification.	Write	autoScaling (p. 1068)	autoScaling:ResourceTag/\${TagKey} (p. 1068) aws:ResourceTag/\${TagKey} (p. 1069)	
DeletePolicy	Deletes the specified Auto Scaling policy.	Write	autoScaling (p. 1068)	autoScaling:ResourceTag/\${TagKey} (p. 1068) aws:ResourceTag/\${TagKey} (p. 1069)	
DeleteScheduledAction	Deletes the specified scheduled action.	Write	autoScaling (p. 1068)	autoScaling:ResourceTag/\${TagKey} (p. 1068) aws:ResourceTag/\${TagKey} (p. 1069)	
DeleteTags	Deletes the specified tags.	Tagging	autoScaling (p. 1068)	autoScaling:ResourceTag/\${TagKey} (p. 1068) aws:ResourceTag/\${TagKey} (p. 1069)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:RequestTag/ \${TagKey} (p. 1069) aws:TagKeys (p. 1069)	
DescribeAccountScaling	Describes the current Auto Scaling resource limits for your AWS account.	List			
DescribeAdjustmentTypes	Describes the policy adjustment types for use with PutScalingPolicy.	List			
DescribeAutoScalingGroups	Describes one or more Auto Scaling groups. If a list of names is not provided, the call describes all Auto Scaling groups.	List			
DescribeAutoScalingInstances	Describes one or more Auto Scaling instances. If a list is not provided, the call describes all instances.	List			
DescribeAutoScalingNotificationTypes	Describes the notification types that are supported by Auto Scaling.	List			
DescribeLaunchConfigurations	Describes one or more launch configurations. If you omit the list of names, then the call describes all launch configurations.	List			
DescribeLifecycleHooks	Describes the available types of lifecycle hooks.	List			
DescribeLifecycleHooksForAutoScaling	Describes the lifecycle hooks for the specified Auto Scaling group.	List			
DescribeLoadBalancers	Describes the target groups for the specified Auto Scaling group.	List			
DescribeLoadBalancersForAutoScaling	Describes the load balancers for the specified Auto Scaling group.	List			
DescribeMetricCollectionWatches	Describes the available CloudWatch metrics for Auto Scaling.	List			
DescribeNotifications	Describes the notification actions associated with the specified Auto Scaling group.	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribePolicies	Describes the policies for the specified Auto Scaling group.	List			
DescribeScalingActivities	Describes one or more scaling activities for the specified Auto Scaling group.	List			
DescribeScalingProcessTypes	Describes the scaling process types for use with ResumeProcesses and SuspendProcesses.	List			
DescribeScheduledActions	Describes the actions scheduled for your Auto Scaling group that haven't run.	List			
DescribeTags	Describes the specified tags.	Read			
DescribeTerminationPolicies	Describes the termination policies supported by Auto Scaling.	List			
DetachInstances	Removes one or more instances from the specified Auto Scaling group.	Write	autoScalingGroups (p. 1068) aws:ResourceTag/\${TagKey} (p. 1068)	autoScaling:ResourceTag/ \${TagKey} (p. 1069)	
DetachLoadBalancers	Detaches one or more target groups from the specified Auto Scaling group.	Write	autoScalingGroups (p. 1068) aws:ResourceTag/\${TagKey} (p. 1068)	autoScaling:ResourceTag/ \${TagKey} (p. 1068)	
					autoscaling:TargetGroupARNs (p. 1068)
DetachLoadBalancers	Removes one or more load balancers from the specified Auto Scaling group.	Write	autoScalingGroups (p. 1068) aws:ResourceTag/\${TagKey} (p. 1068)	autoScaling:ResourceTag/ \${TagKey} (p. 1068)	
					autoscaling:LoadBalancerNames (p. 1068)

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DisableMetricsCollection	Disables monitoring of the specified metrics for the specified Auto Scaling group.	Write	autoScaling (p. 1068)	autoScaling:ResourceTag/\${TagKey} (p. 1068) aws:ResourceTag/\${TagKey} (p. 1069)	
EnableMetricsCollection	Enables monitoring of the specified metrics for the specified Auto Scaling group.	Write	autoScaling (p. 1068)	autoScaling:ResourceTag/\${TagKey} (p. 1068) aws:ResourceTag/\${TagKey} (p. 1069)	
EnterStandby	Moves the specified instances into Standby mode.	Write	autoScaling (p. 1068)	autoScaling:ResourceTag/\${TagKey} (p. 1068) aws:ResourceTag/\${TagKey} (p. 1069)	
ExecutePolicy	Executes the specified policy.	Write	autoScaling (p. 1068)	autoScaling:ResourceTag/\${TagKey} (p. 1068) aws:ResourceTag/\${TagKey} (p. 1069)	
ExitStandby	Moves the specified instances out of Standby mode.	Write	autoScaling (p. 1068)	autoScaling:ResourceTag/\${TagKey} (p. 1068) aws:ResourceTag/\${TagKey} (p. 1069)	
PutLifecycleHook	Creates or updates a lifecycle hook for the specified Auto Scaling Group.	Write	autoScaling (p. 1068)	autoScaling:ResourceTag/\${TagKey} (p. 1068) aws:ResourceTag/\${TagKey} (p. 1069)	
PutNotificationConfigurations	Configures an Auto Scaling group to send notifications when specified events take place.	Write	autoScaling (p. 1068)	autoScaling:ResourceTag/\${TagKey} (p. 1068) aws:ResourceTag/\${TagKey} (p. 1069)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
PutScalingPolicy	Creates or updates a policy for an Auto Scaling group.	Write	autoScaling (p. 1068) autoScaling:ResourceTag/ \${TagKey} (p. 1068) aws:ResourceTag/ \${TagKey} (p. 1069)		
PutScheduledUpdateGroupAction	Creates or updates a scheduled scaling option for an Auto Scaling group.	Write	autoScaling (p. 1068) autoScaling:ResourceTag/ \${TagKey} (p. 1068) aws:ResourceTag/ \${TagKey} (p. 1069)		autoscaling:MaxSize (p. 1068) autoscaling:MinSize (p. 1068)
RecordLifecycleActionHeartbeat	Records a heartbeat for the lifecycle action associated with the specified token or instance.	Write	autoScaling (p. 1068) autoScaling:ResourceTag/ \${TagKey} (p. 1068) aws:ResourceTag/ \${TagKey} (p. 1069)		
ResumeProcesses	Resumes the specified suspended Auto Scaling processes, or all suspended process, for the specified Auto Scaling group.	Write	autoScaling (p. 1068) autoScaling:ResourceTag/ \${TagKey} (p. 1068) aws:ResourceTag/ \${TagKey} (p. 1069)		
SetDesiredCapacity	Sets the size of the specified Auto Scaling group.	Write	autoScaling (p. 1068) autoScaling:ResourceTag/ \${TagKey} (p. 1068) aws:ResourceTag/ \${TagKey} (p. 1069)		
SetInstanceHealth	Sets the health status of the specified instance.	Write	autoScaling (p. 1068) autoScaling:ResourceTag/ \${TagKey} (p. 1068) aws:ResourceTag/ \${TagKey} (p. 1069)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
SetInstanceProtection	Updates the instance protection settings of the specified instances.	Write	autoScaling (p. 1068) aws:ResourceTag/\${TagKey} (p. 1069)	autoScaling:ResourceTag/\${TagKey} (p. 1068) aws:ResourceTag/\${TagKey} (p. 1069)	
SuspendProcesses	Suspends the specified Auto Scaling processes, or all processes, for the specified Auto Scaling group.	Write	autoScaling (p. 1068) aws:ResourceTag/\${TagKey} (p. 1069)	autoScaling:ResourceTag/\${TagKey} (p. 1068) aws:ResourceTag/\${TagKey} (p. 1069)	
TerminateInstances	Terminates the specified instances and optionally adjusts the desired group size.	Write	autoScaling (p. 1068) aws:ResourceTag/\${TagKey} (p. 1069)	autoScaling:ResourceTag/\${TagKey} (p. 1068) aws:ResourceTag/\${TagKey} (p. 1069)	
UpdateAutoScalingGroup	Updates the configuration for the specified Auto Scaling group.	Write	autoScaling (p. 1068) aws:ResourceTag/\${TagKey} (p. 1069)	autoScaling:ResourceTag/\${TagKey} (p. 1068) aws:ResourceTag/\${TagKey} (p. 1069)	
				autoscaling:InstanceTypes (p. 1068) autoscaling:LaunchConfigurationName (p. 1068) autoscaling:MaxSize (p. 1068) autoscaling:MinSize (p. 1068) autoscaling:VPCZoneIdentifiers (p. 1069)	

Resource Types Defined by Amazon EC2 Auto Scaling

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 1059\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
autoScalingGroup	arn:\${Partition}:autoscaling:\${Region}:\${Account}:autoScalingGroup:\${GroupId}:autoScalingGroupName/\${GroupFriendlyName}	autoscaling:ResourceTag/\${TagKey} (p. 1068) aws:ResourceTag/\${TagKey} (p. 1069)
launchConfiguration	arn:\${Partition}:autoscaling:\${Region}:\${Account}:launchConfiguration:\${Id}:launchConfigurationName/\${LaunchConfigurationName}	

Condition Keys for Amazon EC2 Auto Scaling

Amazon EC2 Auto Scaling defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
autoscaling:ImageId	The AMI used to create the instance.	String
autoscaling:InstanceType	The type of instance, in terms of the hardware resources available.	String
autoscaling:InstanceTypes	The types of instances, in terms of the hardware resources available.	String
autoscaling:LaunchConfigurationName	The name of a launch configuration.	String
autoscaling:LoadBalancerNames	The name of the load balancer.	String
autoscaling:MaxSize	The maximum scaling size.	Numeric
autoscaling:MinSize	The minimum scaling size.	Numeric
autoscaling:ResourceTag/\${TagKey}	The value of a tag attached to a resource.	String
autoscaling:SpotPrice	The spot price associated with an instance.	Numeric
autoscaling:TargetGroupARNs	The ARN of a target group.	ARN

Condition Keys	Description	Type
<code>autoscaling:VPCZoneIdentifiers</code>	The identifier of a VPC zone.	String
<code>aws:RequestTag/\${TagKey}</code>	The value of a tag associated with the request.	String
<code>aws:ResourceTag/\${TagKey}</code>	Filters actions based on tag-value associated with the resource.	String
<code>aws:TagKeys</code>	Filters create requests based on the presence of mandatory tags in the request.	String

Actions, Resources, and Condition Keys for Amazon EC2 Image Builder

Amazon EC2 Image Builder (service prefix: `imagebuilder`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon EC2 Image Builder \(p. 1069\)](#)
- [Resource Types Defined by Amazon EC2 Image Builder \(p. 1074\)](#)
- [Condition Keys for Amazon EC2 Image Builder \(p. 1075\)](#)

Actions Defined by Amazon EC2 Image Builder

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CancelImageCreation	Cancel an image creation	Write	image* (p. 1074)		
CreateComponent	Create a new component	Write	component* (p. 1074)		
			kmsKey (p. 1075)		
				aws:RequestTag/ \${TagKey} (p. 1075)	
				aws:TagKeys (p. 1075)	
CreateDistributionConfiguration	Create a new distribution configuration	Write	distributionConfiguration* (p. 1074)		
				aws:RequestTag/ \${TagKey} (p. 1075)	
				aws:TagKeys (p. 1075)	
CreateImage	Create a new image	Write	image* (p. 1074)		imagebuilder:GetImageRe imagebuilder:GetInfrastru
				aws:RequestTag/ \${TagKey} (p. 1075)	
				aws:TagKeys (p. 1075)	
CreateImagePipeline	Create a new image pipeline	Write	imagePipeline* (p. 1075)		imagebuilder:GetImageR
				aws:RequestTag/ \${TagKey} (p. 1075)	
				aws:TagKeys (p. 1075)	
CreateImageRecipe	Create a new Image Recipe	Write	imageRecipe* (p. 1075)		imagebuilder:GetCompon
				aws:RequestTag/ \${TagKey} (p. 1075)	
				aws:TagKeys (p. 1075)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateInfrastructureConfiguration	Create a new infrastructure configuration	Write	infrastructureConfiguration* (p. 1075)	iam:PassRole	
			aws:RequestTag/ \${TagKey} (p. 1075)	aws:TagKeys (p. 1075)	
DeleteComponent	Delete a component	Write	component* (p. 1074)		
DeleteDistributionConfiguration	Delete a distribution configuration	Write	distributionConfiguration* (p. 1074)		
DeleteImage	Delete an image	Write	image* (p. 1074)		
DeleteImagePipeline	Delete an image pipeline	Write	imagePipeline* (p. 1075)		
DeleteImageRecipe	Grants permission to delete an image recipe	Write	imageRecipe* (p. 1075)		
DeleteInfrastructureConfiguration	Delete an infrastructure configuration	Write	infrastructureConfiguration* (p. 1075)		
GetComponent	View details about a component	Read	component* (p. 1074)		
GetComponentPolicy	View the resource policy associated with a component	Permissions management	component* (p. 1074)		
GetDistributionConfiguration	View details about a distribution configuration	Read	distributionConfiguration* (p. 1074)		
GetImage	View details about an image	Read	image* (p. 1074)		
	aws:ResourceTag/ \${TagKey} (p. 1075)				
GetImagePipeline	View details about an image pipeline	Read	imagePipeline* (p. 1075)		
GetImagePolicy	View the resource policy associated with an image	Permissions management	image* (p. 1074)		
GetImageRecipe	View details about an image recipe	Read	imageRecipe* (p. 1075)		
GetImageRecipePolicy	View the resource policy associated with an image recipe	Permissions management	imageRecipe* (p. 1075)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetInfrastructureConfigurations	View details about an Infrastructure configuration	Read	infrastructureConfiguration* (p. 1075)		
ListComponentBuildVersions	List the component build versions in your account	List	componentVersion* (p. 1074)		
ListComponents	List the component versions owned by or shared with your account	List			
ListDistributionConfigurations	List the distribution configurations in your account	List			
ListImageBuildVersions	List the image build versions in your account	List	imageVersion* (p. 1074)		
ListImagePipelines	List the image pipelines in your account	List			
ListImageRecipes	List the image recipes owned by or shared with your account	List			
ListImages	List the image versions owned by or shared with your account	List			
ListInfrastructureConfigurations	List the infrastructure configurations in your account	List			
ListTagsForResource	List tag for an Image Builder resource	Read	component (p. 1074)		
			distributionConfiguration (p. 1074)		
			image (p. 1074)		
			imagePipeline (p. 1075)		
			imageRecipe (p. 1075)		
			infrastructureConfiguration (p. 1075)		
			aws:ResourceTag/ \${TagKey} (p. 1075)		
PutComponentPolicy	Set the resource policy associated with a component	Permissions management	component* (p. 1074)		
PutImagePolicy	Set the resource policy associated with an image	Permissions management	image* (p. 1074)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
PutImageRecipePolicy	Set the resource policy associated with an image recipe	Permissions management	imageRecipe* (p. 1075)		
StartImagePipeline	Create a new image from a pipeline	Write	imagePipeline* (p. 1075)		imagebuilder:GetImagePipeline
TagResource	Tag an Image Builder resource	Tagging	component (p. 1074)		
distributionConfiguration (p. 1074)					
image (p. 1074)					
imagePipeline (p. 1075)					
imageRecipe (p. 1075)					
infrastructureConfiguration (p. 1075)					
	aws:TagKeys (p. 1075)				
UntagResource	Untag an Image Builder resource	Tagging	component (p. 1074)		
			distributionConfiguration (p. 1074)		
			image (p. 1074)		
			imagePipeline (p. 1075)		
			imageRecipe (p. 1075)		
			infrastructureConfiguration (p. 1075)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:ResourceTag/ \${TagKey} (p. 1075) aws:TagKeys (p. 1075)	
UpdateDistributionConfiguration	Update an existing distribution configuration	Write	distributionConfiguration* (p. 1074)		
UpdateImagePipeline	Update an existing image pipeline	Write	imagePipeline* (p. 1075)		
UpdateInfrastructureConfiguration	Update an existing infrastructure configuration	Write	infrastructureConfiguration* (p. 1075)	iam:PassRole	
				aws:ResourceTag/ \${TagKey} (p. 1075)	

Resource Types Defined by Amazon EC2 Image Builder

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1069\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
component	arn:\${Partition}:imagebuilder:\${Region}: \${Account}:component/\${ComponentName}/ \${ComponentVersion}/\${ComponentBuildVersion}	aws:ResourceTag/ \${TagKey} (p. 1075)
componentVersion	arn:\${Partition}:imagebuilder:\${Region}: \${Account}:component/\${ComponentName}/ \${ComponentVersion}	aws:ResourceTag/ \${TagKey} (p. 1075)
distributionConfiguration	arn:\${Partition}:imagebuilder:\${Region}: \${Account}:distribution-configuration/ \${DistributionConfigurationName}	aws:ResourceTag/ \${TagKey} (p. 1075)
image	arn:\${Partition}:imagebuilder: \${Region}: \${Account}:image/\${ImageName}/ \${ImageVersion}/\${ImageBuildVersion}	aws:ResourceTag/ \${TagKey} (p. 1075)
imageVersion	arn:\${Partition}:imagebuilder: \${Region}: \${Account}:image/\${ImageName}/ \${ImageVersion}	aws:ResourceTag/ \${TagKey} (p. 1075)

Resource Types	ARN	Condition Keys
imageRecipe	arn:\${Partition}:imagebuilder:\${Region}: \${Account}:image-recipe/\${ImageRecipeName}/ \${ImageRecipeVersion}	aws:ResourceTag/\${TagKey} (p. 1075)
imagePipeline	arn:\${Partition}:imagebuilder: \${Region}: \${Account}:image-pipeline/ \${ImagePipelineName}	aws:ResourceTag/\${TagKey} (p. 1075)
infrastructureConfiguration	arn:\${Partition}:imagebuilder:\${Region}: \${Account}:infrastructure-configuration/ \${ResourceId}	aws:ResourceTag/\${TagKey} (p. 1075)
kmsKey	arn:\${Partition}:kms:\${Region}: \${Account}:key/\${KeyId}	

Condition Keys for Amazon EC2 Image Builder

Amazon EC2 Image Builder defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Filters actions by the presence of tag key-value pairs in the request	String
aws:ResourceTag/\${TagKey}	Filters actions by tag key-value pairs attached to the resource	String
aws:TagKeys	Filters actions by the presence of tag keys in the request	String

Actions, Resources, and Condition Keys for Amazon EC2 Instance Connect

Amazon EC2 Instance Connect (service prefix: `ec2-instance-connect`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon EC2 Instance Connect \(p. 1076\)](#)

- [Resource Types Defined by Amazon EC2 Instance Connect \(p. 1076\)](#)
- [Condition Keys for Amazon EC2 Instance Connect \(p. 1076\)](#)

Actions Defined by Amazon EC2 Instance Connect

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
SendSSHPublicKey	Grants permission to push the SSH public key to the instance metadata where it remains for 60 seconds.	Write	instance* (p. 1076)		
				ec2:osuser (p. 1077)	

Resource Types Defined by Amazon EC2 Instance Connect

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1076\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
instance	<code>arn:\${Partition}:ec2:\${Region}:\${Account}:instance/\${InstanceId}</code>	

Condition Keys for Amazon EC2 Instance Connect

Amazon EC2 Instance Connect defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
<code>ec2:osuser</code>	Filters access by specifying the default user name for the AMI that you used to launch your instance	String

Actions, Resources, and Condition Keys for AWS Elastic Beanstalk

AWS Elastic Beanstalk (service prefix: `elasticbeanstalk`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Elastic Beanstalk \(p. 1077\)](#)
- [Resource Types Defined by AWS Elastic Beanstalk \(p. 1084\)](#)
- [Condition Keys for AWS Elastic Beanstalk \(p. 1084\)](#)

Actions Defined by AWS Elastic Beanstalk

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AbortEnvironmentUpdate	Grants permission to cancel in progress environment configuration update or application version deployment.	Write	environment (*elasticbeanstalk: InApplication (p. 1084))	(p. 1085)	
AddTags	Grants permission to add tags to an Elastic Beanstalk resource and to update tag values.	Tagging	application (p. 1084)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			applicationversion (p. 1084)		
			configurationtemplate (p. 1084)		
			environment (p. 1084)		
			platform (p. 1084)		
				aws:RequestTag/ \${TagKey} (p. 1085)	
					aws:TagKeys (p. 1085)
ApplyEnvironmentManagedAction	Grants permission to apply a scheduled managed action immediately.	Write	environment (p. 1084)	elasticbeanstalk:InApplication (p. 1085)	
CheckDNSAvailability	Grants permission to check CNAME availability.	Read			
ComposeEnvironmentUpdate	Grants permission to create or update a group of environments, each running a separate component of a single application.	Write	application* (p. 1084)		
			applicationversion (p. 1084)	elasticbeanstalk:InApplication (p. 1085)	
CreateApplication	Grants permission to create a new application.	Write	application* (p. 1084)		
CreateApplicationVersion	Grants permission to create an application version for an application.	Write	application* (p. 1084)		
			applicationversion (p. 1084)	elasticbeanstalk:InApplication (p. 1085)	
				aws:RequestTag/ \${TagKey} (p. 1085)	aws:TagKeys (p. 1085)

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateConfigurationTemplate	Grants permission to create a configuration template.	Write	configurationTemplate* (p. 1084)	elasticbeanstalk:InApplication (p. 1085)	
CreateEnvironment	Grants permission to launch an environment for an application.	Write	environment* (p. 1084)	elasticbeanstalk:InApplication (p. 1085)	
CreatePlatformVersion	Grants permission to create a new version of a custom platform.	Write	platform* (p. 1084)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:RequestTag/ \${TagKey} (p. 1085) aws:TagKeys (p. 1085)	
CreateStorageLocation	Grants permission to create the Amazon S3 storage location for the account.	Write			
DeleteApplication	Grants permission to delete an application along with all associated versions and configurations.	Write	application*	(p. 1084)	
DeleteApplicationVersion	Grants permission to delete an application version from an application.	Write	applicationversion*	elasticbeanstalk:InApplication (p. 1084)	
DeleteConfigurationTemplate	Grants permission to delete a configuration template.	Write	configurationtemplate*	elasticbeanstalk:InApplication (p. 1084)	
DeleteEnvironmentConfiguration	Grants permission to delete the configuration associated with the running environment.	Write	environment*	elasticbeanstalk:InApplication (p. 1084)	
DeletePlatformVersion	Grants permission to delete a version of a custom platform.	Write	platform*	(p. 1084)	
DescribeAccountAttributes	Grants permission to retrieve a list of account attributes, including resource quotas.	Read			
DescribeApplicationVersions	Grants permission to retrieve a list of application versions stored in an AWS Elastic Beanstalk storage bucket.	List	applicationversion*	elasticbeanstalk:InApplication (p. 1084)	
DescribeApplications	Grants permission to retrieve the descriptions of existing applications.	List	application	(p. 1084)	
DescribeConfigurationOptions	Grants permission to retrieve descriptions of environment configuration options.	Read	configurationoptions*	elasticbeanstalk:InApplication (p. 1084)	
DescribeConfigurationSetOptions	Grants permission to retrieve a description of the settings for a configuration set.	Read	configurationsetoptions*	elasticbeanstalk:InApplication (p. 1084)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			environment (p. 1084)	elasticbeanstalk:InApplication (p. 1085)	
DescribeEnvironmentHealth	Grants permission to retrieve information about the overall health of an environment.	Read	environment (p. 1084)		
DescribeEnvironmentManagedActions	Grants permission to retrieve a list of managed environments' completed and failed managed actions.	Read	environment (p. 1084)	elasticbeanstalk:InApplication (p. 1085)	
DescribeEnvironmentUpcomingActions	Grants permission to retrieve a list of an environment's upcoming and in-progress managed actions.	Read	environment (p. 1084)	elasticbeanstalk:InApplication (p. 1085)	
DescribeEnvironmentResources	Grants permission to retrieve a list of AWS resources for an environment.	Read	environment (p. 1084)	elasticbeanstalk:InApplication (p. 1085)	
DescribeEnvironmentDescriptions	Grants permission to retrieve descriptions for existing environments.	List	environment (p. 1084)	elasticbeanstalk:InApplication (p. 1085)	
DescribeEvents	Grants permission to retrieve a list of event descriptions matching a set of criteria.	Read	application (p. 1084)		
			applicationversion (p. 1084)	elasticbeanstalk:InApplication (p. 1085)	
			configuration (p. 1084)	elasticbeanstalk:InApplication (p. 1085)	
			environment (p. 1084)	elasticbeanstalk:InApplication (p. 1085)	
DescribeInstancesHealth	Grants permission to retrieve more detailed information about the health of environment instances.	Read	environment (p. 1084)		
DescribePlatformVersion	Grants permission to retrieve a description of a platform version.	Read	platform (p. 1084)		
ListAvailableSolutionStacks	Grants permission to retrieve a list of the available solution stack names.	List	solutionstack (p. 1084)		
ListPlatformVersions	Grants permission to retrieve a list of the available platforms.	List	platform (p. 1084)		
ListTagsForResource	Grants permission to retrieve a list of tags of an Elastic Beanstalk resource.	Read	application (p. 1084)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			applicationversion (p. 1084)		
			configurationtemplate (p. 1084)		
			environment (p. 1084)		
			platform (p. 1084)		
RebuildEnvironment	Grants permission to delete and recreate all of the AWS resources for an environment and to force a restart.	Write	environment (p. 1084)	elasticbeanstalk:InApplication (p. 1085)	
RemoveTags	Grants permission to remove tags from an Elastic Beanstalk resource.	Tagging	application (p. 1084)		
	applicationversion (p. 1084)				
	configurationtemplate (p. 1084)				
	environment (p. 1084)				
	platform (p. 1084)				
			aws:TagKeys (p. 1085)		
RequestEnvironmentInfo	Grants permission to initiate a request to compile information of the deployed environment.	Read	environment (p. 1084)	elasticbeanstalk:InApplication (p. 1085)	
RestartAppServer	Grants permission to request an environment to restart the application container server running on each Amazon EC2 instance.	Write	environment (p. 1084)	elasticbeanstalk:InApplication (p. 1085)	
RetrieveEnvironmentInfo	Grants permission to retrieve the compiled information from a RequestEnvironmentInfo request.	Read	environment (p. 1084)	elasticbeanstalk:InApplication (p. 1085)	
SwapEnvironmentCNAMES	Grants permission to swap the CNAMES of two environments.		environment (p. 1084)	elasticbeanstalk:InApplication (p. 1085)	
				elasticbeanstalk:FromEnvironment (p. 1085)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
TerminateEnvironment	Grants permission to terminate an environment.	Write	environment (p. 1084)	elasticbeanstalk:InApplication (p. 1085)	
UpdateApplication	Grants permission to update an application with specified properties.	Write	application* (p. 1084)		
UpdateApplicationLifecyclePolicy	Grants permission to update the application lifecycle policy associated with the application.	Write	application* (p. 1084)		
UpdateApplicationVersion	Grants permission to update an application version with specified properties.	Write	applicationversion (p. 1084)	elasticbeanstalk:InApplication (p. 1085)	
UpdateConfigurationTemplate	Grants permission to update a configuration template with specified properties or configuration option values.	Write	configurationtemplate (p. 1084)	elasticbeanstalk:InApplication (p. 1085)	
elasticbeanstalk:FromApplication (p. 1085)					
elasticbeanstalk:FromApplicationVersion (p. 1085)					
elasticbeanstalk:FromConfigurationTemplate (p. 1085)					
elasticbeanstalk:FromEnvironment (p. 1085)					
elasticbeanstalk:FromSolutionStack (p. 1085)					
UpdateEnvironment	Grants permission to update an environment.	Write	environment (p. 1084)	elasticbeanstalk:InApplication (p. 1085)	
elasticbeanstalk:FromApplicationVersion (p. 1085)					
elasticbeanstalk:FromConfigurationTemplate (p. 1085)					
elasticbeanstalk:FromSolutionStack (p. 1085)					
elasticbeanstalk:FromPlatform (p. 1085)					
ValidateConfigurationSettings	Grants permission to check the validity of a set of configuration settings for a configuration template or an environment.	Read	configurationsettings (p. 1084)	elasticbeanstalk:InApplication (p. 1085)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			environment (p. 1084)	elasticbeanstalk:InApplication (p. 1085)	

Resource Types Defined by AWS Elastic Beanstalk

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 1077\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
application	arn:\${Partition}:elasticbeanstalk:\${Region}: \${Account}:application/\${ApplicationName}	aws:ResourceTag/ \${TagKey} (p. 1085)
applicationversion	arn:\${Partition}:elasticbeanstalk: \${Region}: \${Account}:applicationversion/ \${ApplicationName}/\${VersionLabel}	aws:ResourceTag/ \${TagKey} (p. 1085) elasticbeanstalk:InApplication (p. 1085)
configurationtemplate	arn:\${Partition}:elasticbeanstalk: \${Region}: \${Account}:configurationtemplate/ \${ApplicationName}/\${TemplateName}	aws:ResourceTag/ \${TagKey} (p. 1085) elasticbeanstalk:InApplication (p. 1085)
environment	arn:\${Partition}:elasticbeanstalk:\${Region}: \${Account}:environment/\${ApplicationName}/ \${EnvironmentName}	aws:ResourceTag/ \${TagKey} (p. 1085) elasticbeanstalk:InApplication (p. 1085)
solutionstack	arn:\${Partition}:elasticbeanstalk: \${Region}:::solutionstack/ \${SolutionStackName}	
platform	arn:\${Partition}:elasticbeanstalk: \${Region}:::platform/ \${PlatformNameWithVersion}	

Condition Keys for AWS Elastic Beanstalk

AWS Elastic Beanstalk defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Filters actions based on the presence of tag key-value pairs in the request.	String
aws:ResourceTag/\${TagKey}	Filters actions based on tag key-value pairs attached to the resource.	String
aws:TagKeys	Filters actions based on the presence of tag keys in the request.	String
elasticbeanstalk:FrontendConstraint	Filters access by an application as a dependency or a constraint on an input parameter.	ARN
elasticbeanstalk:FrontendVersion	Filters access by an application version as a dependency or a constraint on an input parameter.	ARN
elasticbeanstalk:FrontendConfigurationTemplate	Filters access by a configuration template as a dependency or a constraint on an input parameter.	ARN
elasticbeanstalk:FrontendEnvironment	Filters access by an environment as a dependency or a constraint on an input parameter.	ARN
elasticbeanstalk:FrontendPlatform	Filters access by a platform as a dependency or a constraint on an input parameter.	ARN
elasticbeanstalk:FrontendSolutionStack	Filters access by a solution stack as a dependency or a constraint on an input parameter.	ARN
elasticbeanstalk:InAppliation	Filters access by the application that contains the resource that the action operates on.	ARN

Actions, Resources, and Condition Keys for Amazon Elastic Block Store

Amazon Elastic Block Store (service prefix: `ebs`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Elastic Block Store \(p. 1085\)](#)
- [Resource Types Defined by Amazon Elastic Block Store \(p. 1086\)](#)
- [Condition Keys for Amazon Elastic Block Store \(p. 1086\)](#)

Actions Defined by Amazon Elastic Block Store

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually

allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the **Resource** element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetSnapshotBlock	Grants permission to return the data of a block in an Amazon Elastic Block Store (EBS) snapshot	Read	snapshot* (p. 1086)		
ListChangedBlock	Grants permission to list the block indexes and block tokens for blocks that are different between two Amazon Elastic Block Store (EBS) snapshots of the same volume/snapshot lineage	Read	snapshot* (p. 1086)		
ListSnapshotBlock	Grants permission to list the block indexes and block tokens for blocks in an Amazon Elastic Block Store (EBS) snapshot.	Read	snapshot* (p. 1086)		

Resource Types Defined by Amazon Elastic Block Store

The following resource types are defined by this service and can be used in the **Resource** element of IAM permission policy statements. Each action in the [Actions table \(p. 1085\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
snapshot	<code>arn:\${Partition}:ec2:\${Region}::snapshot/\${SnapshotId}</code>	aws:ResourceTag/\${TagKey} (p. 1087)

Condition Keys for Amazon Elastic Block Store

Amazon Elastic Block Store defines the following condition keys that can be used in the **Condition** element of an IAM policy. You can use these keys to further refine the conditions under which the

policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:ResourceTag/ \${TagKey}	Filters access based on tag key-value pairs assigned to the AWS resource	String

Actions, Resources, and Condition Keys for Amazon Elastic Container Registry

Amazon Elastic Container Registry (service prefix: `ecr`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Elastic Container Registry \(p. 1087\)](#)
- [Resource Types Defined by Amazon Elastic Container Registry \(p. 1090\)](#)
- [Condition Keys for Amazon Elastic Container Registry \(p. 1090\)](#)

Actions Defined by Amazon Elastic Container Registry

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
BatchCheckLayerAvailability	Grants permission to check the availability of multiple image	Read	repository* (p. 1090)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
	layers in a specified registry and repository				
BatchDeleteImage	Grants permission to delete a list of specified images within a specified repository	Write	repository* (p. 1090)		
BatchGetImage	Grants permission to get detailed information for specified images within a specified repository	Read	repository* (p. 1090)		
CompleteLayerUpload	Grants permission to inform Amazon ECR that the image layer upload for a specified registry, repository name, and upload ID, has completed	Write	repository* (p. 1090)		
CreateRepository	Grants permission to create an image repository	Write	repository* (p. 1090)		
				aws:RequestTag/\${TagKey} (p. 1090)	
				aws:TagKeys (p. 1090)	
DeleteLifecyclePolicy	Grants permission to delete the specified lifecycle policy	Write	repository* (p. 1090)		
DeleteRepository	Grants permission to delete an existing image repository	Write	repository* (p. 1090)		
DeleteRepositoryPolicy	Grants permission to delete the repository policy from a specified repository	Write	repository* (p. 1090)		
DescribeImageScanningFindings	Grants permission to describe the image scan findings for the specified image	Read	repository* (p. 1090)		
DescribeImages	Grants permission to get metadata about the images in a repository, including image size, image tags, and creation date	Read	repository* (p. 1090)		
DescribeRepositories	Grants permission to describe image repositories in a registry	List	repository (p. 1090)		
GetAuthorizationToken	Grants permission to retrieve a token that is valid for a specified registry for 12 hours	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetDownloadUrlForLayer	Grants permission to retrieve the download URL corresponding to an image layer	Read	repository* (p. 1090)		
GetLifecyclePolicy	Grants permission to retrieve the specified lifecycle policy	Read	repository* (p. 1090)		
GetLifecyclePolicyPreview	Grants permission to retrieve the results of the specified lifecycle policy preview request	Read	repository* (p. 1090)		
GetRepositoryPolicy	Grants permission to retrieve the repository policy for a specified repository	Read	repository* (p. 1090)		
InitiateLayerUpload	Grants permission to notify Amazon ECR that you intend to upload an image layer	Write	repository* (p. 1090)		
ListImages	Grants permission to list all the image IDs for a given repository	List	repository* (p. 1090)		
ListTagsForResource	Grants permission to list the tags for an Amazon ECR resource	Read	repository* (p. 1090)		
PutImage	Grants permission to create or update the image manifest associated with an image	Write	repository* (p. 1090)		
PutImageScanningConfiguration	Grants permission to update the image scanning configuration for a repository	Write	repository* (p. 1090)		
PutImageTagMutability	Grants permission to update the image tag mutability settings for a repository	Write	repository* (p. 1090)		
PutLifecyclePolicy	Grants permission to create or update a lifecycle policy	Write	repository* (p. 1090)		
SetRepositoryPolicy	Grants permission to apply a repository policy on a specified repository to control access permissions	Permissions management	repository* (p. 1090)		
StartImageScan	Grants permission to start an image scan	Write	repository* (p. 1090)		
StartLifecyclePolicyPreview	Grants permission to start a preview of the specified lifecycle policy	Write	repository* (p. 1090)		
TagResource	Grants permission to tag an Amazon ECR resource	Tagging	repository* (p. 1090)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:RequestTag/ \${TagKey} (p. 1090) aws:TagKeys (p. 1090)	
UntagResource	Grants permission to untag an Amazon ECR resource	Tagging	repository* (p. 1090)		
UploadLayerPart	Grants permission to upload an image layer part to Amazon ECR	Write	repository* (p. 1090)		

Resource Types Defined by Amazon Elastic Container Registry

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1087\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
repository	<code>arn:\${Partition}:ecr:\${Region}: \${Account}:repository/\${RepositoryName}</code>	aws:ResourceTag/ \${TagKey} (p. 1090) ecr:ResourceTag/ \${TagKey} (p. 1091)

Condition Keys for Amazon Elastic Container Registry

Amazon Elastic Container Registry defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
<code>aws:RequestTag/\${TagKey}</code>	Filters create requests based on the allowed set of values for each of the tags.	String
<code>aws:ResourceTag/\${TagKey}</code>	Filters actions based on tag-value associated with the resource.	String
<code>aws:TagKeys</code>	Filters create requests based on the presence of mandatory tags in the request.	String

Condition Keys	Description	Type
ecr:ResourceTag/\${TagKey}	Filters actions based on tag-value associated with the resource.	String

Actions, Resources, and Condition Keys for Amazon Elastic Container Service

Amazon Elastic Container Service (service prefix: `ecs`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Elastic Container Service \(p. 1091\)](#)
- [Resource Types Defined by Amazon Elastic Container Service \(p. 1098\)](#)
- [Condition Keys for Amazon Elastic Container Service \(p. 1099\)](#)

Actions Defined by Amazon Elastic Container Service

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateCluster	Creates a new Amazon ECS cluster.	Write		aws:RequestTag/\${TagKey} (p. 1099) aws:TagKeys (p. 1099)	
CreateService	Runs and maintains a desired number of tasks from a specified task definition.	Write	service* (p. 1098)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				ecs:cluster (p. 1099) ecs:task-definition (p. 1099) aws:RequestTag/\${TagKey} (p. 1099) aws:TagKeys (p. 1099)	
CreateTaskSet	Creates a new Amazon ECS task set.	Write		ecs:cluster (p. 1099) ecs:service (p. 1099) ecs:task-definition (p. 1099)	
DeleteAccountSetting	Modifies the ARN and resource ID format of a resource for a specified IAM user, IAM role, or the root user for an account. You can specify whether the new ARN and resource ID format are disabled for new resources that are created.	Write			
DeleteAttributes	Deletes one or more custom attributes from an Amazon ECS resource.	Write	container-instance* (p. 1098)		
				ecs:cluster (p. 1099)	
DeleteCluster	Deletes the specified cluster.	Write	cluster* (p. 1098)		
DeleteService	Deletes a specified service within a cluster.	Write	service* (p. 1098)		
				ecs:cluster (p. 1099)	
DeleteTaskSet	Deletes the specified task set.	Write	task-set* (p. 1099)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				ecs:cluster (p. 1099) ecs:service (p. 1099)	
DeregisterContainerInstances	Deregisters an Amazon ECS container instance from the specified cluster.	Write	cluster* (p. 1098)		
DeregisterTaskDefinition	Deregisters the specified task definition by family and revision.	Write			
DescribeClusters	Describes one or more of your clusters.	Read	cluster* (p. 1098)		
DescribeContainerInstances	Describes Amazon ECS container instances.	Read	container-instance* (p. 1098)		
				ecs:cluster (p. 1099)	
DescribeServices	Describes the specified services running in your cluster.	Read	service* (p. 1098)		
				ecs:cluster (p. 1099)	
DescribeTaskDefinition	Describes a task definition. You can specify a family and revision to find information about a specific task definition, or you can simply specify the family to find the latest ACTIVE revision in that family.	Read			
DescribeTaskSets	Describes Amazon ECS task sets.	Read	task-set* (p. 1099)		
				ecs:cluster (p. 1099) ecs:service (p. 1099)	
DescribeTasks	Describes a specified task or tasks.	Read	task* (p. 1098)		
				ecs:cluster (p. 1099)	
DiscoverPollEndpoint	Returns an endpoint for the Amazon ECS agent to poll for updates.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListAccountSettings	Lists the account settings for Amazon ECS resource for a specified principal.	List			
ListAttributes	Lists the attributes for Amazon ECS resources within a specified target type and cluster.	List	cluster* (p. 1098)		
ListClusters	Returns a list of existing clusters.	List			
ListContainerInstances	Returns a list of container instances in a specified cluster.	List	cluster* (p. 1098)		
ListServices	Lists the services that are running in a specified cluster.	List		ecs:cluster (p. 1099)	
ListTagsForResource	List tags for the specified resource.	List	cluster (p. 1098)		
			container-instance (p. 1098)		
			task (p. 1098)		
			task-definition (p. 1099)		
ListTaskDefinitionFamilies	Returns a list of task definition families that are registered to your account (which may include task definition families that no longer have any ACTIVE task definitions).	List			
ListTaskDefinitions	Returns a list of task definitions that are registered to your account.	List			
ListTasks	Returns a list of tasks for a specified cluster.	List	container-instance* (p. 1098)		
				ecs:cluster (p. 1099)	
Poll [permission only]	Grants permission to an agent to connect with the Amazon ECS service to report status and get commands.	Write	container-instance* (p. 1098)		
				ecs:cluster (p. 1099)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
PutAccountSetting	Modifies the ARN and resource ID format of a resource for a specified IAM user, IAM role, or the root user for an account. You can specify whether the new ARN and resource ID format are enabled for new resources that are created. Enabling this setting is required to use new Amazon ECS features such as resource tagging.	Write			
PutAccountSettingDefault	Modifies the ARN and resource ID format of a resource type for all IAM users on an account for which no individual account setting has been set. Enabling this setting is required to use new Amazon ECS features such as resource tagging.	Write			
PutAttributes	Create or update an attribute on an Amazon ECS resource.	Write	container-instance* (p. 1098)		
				ecs:cluster (p. 1099)	
RegisterContainerInstances	Registers an EC2 instance into the specified cluster.	Write	cluster* (p. 1098)		
				aws:RequestTag/ {\$TagKey} (p. 1099)	
RegisterTaskDefinition	Registers a new task definition from the supplied family and containerDefinitions.	Write		aws:RequestTag/ {\$TagKey} (p. 1099)	
				aws:TagKeys (p. 1099)	
RunTask	Start a task using random placement and the default Amazon ECS scheduler.	Write	task-definition* (p. 1099)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				ecs:cluster (p. 1099) aws:RequestTag/ {\$TagKey} (p. 1099) aws:TagKeys (p. 1099)	
StartTask	Starts a new task from the specified task definition on the specified container instance or instances.	Write	task-definition* (p. 1099)		
				ecs:cluster (p. 1099) ecs:container- instances (p. 1099) aws:RequestTag/ {\$TagKey} (p. 1099) aws:TagKeys (p. 1099)	
StartTelemetrySession	Grants permission to start a telemetry session.	Write	container- instance* (p. 1098)		
				ecs:cluster (p. 1099)	
StopTask	Stops a running task.	Write	task* (p. 1098)		
				ecs:cluster (p. 1099)	
SubmitAttachmentStateChange	Sent to acknowledge that attachment changed states.	Write	cluster* (p. 1098)		
SubmitContainerStateChange	Sent to acknowledge that a container changed states.	Write	cluster* (p. 1098)		
SubmitTaskStateChange	Sent to acknowledge that a task changed states.	Write	cluster* (p. 1098)		
TagResource	Tags the specified resource.	Tagging	cluster (p. 1098)		
			container- instance (p. 1098)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions	
			service (p. 1098)			
			task (p. 1098)			
			task-definition (p. 1099)			
	Untags the specified resource.	Tagging	aws:TagKeys (p. 1099)			
				container-instance (p. 1098)		
				service (p. 1098)		
				task (p. 1098)		
				task-definition (p. 1099)		
	Updates the Amazon ECS container agent on a specified container instance.	Write	ecs:cluster (p. 1099)			
	Enables the user to modify the status of an Amazon ECS container instance.	Write	container-instance* (p. 1098)			
UpdateService	Modifies the desired count, deployment configuration, or task definition used in a service.	Write	service* (p. 1098)			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				ecs:cluster (p. 1099) ecs:task-definition (p. 1099)	
UpdateServicePrimaryTaskSet	Modifies the primary task set for the specified task service.	Write	service* (p. 1098)		
				ecs:cluster (p. 1099)	
UpdateTaskSet	Updates the specified task set.	Write	task-set* (p. 1099)		
				ecs:cluster (p. 1099) ecs:service (p. 1099)	

Resource Types Defined by Amazon Elastic Container Service

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1091\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
cluster	arn:\${Partition}:ecs:\${Region}: \${Account}:cluster/\${ClusterName}	aws:ResourceTag/ \${TagKey} (p. 1099) ecs:ResourceTag/ \${TagKey} (p. 1099)
container-instance	arn:\${Partition}:ecs:\${Region}: \${Account}:container-instance/ \${ContainerInstanceId}	aws:ResourceTag/ \${TagKey} (p. 1099) ecs:ResourceTag/ \${TagKey} (p. 1099)
service	arn:\${Partition}:ecs:\${Region}: \${Account}:service/\${ServiceName}	aws:ResourceTag/ \${TagKey} (p. 1099) ecs:ResourceTag/ \${TagKey} (p. 1099)
task	arn:\${Partition}:ecs:\${Region}: \${Account}:task/\${TaskId}	aws:ResourceTag/ \${TagKey} (p. 1099)

Resource Types	ARN	Condition Keys
		ecs:ResourceTag/ \${TagKey} (p. 1099)
task-definition	arn:\${Partition}:ecs:\${Region}: \${Account}:task-definition/ \${TaskDefinitionFamilyName}: \${TaskDefinitionRevisionNumber}	aws:ResourceTag/ \${TagKey} (p. 1099) ecs:ResourceTag/ \${TagKey} (p. 1099)
task-set	arn:\${Partition}:ecs:\${region}: \${Account}:task-set/\${ClusterName}/ \${ServiceName}/\${TaskSetId}	

Condition Keys for Amazon Elastic Container Service

Amazon Elastic Container Service defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/ \${TagKey}		String
aws:ResourceTag/ \${TagKey}		String
aws:TagKeys		String
ecs:ResourceTag/ \${TagKey}		String
ecs:cluster	The ARN of an ECS cluster.	ARN
ecs:container-instances	The ARN of an ECS container instance.	ARN
ecs:service	The ARN of an ECS service.	ARN
ecs:task-definition	The ARN of an ECS task definition.	ARN

Actions, Resources, and Condition Keys for Amazon Elastic Container Service for Kubernetes

Amazon Elastic Container Service for Kubernetes (service prefix: eks) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Elastic Container Service for Kubernetes \(p. 1100\)](#)
- [Resource Types Defined by Amazon Elastic Container Service for Kubernetes \(p. 1102\)](#)
- [Condition Keys for Amazon Elastic Container Service for Kubernetes \(p. 1103\)](#)

Actions Defined by Amazon Elastic Container Service for Kubernetes

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateCluster	Creates an Amazon EKS cluster.	Write		aws:RequestTag/\${TagKey} (p. 1103) aws:TagKeys (p. 1103)	
CreateFargateProfile	Creates an AWS Fargate profile.	Write	cluster* (p. 1103)		
				aws:RequestTag/\${TagKey} (p. 1103) aws:TagKeys (p. 1103)	
CreateNodegroup	Creates an Amazon EKS Nodegroup.	Write	cluster* (p. 1103)		
				aws:RequestTag/\${TagKey} (p. 1103)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:TagKeys (p. 1103)	
DeleteCluster	Deletes an Amazon EKS cluster.	Write	cluster* (p. 1103)		
DeleteFargateProfile	Deletes an AWS Fargate profile.	Write	fargateprofile* (p. 1103)		
DeleteNodegroup	Deletes an Amazon EKS Nodegroup.	Write	nodegroup* (p. 1103)		
DescribeCluster	Returns descriptive information about an Amazon EKS cluster.	Read	cluster* (p. 1103)		
DescribeFargateProfile	Returns descriptive information about an AWS Fargate profile associated with a cluster.	Read	fargateprofile* (p. 1103)		
DescribeNodegroup	Returns descriptive information about an Amazon EKS nodegroup.	Read	nodegroup* (p. 1103)		
DescribeUpdate	Describes a given update for a given Amazon EKS cluster/ nodegroup (in the specified or default region).	Read	cluster* (p. 1103)		
ListClusters	Lists the Amazon EKS clusters in your AWS account (in the specified or default region).	List			
ListFargateProfile	Lists the AWS Fargate profiles in your AWS account (in the specified or default region) associated with a given cluster.	List	cluster* (p. 1103)		
ListNodegroups	Lists the Amazon EKS nodegroups in your AWS account (in the specified or default region) attached to given cluster.	List	cluster* (p. 1103)		
ListTagsForResource	List tags for the specified resource.	List	cluster (p. 1103)		
			fargateprofile (p. 1103)		
			nodegroup (p. 1103)		
ListUpdates	Lists the updates for a given Amazon EKS cluster/nodegroup (in the specified or default region).	List	cluster* (p. 1103)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			nodegroup (p. 1103)		
TagResource	Tags the specified resource.	Tagging	cluster (p. 1103)		
			fargateprofile (p. 1103)		
			nodegroup (p. 1103)		
				aws:RequestTag/ \${TagKey} (p. 1103)	
UntagResource	Untags the specified resource.	Tagging		aws:TagKeys (p. 1103)	
			cluster (p. 1103)		
			fargateprofile (p. 1103)		
			nodegroup (p. 1103)		
UpdateClusterConfigurations	Update Amazon EKS cluster configurations (eg: API server endpoint access).	Write	cluster* (p. 1103)		
UpdateClusterVersion	Update the Kubernetes version of an Amazon EKS cluster.	Write	cluster* (p. 1103)		
UpdateNodegroupConfigurations	Update Amazon EKS nodegroup configurations (eg: min/max/ desired capacity or labels).	Write	nodegroup* (p. 1103)		
UpdateNodegroupVersion	Update the Kubernetes version of an Amazon EKS nodegroup.	Write	nodegroup* (p. 1103)		

Resource Types Defined by Amazon Elastic Container Service for Kubernetes

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1100\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
cluster	arn:\${Partition}:eks:\${Region}: \${Account}:cluster/\${ClusterName}	aws:ResourceTag/\${TagKey} (p. 1103)
nodegroup	arn:\${Partition}:eks:\${Region}: \${Account}:nodegroup/\${ClusterName}/ \${NodegroupName}/\${UUID}	aws:ResourceTag/\${TagKey} (p. 1103)
fargateprofile	arn:\${Partition}:eks:\${Region}: \${Account}:fargateprofile/\${ClusterName}/ \${FargateProfileName}/\${UUID}	aws:ResourceTag/\${TagKey} (p. 1103)

Condition Keys for Amazon Elastic Container Service for Kubernetes

Amazon Elastic Container Service for Kubernetes defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Filters access by a key that is present in the request the user makes to the EKS service.	String
aws:ResourceTag/\${TagKey}	Filters access by a tag key and value pair.	String
aws:TagKeys	Filters access by the list of all the tag key names present in the request the user makes to the EKS service.	String

Actions, Resources, and Condition Keys for Amazon Elastic File System

Amazon Elastic File System (service prefix: `elasticfilesystem`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Elastic File System \(p. 1104\)](#)
- [Resource Types Defined by Amazon Elastic File System \(p. 1106\)](#)
- [Condition Keys for Amazon Elastic File System \(p. 1107\)](#)

Actions Defined by Amazon Elastic File System

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
Backup [permission only]	Starts a backup job for an existing file system.	Write	file-system* (p. 1107)		
ClientMount [permission only]	Permission for allowing read-access to a file system.	Read			
ClientRootAccess [permission only]	Permission for allowing root-access to a file system.	Write	file-system* (p. 1107)		
ClientWrite [permission only]	Permission for allowing write-access to a file system.	Write	file-system* (p. 1107)		
CreateAccessPoint	Creates an access point for the specified file system.	Write	file-system* (p. 1107)		
CreateFileSystem	Creates a new, empty file system.	Tagging		aws:RequestTag/\${TagKey} (p. 1107) aws:TagKeys (p. 1107)	
CreateMountTarget	Creates a mount target for a file system.	Write	file-system* (p. 1107)		
CreateTags	Creates or overwrites tags associated with a file system.	Tagging	file-system* (p. 1107)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:RequestTag/ \${TagKey} (p. 1107) aws:TagKeys (p. 1107)	
DeleteAccessPoint	Deletes the specified access point.	Write	access-point* (p. 1107)		
DeleteFileSystem	Deletes a file system, permanently severing access to its contents.	Write	file-system* (p. 1107)		
DeleteFileSystemPolicy	Clears the resource-level policy given file system.	Write	file-system* (p. 1107)		
DeleteMountTarget	Deletes the specified mount target.	Write	file-system* (p. 1107)		
DeleteTags	Deletes the specified tags from a file system.	Tagging	file-system* (p. 1107)		
				aws:TagKeys (p. 1107)	
DescribeAccessPoint	Returns the descriptions of Amazon EFS access points.	List	file-system* (p. 1107)		
DescribeFileSystemLevelPolicy	Returns the current resource-level policy for a given file system.	Read	file-system (p. 1107)		
DescribeFileSystem	Returns the description of a specific Amazon EFS file system if either the file system CreationToken or the FileSystemId is provided; otherwise, returns descriptions of all file systems owned by the caller's AWS account in the AWS region of the endpoint that you're calling.	List	file-system (p. 1107)		
DescribeLifecycleConfiguration	Returns the current Lifecycle Configuration object for the specified Amazon EFS file system.	Read	file-system* (p. 1107)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeMountTargets	Returns the security groups currently in effect for a mount target.	Read	file-system* (p. 1107)		
DescribeMountTargets	Returns the descriptions of all the current mount targets, or a specific mount target, for a file system.	Read	file-system* (p. 1107)		
DescribeTags	Returns the tags associated with a file system.	Read	file-system* (p. 1107)		
ListTagsForResource	Returns the tags associated with the specified Amazon EFS resource.	Read	file-system* (p. 1107)		
ModifyMountTargetGroups	Modifies the set of security groups in effect for a mount target.	Write	file-system* (p. 1107)		
PutFileSystemPolicy	Apply a resource-level policy granting and/or restricting actions from given actors for the specified file system.	Write	file-system* (p. 1107)		
PutLifecycleConfiguration	Enables lifecycle management by creating a new LifecycleConfiguration object.	Write	file-system* (p. 1107)		
Restore [permission only]	Starts a restore job for an existing file system.	Write	file-system* (p. 1107)		
TagResource	Creates or overwrites tags associated with the specified Amazon EFS resource.	Tagging			
UntagResource	Deletes the specified tags from a specified Amazon EFS resource.	Tagging			
UpdateFileSystem	Updates the throughput mode or the amount of provisioned throughput of an existing file system.	Write	file-system* (p. 1107)		

Resource Types Defined by Amazon Elastic File System

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1104\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
file-system	arn:\${Partition}:elasticfilesystem:\${Region}:\${Account}:file-system/\${FileSystemId}	aws:ResourceTag/\${TagKey} (p. 1107)
access-point	arn:\${Partition}:elasticfilesystem:\${Region}:\${Account}:access-point/\${AccessPointId}	aws:ResourceTag/\${TagKey} (p. 1107)

Condition Keys for Amazon Elastic File System

Amazon Elastic File System defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Filters actions based on the allowed set of values for each of the tags	String
aws:ResourceTag/\${TagKey}	Filters actions based on tag key-value pairs attached to the resource	String
aws:TagKeys	Filters actions based on the presence of mandatory tags in the request	String

Actions, Resources, and Condition Keys for Amazon Elastic Inference

Amazon Elastic Inference (service prefix: `elastic-inference`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Elastic Inference \(p. 1108\)](#)
- [Resource Types Defined by Amazon Elastic Inference \(p. 1108\)](#)
- [Condition Keys for Amazon Elastic Inference \(p. 1108\)](#)

Actions Defined by Amazon Elastic Inference

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
Connect	Connects customer to Elastic Inference accelerator	Write	accelerator* (p. 1108)		

Resource Types Defined by Amazon Elastic Inference

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1108\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
accelerator	<code>arn:\${Partition}:elastic-inference:\${Region}:\${Account}:elastic-inference-accelerator/\${AcceleratorId}</code>	

Condition Keys for Amazon Elastic Inference

EI has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions in the IAM Policy Reference](#).

Actions, Resources, and Condition Keys for Elastic Load Balancing

Elastic Load Balancing (service prefix: `elasticloadbalancing`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).

- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Elastic Load Balancing \(p. 1109\)](#)
- [Resource Types Defined by Elastic Load Balancing \(p. 1111\)](#)
- [Condition Keys for Elastic Load Balancing \(p. 1112\)](#)

Actions Defined by Elastic Load Balancing

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AddTags	Adds the specified tags to the specified load balancer. Each load balancer can have a maximum of 10 tags.	Tagging	loadbalancer* (p. 1111)		
ApplySecurityGroupsToLoadBalancer	Associates one or more security groups with your load balancer in a virtual private cloud (VPC).	Write	loadbalancer* (p. 1111)		
AttachLoadBalancerSubnets	Adds one or more subnets to the set of configured subnets for the specified load balancer.	Write	loadbalancer* (p. 1111)		
ConfigureHealthCheck	Specifies the health check settings to use when evaluating the health state of your backend instances.	Write	loadbalancer* (p. 1111)		
CreateAppCookieStickinessPolicy	Generates a stickiness policy with sticky session lifetimes that follow that of an application-generated cookie.	Write	loadbalancer* (p. 1111)		
CreateLBCookieStickinessPolicy	Generates a stickiness policy with sticky session lifetimes controlled by the lifetime of	Write	loadbalancer* (p. 1111)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
	the browser (user-agent) or a specified expiration period.				
CreateLoadBalancer	Creates a load balancer.	Write	loadbalancer (p. 1111)		
CreateLoadBalancerListener	Creates one or more listeners for the specified load balancer.	Write	loadbalancer* (p. 1111)		
CreateLoadBalancerPolicy	Creates a policy with the specified attributes for the specified load balancer.	Write	loadbalancer* (p. 1111)		
DeleteLoadBalancer	Deletes the specified load balancer.	Write	loadbalancer* (p. 1111)		
DeleteLoadBalancerListeners	Deletes the specified listeners from the specified load balancer.	Write	loadbalancer* (p. 1111)		
DeleteLoadBalancerPolicy	Deletes the specified policy from the specified load balancer. This policy must not be enabled for any listeners.	Write	loadbalancer* (p. 1111)		
DeregisterInstancesFromLoadBalancer	Deregisters the specified instances from the specified load balancer.	Write	loadbalancer* (p. 1111)		
DescribeInstanceHealth	Describes the state of the specified instances with respect to the specified load balancer.	Read			
DescribeLoadBalancerAttributes	Describes the attributes for the specified load balancer.	Read			
DescribeLoadBalancerPolicies	Describes the specified policies.	Read			
DescribeLoadBalancerPolicyTypes	Describes the specified load balancer policy types.	Read			
DescribeLoadBalancers	Describes the specified the load balancers. If no load balancers are specified, the call describes all of your load balancers.	List			
DescribeTags	Describes the tags associated with the specified load balancers.	Read			
DetachLoadBalancerFromSubnets	Removes the specified subnets from the set of configured subnets for the load balancer.	Write	loadbalancer* (p. 1111)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DisableAvailabilityZonesForLoadBalancer	Removes the specified Availability Zones from the set of Availability Zones for the specified load balancer.	Write	loadbalancer* (p. 1111)		
EnableAvailabilityZonesForLoadBalancer	Adds the specified Availability Zones to the set of Availability Zones for the specified load balancer.	Write	loadbalancer* (p. 1111)		
ModifyLoadBalancerAttributes	Modifies the attributes of the specified load balancer.	Write	loadbalancer* (p. 1111)		
RegisterInstancesWithLoadBalancer	Adds the specified instances to the specified load balancer.	Write	loadbalancer* (p. 1111)		
RemoveTags	Removes one or more tags from the specified load balancer.	Tagging	loadbalancer* (p. 1111)		
SetLoadBalancerCertificates	Sets the certificate that terminates the specified listener's SSL connections.	Write	loadbalancer* (p. 1111)		
SetLoadBalancerPoliciesAssociatedWithPort	Replaces the set of policies associated with the specified port on which the back-end server is listening with a new set of policies.	Write	loadbalancer* (p. 1111)		
SetLoadBalancerPoliciesForListener	Replaces the current set of policies for the specified load balancer port with the specified set of policies.	Write	loadbalancer* (p. 1111)		

Resource Types Defined by Elastic Load Balancing

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1109\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
listener	arn:\${Partition}:elasticloadbalancing:\${Region}:\${Account}:listener/\${LoadBalancerName}/\${LoadBalancerId}/\${ListenerId}	
loadbalancer	arn:\${Partition}:elasticloadbalancing:\${Region}:\${Account}:loadbalancer/\${LoadBalancerName}	aws:RequestTag/tag-key (p. 1112)

Resource Types	ARN	Condition Keys
		aws:TagKeys (p. 1112) elasticloadbalancing:ResourceTag/tag-key (p. 1112)

Condition Keys for Elastic Load Balancing

Elastic Load Balancing defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
<code>aws:RequestTag/tag-key</code>	A key that is present in the request the user makes to the ELB service.	String
<code>aws:TagKeys</code>	The list of all the tag key names associated with the resource in the request.	String
<code>elasticloadbalancing:ResourceTag/</code>	The preface string for a tag key and value pair attached to a resource.	String
<code>elasticloadbalancing:ResourceTag/tag-key</code>	A tag key and value pair.	String

Actions, Resources, and Condition Keys for Elastic Load Balancing V2

Elastic Load Balancing V2 (service prefix: `elasticloadbalancing`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Elastic Load Balancing V2 \(p. 1113\)](#)
- [Resource Types Defined by Elastic Load Balancing V2 \(p. 1117\)](#)
- [Condition Keys for Elastic Load Balancing V2 \(p. 1118\)](#)

Actions Defined by Elastic Load Balancing V2

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AddListenerCertificates	Adds the specified certificates to the specified secure listener.	Write	listener/app* (p. 1117)		
			listener/net* (p. 1117)		
AddTags	Adds the specified tags to the specified load balancer. Each load balancer can have a maximum of 10 tags.	Tagging	loadbalancer/app/ (p. 1117)		
			loadbalancer/net/ (p. 1118)		
			targetgroup (p. 1118)		
CreateListener	Creates a listener for the specified Application Load Balancer.	Write	loadbalancer/app/ (p. 1117)		
			loadbalancer/net/ (p. 1118)		
CreateLoadBalancer	Creates a load balancer.	Write	loadbalancer/app/ (p. 1117)		
			loadbalancer/net/ (p. 1118)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateRule	Creates a rule for the specified listener.	Write	listener/app* (p. 1117)		
			listener/net* (p. 1117)		
CreateTargetGroup	Creates a target group.	Write	targetgroup* (p. 1118)		
DeleteListener	Deletes the specified listener.	Write	listener/app* (p. 1117)		
			listener/net* (p. 1117)		
DeleteLoadBalancer	Deletes the specified load balancer.	Write	loadbalancer/app/ (p. 1117)		
			loadbalancer/net/ (p. 1118)		
DeleteRule	Deletes the specified rule.	Write	listener-rule/app* (p. 1117)		
			listener-rule/net* (p. 1117)		
DeleteTargetGroup	Deletes the specified target group.	Write	targetgroup* (p. 1118)		
DeregisterTargets	Deregisters the specified targets from the specified target group.	Write	targetgroup* (p. 1118)		
DescribeAccountBalancing	Describes the Elastic Load Balancing resource limits for the AWS account.	Read			
DescribeListenerCertificates	Describes the certificates for the specified secure listener.	Read			
DescribeListeners	Describes the specified listeners or the listeners for the specified Application Load Balancer.	Read			
DescribeLoadBalancers	Describes the attributes for the specified load balancer.	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeLoadBalancers	Describes the specified load balancers. If no load balancers are specified, the call describes all of your load balancers.	Read			
DescribeRules	Describes the specified rules or the rules for the specified listener.	Read			
DescribeSSLPolicies	Describes the specified policies or all policies used for SSL negotiation.	Read			
DescribeTags	Describes the tags associated with the specified load balancers.	Read			
DescribeTargetGroups	Describes the attributes for the specified target group.	Read			
DescribeTargetGroups	Describes the specified target groups or all of your target groups.	Read			
DescribeTargetHealth	Describes the health of the specified targets or all of your targets.	Read			
ModifyListener	Modifies the specified properties of the specified listener.	Write	listener/app* (p. 1117)		
			listener/net* (p. 1117)		
ModifyLoadBalancerAttributes	Modifies the attributes of the specified load balancer.	Write	loadbalancer/app/ (p. 1117)		
			loadbalancer/net/ (p. 1118)		
ModifyRule	Modifies the specified rule.	Write	listener-rule/app* (p. 1117)		
			listener-rule/net* (p. 1117)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ModifyTargetGroup	Modifies the health checks used when evaluating the health state of the targets in the specified target group.	Write	targetgroup* (p. 1118)		
ModifyTargetGroup	Modifies the specified attributes of the specified target group.	Write	targetgroup* (p. 1118)		
RegisterTargets	Registers the specified targets with the specified target group.	Write	targetgroup* (p. 1118)		
RemoveListenerCertificates	Removes the specified certificates of the specified secure listener.	Write	listener/app* (p. 1117)		
			listener/net* (p. 1117)		
RemoveTags	Removes one or more tags from the specified load balancer.	Tagging	loadbalancer/app/ (p. 1117)		
loadbalancer/net/ (p. 1118)					
targetgroup (p. 1118)					
SetIpAddressType	Not found	Write	loadbalancer/app/ (p. 1117)		
	loadbalancer/net/ (p. 1118)				
SetRulePriorities	Sets the priorities of the specified rules.	Write	listener-rule/app* (p. 1117)		
	listener-rule/net* (p. 1117)				
SetSecurityGroup	Associates the specified security groups with the specified load balancer.	Write	loadbalancer/app/ (p. 1117)		
	loadbalancer/net/ (p. 1118)				

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
SetSubnets	Enables the Availability Zone for the specified subnets for the specified load balancer.	Write	loadbalancer/app/ (p. 1117)		
			loadbalancer/net/ (p. 1118)		
SetWebAcl [permission only]	Gives WebAcl permission to WAF	Write			

Resource Types Defined by Elastic Load Balancing V2

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1113\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
listener/app	arn:\${Partition}:elasticloadbalancing:\${Region}:\${Account}:listener/app/\${LoadBalancerName}/\${LoadBalancerId}/\${ListenerId}	
listener-rule/app	arn:\${Partition}:elasticloadbalancing:\${Region}:\${Account}:listener-rule/app/\${LoadBalancerName}/\${LoadBalancerId}/\${ListenerId}/\${ListenerRuleId}	
listener/net	arn:\${Partition}:elasticloadbalancing:\${Region}:\${Account}:listener/net/\${LoadBalancerName}/\${LoadBalancerId}/\${ListenerId}	
listener-rule/net	arn:\${Partition}:elasticloadbalancing:\${Region}:\${Account}:listener-rule/net/\${LoadBalancerName}/\${LoadBalancerId}/\${ListenerId}/\${ListenerRuleId}	
loadbalancer/app/	arn:\${Partition}:elasticloadbalancing:\${Region}:\${Account}:loadbalancer/app/\${LoadBalancerName}/\${LoadBalancerId}	aws:RequestTag/tag-key (p. 1118) aws:TagKeys (p. 1118) elasticloadbalancing:ResourceTag>tag-key (p. 1118)

Resource Types	ARN	Condition Keys
loadbalancer/net/	arn:\${Partition}:elasticloadbalancing:\${Region}:\${Account}:loadbalancer/net/\${LoadBalancerName}/\${LoadBalancerId}	aws:RequestTag/tag-key (p. 1118) aws:TagKeys (p. 1118) elasticloadbalancing:ResourceTag/tag-key (p. 1118)
targetgroup	arn:\${Partition}:elasticloadbalancing:\${Region}:\${Account}:targetgroup/\${TargetGroupName}/\${TargetGroupId}	aws:RequestTag/tag-key (p. 1118) aws:TagKeys (p. 1118) elasticloadbalancing:ResourceTag/tag-key (p. 1118)

Condition Keys for Elastic Load Balancing V2

Elastic Load Balancing V2 defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
<code>aws:RequestTag/tag-key</code>	A key that is present in the request the user makes to the ELB service.	String
<code>aws:TagKeys</code>	The list of all the tag key names associated with the resource in the request.	String
<code>elasticloadbalancing:ResourceTag/tag-key</code>	A tag key and value pair.	String

Actions, Resources, and Condition Keys for Amazon Elastic MapReduce

Amazon Elastic MapReduce (service prefix: `elasticmapreduce`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Elastic MapReduce \(p. 1119\)](#)
- [Resource Types Defined by Amazon Elastic MapReduce \(p. 1122\)](#)
- [Condition Keys for Amazon Elastic MapReduce \(p. 1123\)](#)

Actions Defined by Amazon Elastic MapReduce

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AddInstanceFleet	Grants permission to add an instance fleet to a running cluster.	Write	cluster* (p. 1122)		
AddInstanceGroups	Grants permission to add instance groups to a running cluster.	Write	cluster* (p. 1122)		
AddJobFlowSteps	Grants permission to add new steps to a running cluster.	Write	cluster* (p. 1122)		
AddTags	Grants permission to add tags to an Amazon EMR resource.	Tagging	cluster (p. 1122) editor (p. 1123)	aws:RequestTag/\${TagKey} (p. 1123) aws:TagKeys (p. 1123) elasticmapreduce:RequestTag/\${TagKey} (p. 1123)	
CancelSteps	Grants permission to cancel a pending step or steps in a running cluster.	Write	cluster* (p. 1122)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateEditor [permission only]	Grants permission to create an EMR notebook.	Tagging	cluster* (p. 1122)		
	aws:RequestTag/ \${TagKey} (p. 1123)				
	aws:TagKeys (p. 1123)		elasticmapreduce:RequestTag/ \${TagKey} (p. 1123)		
CreateSecurityConfiguration	Grants permission to create a security configuration.	Write			
DeleteEditor [permission only]	Grants permission to delete an EMR notebook.	Write	editor* (p. 1123)		
DeleteSecurityConfiguration	Grants permission to delete a security configuration.	Write			
DescribeCluster	Grants permission to get details about a cluster, including status, hardware and software configuration, VPC settings, and so on.	Read	cluster* (p. 1122)		
DescribeEditor [permission only]	Grants permission to view information about a notebook, including status, user, role, tags, location, and more.	Read	editor* (p. 1123)		
DescribeSecurityConfiguration	Grants permission to get details about a security configuration.	Read			
DescribeStep	Grants permission to get details about a cluster step.	Read	cluster* (p. 1122)		
GetBlockPublicAccess	Grants permission to retrieve the EMR block public access configuration for the AWS account in the Region.	Read			
ListBootstrapActions	Grants permission to get details about the bootstrap actions associated with a cluster.	List	cluster* (p. 1122)		
ListClusters	Grants permission to get the status of accessible clusters.	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListEditors [permission only]	Grants permission to list summary information for accessible EMR notebooks.	List			
ListInstanceFleets	Grants permission to get details of instance fleets in a cluster.	Read	cluster* (p. 1122)		
ListInstanceGroups	Grants permission to get details of instance groups in a cluster.	List	cluster* (p. 1122)		
ListInstances	Grants permission to get details about the Amazon EC2 instances in a cluster.	List	cluster* (p. 1122)		
ListSecurityConfigurations	Grants permission to list available security configurations in this account by name, along with creation dates and times.	List			
ListSteps	Grants permission to list steps associated with a cluster.	List	cluster* (p. 1122)		
ModifyInstanceFleet	Grants permission to change the target On-Demand and target Spot capacities for a instance fleet.	Write	cluster* (p. 1122)		
ModifyInstanceGroup	Grants permission to change the number and configuration of EC2 instances for an instance group.	Write	cluster* (p. 1122)		
OpenEditorInConsole [permission only]	Grants permission to launch the Jupyter notebook editor for an EMR notebook from within the console.	Write	cluster* (p. 1122)		
			editor* (p. 1123)		
PutAutoScalingPolicy	Grants permission to create or update an automatic scaling policy for a core instance group or task instance group.	Write	cluster* (p. 1122)		
PutBlockPublicAccess	Grants permission to create or update the EMR block public access configuration for the AWS account in the Region.	Permissions management			
RemoveAutoScalingPolicy	Grants permission to remove an automatic scaling policy from an instance group.	Write	cluster* (p. 1122)		
RemoveTags	Grants permission to remove tags from an Amazon EMR resource.	Tagging	cluster (p. 1122)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			editor (p. 1123)		
				aws:TagKeys (p. 1123)	
RunJobFlow	Grants permission to create and launch a cluster (job flow).	Tagging		aws:RequestTag/ \${TagKey} (p. 1123)	
				aws:TagKeys (p. 1123)	
				elasticmapreduce:RequestTag/ \${TagKey} (p. 1123)	
SetTerminationProtection	Grants permission to add and remove termination protection for a cluster.	Write	cluster* (p. 1122)		
StartEditor [permission only]	Grants permission to start an EMR notebook.	Write	cluster* (p. 1122)		
editor* (p. 1123)					
StopEditor [permission only]	Grants permission to shut down an EMR notebook.	Write	editor* (p. 1123)		
TerminateJobFlow	Grants permission to terminate a cluster (job flow).	Write	cluster* (p. 1122)		
ViewEventsFromEMRUsingConsole	Grants permission to use the EMR Using the Console console to view events from all clusters.	List			

Resource Types Defined by Amazon Elastic MapReduce

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1119\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
cluster	arn:\${Partition}:elasticmapreduce:\${Region}:\${Account}:cluster/\${ClusterId}	aws:ResourceTag/ \${TagKey} (p. 1123)

Resource Types	ARN	Condition Keys
		elasticmapreduce:ResourceTag/ \${TagKey} (p. 1123)
editor	arn:\${Partition}:elasticmapreduce:\${Region}: \${Account}:editor/\${EditorId}	aws:ResourceTag/ \${TagKey} (p. 1123) elasticmapreduce:ResourceTag/ \${TagKey} (p. 1123)

Condition Keys for Amazon Elastic MapReduce

Amazon Elastic MapReduce defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/ \${TagKey}	Filters access based on whether the tag and value pair is provided with the action	String
aws:ResourceTag/ \${TagKey}	Filters access based on the tag and value pair associated with an Amazon EMR resource	String
aws:TagKeys	Filters access based on whether the tag keys are provided with the action regardless of tag value	String
elasticmapreduce:ProvidedTag/ \${TagKey}	Filters actions based on whether the tag and value pair is provided with the action	String
elasticmapreduce:ResourceTag/ \${TagKey}	Filters actions based on the tag and value pair associated with an Amazon EMR resource	String

Actions, Resources, and Condition Keys for Amazon Elastic Transcoder

Amazon Elastic Transcoder (service prefix: `elastictranscoder`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Elastic Transcoder \(p. 1124\)](#)
- [Resource Types Defined by Amazon Elastic Transcoder \(p. 1125\)](#)
- [Condition Keys for Amazon Elastic Transcoder \(p. 1126\)](#)

Actions Defined by Amazon Elastic Transcoder

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CancelJob	Cancel a job that Elastic Transcoder has not begun to process	Write	job* (p. 1125)		
CreateJob	Create a job.	Write	pipeline* (p. 1125)		
			preset* (p. 1125)		
CreatePipeline	Create a pipeline	Write	pipeline* (p. 1125)		
CreatePreset	Create a preset.	Write	preset* (p. 1125)		
DeletePipeline	Delete a pipeline	Write	pipeline* (p. 1125)		
DeletePreset	Delete a preset	Write	preset* (p. 1125)		
ListJobsByPipeline	Get a list of the jobs that you assigned to a pipeline	List	pipeline* (p. 1125)		
ListJobsByStatus	Get information about all of the jobs associated with the current AWS account that have a specified status	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListPipelines	Get a list of the pipelines associated with the current AWS account	List			
ListPresets	Get a list of all presets associated with the current AWS account.	List			
ReadJob	Get detailed information about a job	Read	job* (p. 1125)		
ReadPipeline	Get detailed information about a pipeline	Read	pipeline* (p. 1125)		
ReadPreset	Get detailed information about a preset.	Read	preset* (p. 1125)		
TestRole	Test the settings for a pipeline to ensure that Elastic Transcoder can create and process jobs	Write			
UpdatePipeline	Update settings for a pipeline	Write	pipeline* (p. 1125)		
UpdatePipelineNotification	Update only Amazon Simple Notification Service (Amazon SNS) notifications for a pipeline	Write	pipeline* (p. 1125)		
UpdatePipelineStatus	Pause or reactivate a pipeline, so the pipeline stops or restarts processing jobs, update the status for the pipeline.	Write	pipeline* (p. 1125)		

Resource Types Defined by Amazon Elastic Transcoder

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1124\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
job	<code>arn:\${Partition}:elastictranscoder:\${Region}:\${Account}:job/\${JobId}</code>	
pipeline	<code>arn:\${Partition}:elastictranscoder:\${Region}:\${Account}:pipeline/\${PipelineId}</code>	
preset	<code>arn:\${Partition}:elastictranscoder:\${Region}:\${Account}:preset/\${PresetId}</code>	

Condition Keys for Amazon Elastic Transcoder

Elastic Transcoder has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Amazon ElastiCache

Amazon ElastiCache (service prefix: `elasticache`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon ElastiCache \(p. 1126\)](#)
- [Resource Types Defined by Amazon ElastiCache \(p. 1131\)](#)
- [Condition Keys for Amazon ElastiCache \(p. 1131\)](#)

Actions Defined by Amazon ElastiCache

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Note

When you create an ElastiCache policy in IAM you must use the "*" wildcard character for the `Resource` block. For information about using the following ElastiCache API actions in an IAM policy, see [ElastiCache Actions and IAM](#) in the *Amazon ElastiCache User Guide*.

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AddTagsToResource	The <code>AddTagsToResource</code> action adds up to 10 cost allocation tags to the named resource.	Tagging			
AuthorizeCacheSecurityGroupIngress	The <code>AuthorizeCacheSecurityGroupIngress</code>	Write			<code>ec2:AuthorizeSecurityGroupIngress</code>

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
	action allows network ingress to a cache security group.				
CopySnapshot	The CopySnapshot action makes a copy of an existing snapshot.	Write			s3:DeleteObject s3:GetBucketAcl s3:PutObject
CreateCacheCluster	The CreateCacheCluster action creates a cache cluster.	Write			ec2:CreateNetworkInterface ec2:DeleteNetworkInterface ec2:DescribeNetworkInterface ec2:DescribeSubnets ec2:DescribeVpcs s3:GetObject
CreateCacheParameterGroup	The CreateCacheParameterGroup action creates a new cache parameter group.	Write			
CreateCacheSecurityGroup	The CreateCacheSecurityGroup action creates a new cache security group.	Write			
CreateCacheSubnetGroup	The CreateCacheSubnetGroup action creates a new cache subnet group.	Write			
CreateReplicationGroup	The CreateReplicationGroup action creates a replication group.	Write			ec2:CreateNetworkInterface ec2:DeleteNetworkInterface ec2:DescribeNetworkInterface ec2:DescribeSubnets ec2:DescribeVpcs s3:GetObject
CreateSnapshot	The CreateSnapshot action creates a copy of an entire cache cluster at a specific moment in time.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DecreaseReplicaCount	The DecreaseReplicaCount action decreases the number of replicas in a Redis replication group.	Write			ec2:CreateNetworkInterface ec2:DeleteNetworkInterface ec2:DescribeNetworkInterface ec2:DescribeSubnets ec2:DescribeVpcs
DeleteCacheCluster	The DeleteCacheCluster action deletes a previously provisioned cache cluster.	Write			
DeleteCacheParameterGroup	The DeleteCacheParameterGroup action deletes the specified cache parameter group.	Write			
DeleteCacheSecurityGroup	The DeleteCacheSecurityGroup action deletes a cache security group.	Write			
DeleteCacheSubnetGroup	The DeleteCacheSubnetGroup action deletes a cache subnet group.	Write			
DeleteReplicationGroup	The DeleteReplicationGroup action deletes an existing replication group.	Write			
DeleteSnapshot	The DeleteSnapshot action deletes an existing snapshot.	Write			
DescribeCacheClusters	The DescribeCacheClusters action returns information about all provisioned cache clusters if no cache cluster identifier is specified, or about a specific cache cluster if a cache cluster identifier is supplied.	List			
DescribeCacheEngines	The DescribeCacheEngines action returns a list of the available cache engines and their versions.	List			
DescribeCacheParameterGroups	The DescribeCacheParameterGroups action returns a list of cache parameter group descriptions.	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeCacheParameters	The <code>DescribeCacheParameters</code> action returns the detailed parameter list for a particular cache parameter group.	List			
DescribeCacheSecurityGroups	The <code>DescribeCacheSecurityGroups</code> action returns a list of cache security group descriptions.	List			
DescribeCacheSubnetGroups	The <code>DescribeCacheSubnetGroups</code> action returns a list of cache subnet group descriptions.	List			
DescribeEngineDefaultParameters	The <code>DescribeEngineDefaultParameters</code> action returns the default engine and system parameter information for the specified cache engine.	List			
DescribeEvents	The <code>DescribeEvents</code> action returns events related to cache clusters, cache security groups, and cache parameter groups.	List			
DescribeReplicationGroups	The <code>DescribeReplicationGroups</code> action returns information about a particular replication group.	List			
DescribeReservedCacheNodes	The <code>DescribeReservedCacheNodes</code> action returns information about reserved cache nodes for this account, or about a specified reserved cache node.	List			
DescribeReservedCacheNodesOfferings	The <code>DescribeReservedCacheNodesOfferings</code> action lists available reserved cache node offerings.	List			
DescribeSnapshots	The <code>DescribeSnapshots</code> action returns information about cache cluster snapshots.	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
IncreaseReplicaCount	The IncreaseReplicaCount action increases the number of replicas in a Redis replication group.	Write			ec2:CreateNetworkInterface ec2:DeleteNetworkInterface ec2:DescribeNetworkInterfaces ec2:DescribeSubnets ec2:DescribeVpcs
ListAllowedNodeTypeModifications	List Allowed Node Type Modifications	List			
ListTagsForResource	The ListTagsForResource action lists all cost allocation tags currently on the named resource.	Read			
ModifyCacheCluster	The ModifyCacheCluster action modifies the settings for a cache cluster.	Write			
ModifyCacheParameterGroup	The ModifyCacheParameterGroup action modifies the parameters of a cache parameter group.	Write			
ModifyCacheSubnetGroup	The ModifyCacheSubnetGroup action modifies an existing cache subnet group.	Write			
ModifyReplicationGroup	The ModifyReplicationGroup action modifies the settings for a replication group.	Write			
ModifyReplicationGroupShardConfiguration	The ModifyReplicationGroupShardConfiguration action allows you to add shards, remove shards, or rebalance the keyspaces among existing shards.	Write			ec2:CreateNetworkInterface ec2:DeleteNetworkInterface ec2:DescribeNetworkInterfaces ec2:DescribeSubnets ec2:DescribeVpcs
PurchaseReservedCacheNodesOffering	The PurchaseReservedCacheNodesOffering action allows you to purchase a reserved cache node offering.	Write			
RebootCacheCluster	The RebootCacheCluster action reboots some, or all, of the cache nodes within a provisioned cache cluster.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
RemoveTagsFromResource	The RemoveTagsFromResource action removes the tags identified by the TagKeys list from the named resource.	Tagging			
ResetCacheParameterGroup	The ResetCacheParameterGroup action modifies the parameters of a cache parameter group to the engine or system default value.	Write			
RevokeCacheSecurityGroupIngress	The RevokeCacheSecurityGroupIngress action revokes ingress from a cache security group.	Write			
TestFailover	The TestFailover action allows you to test automatic failover on a specified node group in a replication group	Write			ec2:CreateNetworkInterface ec2:DeleteNetworkInterface ec2:DescribeNetworkInterfaces ec2:DescribeSubnets ec2:DescribeVpcs

Resource Types Defined by Amazon ElastiCache

Amazon ElastiCache does not support specifying a resource ARN in the Resource element of an IAM policy statement. To allow access to Amazon ElastiCache, specify “Resource”: “*” in your policy.

Condition Keys for Amazon ElastiCache

ElastiCache has no service-specific context keys that can be used in the Condition element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Note

For information about conditions in an IAM policy to control access to ElastiCache, see [ElastiCache Keys](#) in the *Amazon ElastiCache User Guide*.

Actions, Resources, and Condition Keys for Amazon Elasticsearch Service

Amazon Elasticsearch Service (service prefix: es) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).

Topics

- [Actions Defined by Amazon Elasticsearch Service \(p. 1132\)](#)
- [Resource Types Defined by Amazon Elasticsearch Service \(p. 1134\)](#)
- [Condition Keys for Amazon Elasticsearch Service \(p. 1134\)](#)

Actions Defined by Amazon Elasticsearch Service

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AddTags	Grants permission to attach resource tags to an Amazon ES domain.	Tagging	domain* (p. 1134)		
CreateElasticsearchDomain	Grants permission to create an Amazon ES domain.	Write	domain (p. 1134)		
DeleteElasticsearchDomain	Grants permission to delete an Amazon ES domain and all of its data.	Write	domain* (p. 1134)		
DeleteElasticsearchServiceLinkedRole	Grants permission to delete the service-linked role required for Amazon ES domains that use VPC access.	Write			
DescribeElasticsearchDomain	Grants permission to view a <code>description</code> of the domain configuration for the specified Amazon ES domain, including the domain ID, domain service endpoint, and domain ARN.	Read	domain* (p. 1134)		
DescribeElasticsearchDomainConfig	Grants permission to view a <code>description</code> of the configuration options and status of an Amazon ES domain.	Read	domain* (p. 1134)		
DescribeElasticsearchDomainStatistics	Grants permission to view a <code>description</code> of the domain	List	domain* (p. 1134)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
	configuration for up to five specified Amazon ES domains.				
DescribeElasticsearchDomainType	Grants permission to view the instance type, storage, and master node limits for a given Elasticsearch version and instance type.	List			
DescribeReservedESInstancesOfferings	Grants permission to fetch Reserved instance offerings for ES	List			
DescribeReservedESInstances	Grants permission to fetch ES reserved instances already purchased by customer	List			
ESHttpDelete	Grants permission to send HTTP DELETE requests to the Elasticsearch APIs.	Write	domain (p. 1134)		
ESHttpGet	Grants permission to send HTTP GET requests to the Elasticsearch APIs.	Read	domain (p. 1134)		
ESHttpHead	Grants permission to send HTTP HEAD requests to the Elasticsearch APIs.	Read	domain (p. 1134)		
ESHttpPost	Grants permission to send HTTP POST requests to the Elasticsearch APIs.	Write	domain (p. 1134)		
ESHttpPut	Grants permission to send HTTP PUT requests to the Elasticsearch APIs.	Write	domain (p. 1134)		
GetCompatibleElasticsearchVersions	Grants permission to fetch list of compatible elastic search versions to which Amazon ES domain can be upgraded	List	domain* (p. 1134)		
GetUpgradeHistory	Grants permission to fetch upgrade history for given ES domain	Read	domain* (p. 1134)		
GetUpgradeStatus	Grants permission to fetch upgrade status for given ES domain	Read	domain* (p. 1134)		
ListDomainNames	Grants permission to display the names of all Amazon ES domains that the current user owns.	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListElasticsearchInstanceTypes	Grants permission to list all Elasticsearch instance types that are supported for a given Elasticsearch version.	List			
ListElasticsearchVersions	Grants permission to list all supported Elasticsearch versions on Amazon ES.	List			
ListTags	Grants permission to display all of the tags for an Amazon ES domain.	Read	domain* (p. 1134)		
PurchaseReservedESInstancesOffering	Grants permission to purchase Reserved Instances.	Write			
RemoveTags	Grants permission to remove tags from Amazon ES domains.	Tagging	domain* (p. 1134)		
UpdateElasticsearchDomainConfiguration	Grants permission to modify the configuration of an Amazon ES domain, such as the instance type or number of instances.	Write	domain* (p. 1134)		
UpgradeElasticsearchDomain	Grants permission to initiate upgrade of an Elasticsearch domain to given version	Write	domain* (p. 1134)		

Resource Types Defined by Amazon Elasticsearch Service

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1132\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
domain	arn:\${Partition}:es:\${Region}: \${Account}:domain/\${DomainName}	

Condition Keys for Amazon Elasticsearch Service

Elasticsearch Service has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for AWS Elemental MediaConnect

AWS Elemental MediaConnect (service prefix: `mediaconnect`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Elemental MediaConnect \(p. 1135\)](#)
- [Resource Types Defined by AWS Elemental MediaConnect \(p. 1136\)](#)
- [Condition Keys for AWS Elemental MediaConnect \(p. 1137\)](#)

Actions Defined by AWS Elemental MediaConnect

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AddFlowOutputs	Grants permission to add outputs to any flow.	Write			
CreateFlow	Grants permission to create flows.	Write			
DeleteFlow	Grants permission to delete flows.	Write			
DescribeFlow	Grants permission to display the details of a flow including the flow ARN, name, and Availability Zone, as well as details about the source, outputs, and entitlements.	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GrantFlowEntitlements	Grants permission to grant entitlements on any flow.	Write			
ListEntitlements	Grants permission to display a list of all entitlements that have been granted to the account.	List			
ListFlows	Grants permission to display a list of flows that are associated with this account.	List			
RemoveFlowOutputs	Grants permission to remove outputs from any flow.	Write			
RevokeFlowEntitlements	Grants permission to revoke entitlements on any flow.	Write			
StartFlow	Grants permission to start flows.	Write			
StopFlow	Grants permission to stop flows.	Write			
UpdateFlowEntitlements	Grants permission to update entitlements on any flow.	Write			
UpdateFlowOutputs	Grants permission to update outputs on any flow.	Write			
UpdateFlowSources	Grants permission to update the source of any flow.	Write			

Resource Types Defined by AWS Elemental MediaConnect

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1135\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
Entitlement	<code>arn:\${Partition}:mediaconnect:\${Region}: \${Account}:entitlement:\${FlowId}: \${EntitlementName}</code>	
Flow	<code>arn:\${Partition}:mediaconnect:\${Region}: \${Account}:flow:\${FlowId}: \${FlowName}</code>	
Output	<code>arn:\${Partition}:mediaconnect:\${Region}: \${Account}:output:\${OutputId}: \${OutputName}</code>	
Source	<code>arn:\${Partition}:mediaconnect:\${Region}: \${Account}:source:\${SourceId}: \${SourceName}</code>	

Condition Keys for AWS Elemental MediaConnect

MediaConnect has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for AWS Elemental MediaConvert

AWS Elemental MediaConvert (service prefix: `mediaconvert`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Elemental MediaConvert \(p. 1137\)](#)
- [Resource Types Defined by AWS Elemental MediaConvert \(p. 1140\)](#)
- [Condition Keys for AWS Elemental MediaConvert \(p. 1141\)](#)

Actions Defined by AWS Elemental MediaConvert

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AssociateCertificate	Grants permission to associate an AWS Certificate Manager (ACM) Amazon Resource Name (ARN) with AWS Elemental MediaConvert.	Write			
CancelJob	Grants permission to cancel an AWS Elemental MediaConvert job that is waiting in queue	Write	Job* (p. 1141)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateJob	Grants permission to create and submit an AWS Elemental MediaConvert job	Write	JobTemplate (p. 1141)		
			Preset (p. 1141)		
			Queue (p. 1141)		
CreateJobTemplate	Grants permission to create an AWS Elemental MediaConvert custom job template	Write	Preset (p. 1141)		
			Queue (p. 1141)		
				aws:RequestTag/\${TagKey} (p. 1141) aws:TagKeys (p. 1141)	
CreatePreset	Grants permission to create an AWS Elemental MediaConvert custom output preset	Write		aws:RequestTag/\${TagKey} (p. 1141) aws:TagKeys (p. 1141)	
CreateQueue	Grants permission to create an AWS Elemental MediaConvert job queue	Write		aws:RequestTag/\${TagKey} (p. 1141) aws:TagKeys (p. 1141)	
DeleteJobTemplate	Grants permission to delete an AWS Elemental MediaConvert custom job template	Write	JobTemplate* (p. 1141)		
DeletePreset	Grants permission to delete an AWS Elemental MediaConvert custom output preset	Write	Preset* (p. 1141)		
DeleteQueue	Grants permission to delete an AWS Elemental MediaConvert job queue	Write	Queue* (p. 1141)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeEndpoint	Grants permission to subscribe to the AWS Elemental MediaConvert service, by sending a request for an account-specific endpoint. All transcoding requests must be sent to the endpoint that the service returns.	List			
DisassociateCertificateArn	Grants permission to remove an association between the Amazon Resource Name (ARN) of an AWS Certificate Manager (ACM) certificate and an AWS Elemental MediaConvert resource.	Write			
GetJob	Grants permission to get an AWS Elemental MediaConvert job	Read	Job* (p. 1141)		
GetJobTemplate	Grants permission to get an AWS Elemental MediaConvert job template	Read	JobTemplate* (p. 1141)		
GetPreset	Grants permission to get an AWS Elemental MediaConvert output preset	Read	Preset* (p. 1141)		
GetQueue	Grants permission to get an AWS Elemental MediaConvert job queue	Read	Queue* (p. 1141)		
ListJobTemplates	Grants permission to list AWS Elemental MediaConvert job templates	List			
ListJobs	Grants permission to list AWS Elemental MediaConvert jobs	List	Queue (p. 1141)		
ListPresets	Grants permission to list AWS Elemental MediaConvert output presets	List			
ListQueues	Grants permission to list AWS Elemental MediaConvert job queues	List			
ListTagsForResource	Grants permission to retrieve the tags for a MediaConvert queue, preset, or job template	Read	JobTemplate (p. 1141)		
Preset (p. 1141)					
Queue (p. 1141)					

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
TagResource	Grants permission to add tags to a MediaConvert queue, preset, or job template	Tagging	JobTemplate (p. 1141)		
			Preset (p. 1141)		
			Queue (p. 1141)		
				aws:RequestTag/\${TagKey} (p. 1141)	aws:TagKeys (p. 1141)
UntagResource	Grants permission to remove tags from a MediaConvert queue, preset, or job template	Tagging	JobTemplate (p. 1141)		
			Preset (p. 1141)		
			Queue (p. 1141)		
				aws:TagKeys (p. 1141)	
UpdateJobTemplate	Grants permission to update an AWS Elemental MediaConvert custom job template	Write	JobTemplate* (p. 1141)		
			Preset (p. 1141)		
			Queue (p. 1141)		
UpdatePreset	Grants permission to update an AWS Elemental MediaConvert custom output preset	Write	Preset* (p. 1141)		
UpdateQueue	Grants permission to update an AWS Elemental MediaConvert job queue	Write	Queue* (p. 1141)		

Resource Types Defined by AWS Elemental MediaConvert

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 1137\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
Job	arn:\${Partition}:mediaconvert:\${Region}: \${Account}:jobs/\${JobId}	
Queue	arn:\${Partition}:mediaconvert:\${Region}: \${Account}:queues/\${QueueName}	aws:ResourceTag/\${TagKey} (p. 1141)
Preset	arn:\${Partition}:mediaconvert:\${Region}: \${Account}:presets/\${PresetName}	aws:ResourceTag/\${TagKey} (p. 1141)
JobTemplate	arn:\${Partition}:mediaconvert:\${Region}: \${Account}:jobTemplates/\${JobTemplateName}	aws:ResourceTag/\${TagKey} (p. 1141)
CertificateAssociation	arn:\${Partition}:mediaconvert:\${Region}: \${Account}:certificates/\${CertificateArn}	

Condition Keys for AWS Elemental MediaConvert

AWS Elemental MediaConvert defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Filters actions based on the presence of tag key-value pairs in the request	String
aws:ResourceTag/\${TagKey}	Filters actions based on tag key-value pairs attached to the resource	String
aws:TagKeys	Filters actions based on the presence of tag keys in the request	String

Actions, Resources, and Condition Keys for AWS Elemental MediaLive

AWS Elemental MediaLive (service prefix: `medialive`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Elemental MediaLive \(p. 1142\)](#)
- [Resource Types Defined by AWS Elemental MediaLive \(p. 1146\)](#)
- [Condition Keys for AWS Elemental MediaLive \(p. 1146\)](#)

Actions Defined by AWS Elemental MediaLive

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
BatchUpdateSchedule	Grants permission to add and remove actions from a channel's schedule.	Write	channel* (p. 1146)		
CreateChannel	Grants permission to create a channel	Tagging	channel* (p. 1146)		
			input* (p. 1146)		
				aws:RequestTag/\${TagKey} (p. 1147) aws:TagKeys (p. 1147)	
CreateInput	Grants permission to create an input	Tagging	input* (p. 1146)		
			input-security-group* (p. 1146)		
				aws:RequestTag/\${TagKey} (p. 1147) aws:TagKeys (p. 1147)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateInputSecurityGroup	Grants permission to create an input security group	Tagging	input-security-group* (p. 1146)		
				aws:RequestTag/ {\$TagKey} (p. 1147) aws:TagKeys (p. 1147)	
CreateMultiplex	Grants permission to create a multiplex	Tagging	multiplex* (p. 1146)		
				aws:RequestTag/ {\$TagKey} (p. 1147) aws:TagKeys (p. 1147)	
CreateTags	Grants permission to create tags for channels, inputs, input security groups, multiplexes, and reservations.	Tagging	channel (p. 1146)		
			input (p. 1146)		
			input-security-group (p. 1146)		
			multiplex (p. 1146)		
			reservation (p. 1146)		
				aws:TagKeys (p. 1147) aws:RequestTag/ {\$TagKey} (p. 1147)	
DeleteChannel	Grants permission to delete a channel	Write	channel* (p. 1146)		
DeleteInput	Grants permission to delete an input	Write	input* (p. 1146)		
DeleteInputSecurityGroup	Grants permission to delete an input security group	Write	input-security-group* (p. 1146)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteMultiplex	Grants permission to delete a multiplex	Write	multiplex* (p. 1146)		
DeleteReservation	Grants permission to delete an expired reservation	Write	reservation* (p. 1146)		
DeleteTags	Grants permission to delete tags from channels, inputs, input security groups, multiplexes, and reservations.	Tagging	channel (p. 1146)		
			input (p. 1146)		
			input-security-group (p. 1146)		
			multiplex (p. 1146)		
			reservation (p. 1146)		
				aws:TagKeys (p. 1147)	
DescribeChannel	Grants permission to get details about a channel	Read	channel* (p. 1146)		
DescribeInput	Grants permission to describe an input	Read	input* (p. 1146)		
DescribeInputSecurityGroup	Grants permission to describe an input security group	Read	input-security-group* (p. 1146)		
DescribeMultiplex	Grants permission to describe a multiplex	Read	multiplex* (p. 1146)		
DescribeOffering	Grants permission to get details about a reservation offering	Read	offering* (p. 1146)		
DescribeReservation	Grants permission to get details about a reservation	Read	reservation* (p. 1146)		
DescribeSchedule	Grants permission to view a list of actions scheduled on a channel.	Read	channel* (p. 1146)		
ListChannels	Grants permission to list channels	List			
ListInputSecurityGroups	Grants permission to list input security groups	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListInputs	Grants permission to list inputs	List			
ListMultiplexes	Grants permission to list multiplexes	List			
ListOfferings	Grants permission to list reservation offerings	List			
ListReservations	Grants permission to list reservations	List			
ListTagsForResource	Grants permission to list tags for channels, inputs, input security groups, multiplexes, and reservations.	List	channel (p. 1146)		
			input (p. 1146)		
			input-security-group (p. 1146)		
			multiplex (p. 1146)		
			reservation (p. 1146)		
PurchaseOffering	Grants permission to purchase a reservation offering	Tagging	offering* (p. 1146)		
			reservation* (p. 1146)		
				aws:RequestTag/\${TagKey} (p. 1147)	aws:TagKeys (p. 1147)
StartChannel	Grants permission to start a channel	Write	channel* (p. 1146)		
StartMultiplex	Grants permission to start a multiplex	Write	multiplex* (p. 1146)		
StopChannel	Grants permission to stop a channel	Write	channel* (p. 1146)		
StopMultiplex	Grants permission to stop a multiplex	Write	multiplex* (p. 1146)		
UpdateChannel	Grants permission to update a channel	Write	channel* (p. 1146)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UpdateChannelClass	Grants permission to update the class of a channel	Write	channel* (p. 1146)		
UpdateInput	Grants permission to update an input	Write	input* (p. 1146)		
UpdateInputSecurityGroup	Grants permission to update an input security group	Write	input-security-group* (p. 1146)		
UpdateMultiplex	Grants permission to update a multiplex	Write	multiplex* (p. 1146)		
UpdateReservation	Grants permission to update a reservation	Write	reservation* (p. 1146)		

Resource Types Defined by AWS Elemental MediaLive

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1142\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
channel	arn:\${Partition}:medialive:\${Region}: \${Account}:channel:*	aws:ResourceTag/\${TagKey} (p. 1147)
input	arn:\${Partition}:medialive:\${Region}: \${Account}:input:*	aws:ResourceTag/\${TagKey} (p. 1147)
input-security-group	arn:\${Partition}:medialive:\${Region}: \${Account}:inputSecurityGroup:*	aws:ResourceTag/\${TagKey} (p. 1147)
multiplex	arn:\${Partition}:medialive:\${Region}: \${Account}:multiplex:*	aws:ResourceTag/\${TagKey} (p. 1147)
reservation	arn:\${Partition}:medialive:\${Region}: \${Account}:reservation:*	aws:ResourceTag/\${TagKey} (p. 1147)
offering	arn:\${Partition}:medialive:\${Region}: \${Account}:offering:*	

Condition Keys for AWS Elemental MediaLive

AWS Elemental MediaLive defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	The tag for a MediaLive request.	String
aws:ResourceTag/\${TagKey}	The tag for a MediaLive resource.	String
aws:TagKeys	The tag keys for a MediaLive resource or request.	String

Actions, Resources, and Condition Keys for AWS Elemental MediaPackage

AWS Elemental MediaPackage (service prefix: `mediapackage`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Elemental MediaPackage \(p. 1147\)](#)
- [Resource Types Defined by AWS Elemental MediaPackage \(p. 1149\)](#)
- [Condition Keys for AWS Elemental MediaPackage \(p. 1149\)](#)

Actions Defined by AWS Elemental MediaPackage

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateChannel	Grants permission to create a channel in AWS Elemental MediaPackage.	Write		aws:RequestTag/\${TagKey} (p. 1150) aws:TagKeys (p. 1150)	
CreateOriginEndpoint	Grants permission to create an endpoint in AWS Elemental MediaPackage.	Write		aws:RequestTag/\${TagKey} (p. 1150) aws:TagKeys (p. 1150)	
DeleteChannel	Grants permission to delete a channel in AWS Elemental MediaPackage.	Write	channels* (p. 1149)		
DeleteOriginEndpoint	Grants permission to delete an endpoint in AWS Elemental MediaPackage.	Write	origin_endpoints* (p. 1149)		
DescribeChannel	Grants permission to view the details of a channel in AWS Elemental MediaPackage.	Read	channels* (p. 1149)		
DescribeOriginEndpoint	Grants permission to view the details of an endpoint in AWS Elemental MediaPackage.	Read	origin_endpoints* (p. 1149)		
ListChannels	Grants permission to view a list of channels in AWS Elemental MediaPackage.	Read			
ListOriginEndpoints	Grants permission to view a list of endpoints in AWS Elemental MediaPackage.	Read			
ListTagsForResource	Grants permission to list the tags assigned to a Channel or OriginEndpoint.	Read	channels (p. 1149)		
			origin_endpoints (p. 1149)		
RotateIngestEndpointCredentials	Grants permission to rotate credentials for a Channel in AWS Elemental MediaPackage.	Write	channels* (p. 1149)		
TagResource	Grants permission to assign tags to a Channel or OriginEndpoint.	Write	channels (p. 1149)		
			origin_endpoints (p. 1149)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:RequestTag/\${TagKey} (p. 1150) aws:TagKeys (p. 1150)	
UntagResource	Grants permission to delete tags to a Channel or OriginEndpoint.	Write	channels (p. 1149)		
			origin_endpoints (p. 1149)		
				aws:TagKeys (p. 1150)	
UpdateChannel	Grants permission to make changes to a channel in AWS Elemental MediaPackage.	Write	channels* (p. 1149)		
UpdateOriginEndpoint	Grants permission to make changes to an endpoint in AWS Elemental MediaPackage.	Write	origin_endpoints* (p. 1149)		

Resource Types Defined by AWS Elemental MediaPackage

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1147\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
channels	<code>arn:\${Partition}:mediapackage:\${Region}: \${Account}:channels/\${ChannelIdentifier}</code>	aws:ResourceTag/\${TagKey} (p. 1150)
origin_endpoints	<code>arn:\${Partition}:mediapackage: \${Region}: \${Account}:origin_endpoints/ \${OriginEndpointIdentifier}</code>	aws:ResourceTag/\${TagKey} (p. 1150)

Condition Keys for AWS Elemental MediaPackage

AWS Elemental MediaPackage defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}		String
aws:ResourceTag/\${TagKey}		String
aws:TagKeys		String

Actions, Resources, and Condition Keys for AWS Elemental MediaPackage VOD

AWS Elemental MediaPackage VOD (service prefix: `mediapackage-vod`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Elemental MediaPackage VOD \(p. 1150\)](#)
- [Resource Types Defined by AWS Elemental MediaPackage VOD \(p. 1151\)](#)
- [Condition Keys for AWS Elemental MediaPackage VOD \(p. 1152\)](#)

Actions Defined by AWS Elemental MediaPackage VOD

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateAsset	Grants permission to create an asset in AWS Elemental MediaPackage	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreatePackagingConfiguration	Grants permission to create a packaging configuration in AWS Elemental MediaPackage	Write			
CreatePackagingGroup	Grants permission to create a packaging group in AWS Elemental MediaPackage	Write			
DeleteAsset	Grants permission to delete an asset in AWS Elemental MediaPackage	Write	assets* (p. 1152)		
DeletePackagingConfiguration	Grants permission to delete a packaging configuration in AWS Elemental MediaPackage	Write	packaging-configurations* (p. 1152)		
DeletePackagingGroup	Grants permission to delete a packaging group in AWS Elemental MediaPackage	Write	packaging-groups* (p. 1152)		
DescribeAsset	Grants permission to view the details of an asset in AWS Elemental MediaPackage	Read	assets* (p. 1152)		
DescribePackagingConfiguration	Grants permission to view the details of a packaging configuration in AWS Elemental MediaPackage	Read	packaging-configurations* (p. 1152)		
DescribePackagingGroup	Grants permission to view the details of a packaging group in AWS Elemental MediaPackage	Read	packaging-groups* (p. 1152)		
ListAssets	Grants permission to view a list of assets in AWS Elemental MediaPackage	List			
ListPackagingConfigurations	Grants permission to view a list of packaging configurations in AWS Elemental MediaPackage	List			
ListPackagingGroups	Grants permission to view a list of packaging groups in AWS Elemental MediaPackage	List			

Resource Types Defined by AWS Elemental MediaPackage VOD

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1150\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
assets	arn:\${Partition}:mediapackage-vod:\${Region}: \${Account}:assets/\${AssetIdentifier}	
packaging-configurations	arn:\${Partition}:mediapackage-vod:\${Region}: \${Account}:packaging-configurations/ \${PackagingConfigurationIdentifier}	
packaging-groups	arn:\${Partition}:mediapackage-vod: \${Region}: \${Account}:packaging-groups/ \${PackagingGroupIdentifier}	

Condition Keys for AWS Elemental MediaPackage VOD

MediaPackage VOD has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for AWS Elemental MediaStore

AWS Elemental MediaStore (service prefix: `mediastore`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Elemental MediaStore \(p. 1152\)](#)
- [Resource Types Defined by AWS Elemental MediaStore \(p. 1154\)](#)
- [Condition Keys for AWS Elemental MediaStore \(p. 1154\)](#)

Actions Defined by AWS Elemental MediaStore

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateContainer	Grants permission to create containers.	Write			
DeleteContainer	Grants permission to delete any container in the current account.	Write			
DeleteContainerPolicy	Grants permission to delete the access policy of any container in the current account.	Permissions management			
DeleteCorsPolicy	Grants permission to delete the CORS policy from any container in the current account.	Write			
DeleteLifecyclePolicy	Grants permission to delete the lifecycle policy from any container in the current account.	Write			
DeleteObject	Grants permission to delete objects.	Write			
DescribeContainer	Grants permission to retrieve details on any container in the current account.	List			
DescribeObject	Grants permission to retrieve object metadata.	List			
GetContainerPolicy	Grants permission to retrieve the access policy of any container in the current account.	Read			
GetCorsPolicy	Grants permission to retrieve the CORS policy of any container in the current account.	Read			
GetLifecyclePolicy	Grants permission to retrieve the lifecycle policy that is assigned to any container in the current account.	Read			
GetObject	Grants permission to retrieve objects.	Read			
ListContainers	Grants permission to retrieve a list of containers in the current account.	List			
ListItems	Grants permission to retrieve a list of objects and folders in the current account.	List			
PutContainerPolicy	Grants permission to create or replace the access policy of any container in the current account.	Permissions management			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
PutCorsPolicy	Grants permission to add or modify the CORS policy of any container in the current account.	Write			
PutLifecyclePolicy	Grants permission to add or modify the lifecycle policy that is assigned to any container in the current account.	Write			
PutObject	Grants permission to upload objects.	Write			
StartAccessLogging	Grants permission to enable access logging on any container in the current account.	Write			
StopAccessLogging	Grants permission to disable access logging on any container in the current account.	Write			

Resource Types Defined by AWS Elemental MediaStore

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1152\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
container	<code>arn:\${Partition}:mediastore:\${Region}:\${Account}:container/\${ContainerName}</code>	

Condition Keys for AWS Elemental MediaStore

MediaStore has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for AWS Elemental MediaTailor

AWS Elemental MediaTailor (service prefix: `mediatailor`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Elemental MediaTailor \(p. 1155\)](#)
- [Resource Types Defined by AWS Elemental MediaTailor \(p. 1156\)](#)
- [Condition Keys for AWS Elemental MediaTailor \(p. 1156\)](#)

Actions Defined by AWS Elemental MediaTailor

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeletePlaybackConfiguration	Deletes the playback configuration for the specified name	Write	playbackConfiguration* (p. 1156)		
GetPlaybackConfiguration	Grants permission to retrieve the playback configuration for the specified name	Read	playbackConfiguration* (p. 1156)		
ListPlaybackConfigurations	Grants permission to retrieve the list of available configurations	List			
ListTagsForResource	Returns a list of the tags assigned to the specified playback configuration resource.	Read			
PutPlaybackConfiguration	Grants permission to add a new configuration	Write	playbackConfiguration* (p. 1156)		
				aws:RequestTag/\${TagKey} (p. 1156)	
				aws:TagKeys (p. 1156)	
TagResource	Adds tags to the specified playback configuration resource.	Tagging		aws:RequestTag/\${TagKey} (p. 1156)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:TagKeys (p. 1156)	
UntagResource	Removes tags from the specified playback configuration resource.	Tagging		aws:RequestTag/\${TagKey} (p. 1156) aws:TagKeys (p. 1156)	

Resource Types Defined by AWS Elemental MediaTailor

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1155\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
playbackConfiguration	<code>arn:\${Partition}:mediatailor:\${Region}: \${Account}:playbackConfiguration/\${ResourceId}</code>	aws:ResourceTag/\${TagKey} (p. 1156)

Condition Keys for AWS Elemental MediaTailor

AWS Elemental MediaTailor defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Filters actions based on the presence of tag key-value pairs in the request	String
aws:ResourceTag/\${TagKey}	Filters actions based on tag key-value pairs attached to the resource	String
aws:TagKeys	Filters actions based on the presence of tag keys in the request	String

Actions, Resources, and Condition Keys for Amazon EventBridge

Amazon EventBridge (service prefix: events) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon EventBridge \(p. 1157\)](#)
- [Resource Types Defined by Amazon EventBridge \(p. 1161\)](#)
- [Condition Keys for Amazon EventBridge \(p. 1161\)](#)

Actions Defined by Amazon EventBridge

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ActivateEventSource	Activates a partner event source that has been deactivated. Once activated, your matching event bus will start receiving events from the event source.	Write	event-source* (p. 1161)		
CreateEventBus	Creates a new event bus within your account. This can be a custom event bus which you can use to receive events from your own custom applications and services, or it can be a partner event bus which can be matched to a partner event source.	Write	event-bus* (p. 1161)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreatePartnerEventSource	Called by an AWS partner to create a partner event source.	Write	event-source* (p. 1161)		
DeactivateEventSource	Called by an AWS partner to deactivate a partner event source.	Write	event-source* (p. 1161)		
DeleteEventBus	Deletes the specified custom event bus or partner event bus. All rules associated with this event bus are also deleted. You can't delete your account's default event bus.	Write	event-bus* (p. 1161)		
DeletePartnerEventSource	Called by an AWS partner to delete a partner event source.	Write	event-source* (p. 1161)		
DeleteRule	Deletes a rule. You must remove all targets from a rule using RemoveTargets before you can delete the rule.	Write	rule* (p. 1161)		
DescribeEventBus	Displays the external AWS accounts that are permitted to write events to your account using your account's event bus, and the associated policy.	Read	event-bus (p. 1161)		
DescribeEventSource	Describes the details of the specified partner event source that is shared with your account.	Read	event-source* (p. 1161)		
DescribePartnerEventSource	Called by an AWS partner to describe the details of the specified partner event source that they have created.	Read	event-source* (p. 1161)		
DescribeRule	Describes the details of the specified rule.	Read	rule* (p. 1161)		
DisableRule	Disables a rule. A disabled rule won't match any events, and won't self-trigger if it has a schedule expression.	Write	rule* (p. 1161)		
EnableRule	Enables a rule. If the rule does not exist, the operation fails.	Write	rule* (p. 1161)		
ListEventBuses	Lists all the event buses in your account, including the default event bus, custom event buses, and partner event buses.	List	event-bus* (p. 1161)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListEventSources	Lists the event sources shared with this account.	List	event-source* (p. 1161)		
ListPartnerEvents	Called by an AWS partner to display the AWS account ID that the specified partner event source is associated with.	List	event-source* (p. 1161)		
ListPartnerEventSources	Called by an AWS partner to list all the partner event sources that they have created.	List	event-source* (p. 1161)		
ListRuleNamesByTarget	Lists the names of the rules that the given target is put to.	List	rule* (p. 1161)		
ListRules	Lists the Amazon EventBridge rules in your account.	List	rule* (p. 1161)		
ListTagsForResource	This action lists tags for an Amazon EventBridge resource.	List	rule* (p. 1161)		
ListTargetsByRule	Lists of targets assigned to the rule.	List	rule* (p. 1161)		
PutEvents	Sends custom events to Amazon EventBridge so that they can be matched to rules.	Write			
PutPartnerEvents	Sends custom events to Amazon EventBridge so that they can be matched to rules.	Write			
PutPermission	Running PutPermission permits the specified AWS account to put events to your account's default event bus.	Write			
PutRule	Creates or updates a rule. Rules are enabled by default, or based on value of the State parameter.	Tagging	rule* (p. 1161)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				events:detail.userIdentity.principalId (p. 1162) events:detail-type (p. 1161) events:source (p. 1162) events:detail.service (p. 1162) events:detail.eventTypeCode (p. 1161) aws:RequestTag/ \${TagKey} (p. 1161) aws:TagKeys (p. 1161)	
PutTargets	Adds target(s) to a rule. Targets are the resources that can be invoked when a rule is triggered.	Write	rule* (p. 1161)		
				events:TargetArn (p. 1161)	
RemovePermissions	Revokes the permission of another AWS account to be able to put events to your default event bus.	Write			
RemoveTargets	Removes target(s) from a rule so that when the rule is triggered, those targets will no longer be invoked.	Write	rule* (p. 1161)		
TagResource	This action tags an Amazon EventBridge resource.	Tagging	rule* (p. 1161)		
				aws:TagKeys (p. 1161) aws:RequestTag/ \${TagKey} (p. 1161)	
TestEventPattern	Tests whether an event pattern matches the provided event.	Read			
UntagResource	This action removes a tag from an Amazon EventBridge resource.	Tagging	rule* (p. 1161)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:TagKeys (p. 1161)	

Resource Types Defined by Amazon EventBridge

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1157\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
<code>event-source</code>	<code>arn:\${Partition}:events:\${Region}::event-source/\${EventSourceName}</code>	
<code>event-bus</code>	<code>arn:\${Partition}:events:\${Region}:\${Account}:event-bus/\${EventBusName}</code>	
<code>rule</code>	<code>arn:\${Partition}:events:\${Region}:\${Account}:rule/[\${EventBusName}/]\${RuleName}</code>	aws:ResourceTag/ <code> \${TagKey}</code> (p. 1161)

Condition Keys for Amazon EventBridge

Amazon EventBridge defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
<code>aws:RequestTag/ \${TagKey}</code>	Filters actions based on the allowed set of values for each of the tags	String
<code>aws:ResourceTag/ \${TagKey}</code>	Filters actions based on tag-value associated with the resource	String
<code>aws:TagKeys</code>	Filters actions based on the presence of mandatory tags in the request	String
<code>events:TargetArn</code>	The ARN of a target that can be put to a rule.	ARN
<code>events:detail-type</code>	Matches the literal string of the detail-type field of the event.	String
<code>events:detail.eventTypeCode</code>	Matches the literal string for the detail.eventTypeCode field of the event.	String

Condition Keys	Description	Type
events:detail.service	Matches the literal string for the detail.service field of the event.	String
events:detail.userIdentity.principalId	Matches the literal string for the detail.userIdentity.principalId field of the event.	String
events:source	The AWS service or AWS partner event source that generated the event. Matches the literal string of the source field of the event.	String

Actions, Resources, and Condition Keys for Amazon EventBridge Schemas

Amazon EventBridge Schemas (service prefix: schemas) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon EventBridge Schemas \(p. 1162\)](#)
- [Resource Types Defined by Amazon EventBridge Schemas \(p. 1165\)](#)
- [Condition Keys for Amazon EventBridge Schemas \(p. 1165\)](#)

Actions Defined by Amazon EventBridge Schemas

You can specify the following actions in the Action element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the Resource element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateDiscoverer	Creates an event schema discoverer. Once created, your	Write	discoverer* (p. 1165)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
	events will be automatically map into corresponding schema documents				
CreateRegistry	Create a new schema registry in your account.	Write	registry* (p. 1165)		
CreateSchema	Create a new schema in your account.	Write	schema* (p. 1165)		
DeleteDiscoverer	Deletes discoverer in your account.	Write	discoverer* (p. 1165)		
DeleteRegistry	Deletes an existing registry in your account.	Write	registry* (p. 1165)		
DeleteSchema	Deletes an existing schema in your account.	Write	schema* (p. 1165)		
DeleteSchemaVersion	Deletes a specific version of schema in your account.	Write	schema* (p. 1165)		
DescribeCodeBinding	Retrieves metadata for generated code for specific schema in your account.	Read	schema* (p. 1165)		
DescribeDiscoverer	Retrieves discoverer metadata in your account.	Read	discoverer* (p. 1165)		
DescribeRegistry	Describes an existing registry metadata in your account.	Read	registry* (p. 1165)		
DescribeSchema	Retrieves an existing schema in your account.	Read	schema* (p. 1165)		
GetCodeBinding	Retrieves metadata for generated code for specific schema in your account.	Read	schema* (p. 1165)		
GetDiscoveredSchema	Retrieves schema for the provided list of sample events.	Read			
ListDiscoverers	Lists all the discoverers in your account.	List	discoverer* (p. 1165)		
ListRegistries	List all discoverers in your account.	List	registry* (p. 1165)		
ListSchemaVersions	List all versions of a schema.	List	schema* (p. 1165)		
ListSchemas	List all schemas.	List	schema* (p. 1165)		
ListTagsForResource	This action lists tags for a resource.	List	discoverer* (p. 1165)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			registry* (p. 1165)		
			schema* (p. 1165)		
PutCodeBinding	Generates code for specific schema in your account.	Write	schema* (p. 1165)		
SearchSchemas	Searches schemas based on specified keywords in your account.	List	schema* (p. 1165)		
StartDiscoverer	Starts the specified discoverer. Once started the discoverer will automatically register schemas for published events to configured source in your account	Write	discoverer* (p. 1165)		
StopDiscoverer	Starts the specified discoverer. Once started the discoverer will automatically register schemas for published events to configured source in your account	Write	discoverer* (p. 1165)		
TagResource	This action tags an resource.	Tagging	discoverer* (p. 1165)		
registry* (p. 1165)					
schema* (p. 1165)					
	aws:TagKeys (p. 1165) aws:RequestTag/ {\$TagKey} (p. 1165)				
UntagResource	This action removes a tag from on a resource.	Tagging	discoverer* (p. 1165)		
registry* (p. 1165)					
schema* (p. 1165)					
	aws:TagKeys (p. 1165)				

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UpdateDiscoverer	Updates an existing discoverer in your account.	Write	discoverer* (p. 1165)		
UpdateRegistry	Updates an existing registry metadata in your account.	Write	registry* (p. 1165)		
UpdateSchema	Updates an existing schema in your account.	Write	schema* (p. 1165)		

Resource Types Defined by Amazon EventBridge Schemas

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1162\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
discoverer	<code>arn:\${Partition}:schemas:\${Region}: \${Account}:discoverer/\${DiscovererId}</code>	aws:ResourceTag/\${TagKey} (p. 1165)
registry	<code>arn:\${Partition}:schemas:\${Region}: \${Account}:registry/\${RegistryName}</code>	aws:ResourceTag/\${TagKey} (p. 1165)
schema	<code>arn:\${Partition}:schemas:\${Region}: \${Account}:schema/\${RegistryName}/ \${SchemaName}</code>	aws:ResourceTag/\${TagKey} (p. 1165)

Condition Keys for Amazon EventBridge Schemas

Amazon EventBridge Schemas defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Filters actions based on the allowed set of values for each of the tags	String
aws:ResourceTag/\${TagKey}	Filters actions based on tag-value associated with the resource	String
aws:TagKeys	Filters actions based on the presence of mandatory tags in the request	String

Actions, Resources, and Condition Keys for AWS Firewall Manager

AWS Firewall Manager (service prefix: `fms`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- View a list of the [API operations available for this service](#).

Topics

- [Actions Defined by AWS Firewall Manager \(p. 1166\)](#)
- [Resource Types Defined by AWS Firewall Manager \(p. 1168\)](#)
- [Condition Keys for AWS Firewall Manager \(p. 1169\)](#)

Actions Defined by AWS Firewall Manager

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AssociateAdminAccount	Sets the AWS Firewall Manager administrator account and enables the service in all organization accounts	Write			
DeleteNotificationManager	Deletes an AWS Firewall Manager association with the IAM role and the Amazon Simple Notification Service (SNS) topic that is used to notify the FM administrator about major FM events and errors across the organization.	Write			
DeletePolicy	Permanently deletes an AWS Firewall Manager policy.	Write	policy* (p. 1168)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:ResourceTag/ \${TagKey} (p. 1169)	
DisassociateAdminAccount	Disassociates the account that DisassociateAdminAccount has been set as the AWS Firewall Manager administrator account and disables the service in all organization accounts	Write			
GetAdminAccount	Returns the AWS Organizations master account that is associated with AWS Firewall Manager as the AWS Firewall Manager administrator.	Read			
GetComplianceDetails	Returns detailed compliance information about the specified member account. Details include resources that are in and out of compliance with the specified policy.	Read	policy* (p. 1168)		
GetNotificationChannel	Returns information about the Amazon Simple Notification Service (SNS) topic that is used to record AWS Firewall Manager SNS logs.	Read			
GetPolicy	Returns information about the specified AWS Firewall Manager policy.	Read	policy* (p. 1168)		
GetProtectionStatus	Returns policy-level attack summary information in the event of a potential DDoS attack.	Read	policy* (p. 1168)		
ListComplianceStatus	Returns an array of PolicyComplianceStatus objects in the response. Use PolicyComplianceStatus to get a summary of which member accounts are protected by the specified policy.	List	policy* (p. 1168)		
ListMemberAccounts	Returns an array of member account ids if the caller is FMS admin account.	List			
ListPolicies	Returns an array of PolicySummary objects in the response.	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListTagsForResource	Lists the Tags for a given resource.	Read	policy* (p. 1168)		
PutNotificationChannel	Designates the IAM role and Amazon Simple Notification Service (SNS) topic that AWS Firewall Manager (FM) could use to notify the FM administrator about major FM events and errors across the organization.	Write			
PutPolicy	Creates an AWS Firewall Manager policy.	Write	policy* (p. 1168)		
				aws:RequestTag/\${TagKey} (p. 1169) aws:TagKeys (p. 1169)	
TagResource	Adds a Tag to a given resource.	Tagging	policy* (p. 1168)		
				aws:RequestTag/\${TagKey} (p. 1169) aws:TagKeys (p. 1169)	
UntagResource	Removes a Tag from a given resource.	Tagging	policy* (p. 1168)		
				aws:TagKeys (p. 1169)	

Resource Types Defined by AWS Firewall Manager

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1166\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
policy	<code>arn:\${Partition}:fms:\${Region}:\${Account}:policy/\${Id}</code>	aws:ResourceTag/\${TagKey} (p. 1169)

Condition Keys for AWS Firewall Manager

AWS Firewall Manager defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
<code>aws:RequestTag/\${TagKey}</code>	Filters actions based on the allowed set of values for each of the tags	String
<code>aws:ResourceTag/\${TagKey}</code>	Filters actions based on tag-value associated with the resource	String
<code>aws:TagKeys</code>	Filters actions based on the presence of mandatory tags in the request	String

Actions, Resources, and Condition Keys for Amazon Forecast

Amazon Forecast (service prefix: `forecast`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).

Topics

- [Actions Defined by Amazon Forecast \(p. 1169\)](#)
- [Resource Types Defined by Amazon Forecast \(p. 1171\)](#)
- [Condition Keys for Amazon Forecast \(p. 1172\)](#)

Actions Defined by Amazon Forecast

You can specify the following actions in the Action element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the Resource element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateDataset	Creates a dataset	Write	dataset* (p. 1171)		
CreateDatasetGroup	Creates a dataset group	Write	datasetGroup* (p. 1171)		
CreateDatasetImportJob	Creates a dataset import job	Write	datasetImportJob* (p. 1171)		
CreateForecast	Creates a forecast	Write	predictor* (p. 1171)		
CreateForecastExportJob	Creates a forecast export job	Write	forecast* (p. 1171)		
CreatePredictor	Creates a predictor	Write	datasetGroup* (p. 1171)		
DeleteDataset	Deletes a dataset	Write	dataset* (p. 1171)		
DeleteDatasetGroup	Deletes a dataset group	Write	datasetGroup* (p. 1171)		
DeleteDatasetImportJob	Deletes a dataset import job	Write	datasetImportJob* (p. 1171)		
DeleteForecast	Deletes a forecast	Write	forecast* (p. 1171)		
DeleteForecastExportJob	Deletes a forecast export job	Write	forecastExport* (p. 1172)		
DeletePredictor	Deletes a predictor	Write	predictor* (p. 1171)		
DescribeDataset	Describes a dataset	Read	dataset* (p. 1171)		
DescribeDatasetGroup	Describes a dataset group	Read	datasetGroup* (p. 1171)		
DescribeDatasetImportJob	Describes a dataset import job	Read	datasetImportJob* (p. 1171)		
DescribeForecast	Describes a forecast	Read	forecast* (p. 1171)		
DescribeForecastExportJob	Describes a forecast export job	Read	forecastExport* (p. 1172)		
DescribePredictor	Describes a predictor	Read	predictor* (p. 1171)		
GetAccuracyMetrics	Gets Accuracy Metrics for a predictor	Read	predictor* (p. 1171)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListDatasetGroups	Lists dataset groups	List			
ListDatasetImportJobs	Lists dataset import jobs	List			
ListDatasets	Lists datasets	List			
ListForecastExportJobs	Lists forecast export jobs	List			
ListForecasts	Lists forecasts	List			
ListPredictors	Lists predictors	List			
QueryForecast	Retrieves a forecast for a single item	Read	forecast* (p. 1171)		
UpdateDatasetGroup	Updates a dataset group	Write	dataset* (p. 1171)		
			datasetGroup* (p. 1171)		

Resource Types Defined by Amazon Forecast

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1169\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
dataset	<code>arn:\${Partition}:forecast:\${Region}: \${Account}:dataset/\${ResourceId}</code>	
datasetGroup	<code>arn:\${Partition}:forecast:\${Region}: \${Account}:dataset-group/\${ResourceId}</code>	
datasetImportJob	<code>arn:\${Partition}:forecast:\${Region}: \${Account}:dataset-import-job/\${ResourceId}</code>	
algorithm	<code>arn:\${Partition}:forecast:::algorithm/ \${ResourceId}</code>	
predictor	<code>arn:\${Partition}:forecast:\${Region}: \${Account}:predictor/\${ResourceId}</code>	
forecast	<code>arn:\${Partition}:forecast:\${Region}: \${Account}:forecast/\${ResourceId}</code>	

Resource Types	ARN	Condition Keys
forecastExport	arn:\${Partition}:forecast:\${Region}: \${Account}:forecast-export-job/\${ResourceId}	

Condition Keys for Amazon Forecast

Forecast has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Amazon Fraud Detector

Amazon Fraud Detector (service prefix: `frauddetector`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Fraud Detector \(p. 1172\)](#)
- [Resource Types Defined by Amazon Fraud Detector \(p. 1176\)](#)
- [Condition Keys for Amazon Fraud Detector \(p. 1176\)](#)

Actions Defined by Amazon Fraud Detector

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
BatchCreateVariable	Creates a batch of variables.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
BatchGetVariable	Gets a batch of variables.	List			
CreateDetectorVersion	Creates a detector version. The detector version starts in a DRAFT status.	Write			
CreateModelVersion	Creates a version of the model using the specified model type.	Write			
CreateRule	Creates a rule for use with the specified detector.	Write			
CreateVariable	Creates a variable.	Write			
DeleteDetectorVersion	Deletes the detector version.	Write			
DeleteEvent	Deletes the specified event.	Write			
DescribeDetector	Gets all versions for a specified detector.	Read			
DescribeModelVersion	Gets all of the model versions for the specified model type or for the specified model type and model ID. You can also get details for a single, specified model version.	Read			
GetDetectorVersion	Gets a particular detector version.	List			
GetDetectors	Gets all of detectors. This is a paginated API. If you provide a null maxSizePerPage, this actions retrieves a maximum of 10 records per page. If you provide a maxSizePerPage, the value must be between 5 and 10. To get the next page results, provide the pagination token from the GetEventTypesResponse as part of your request. A null pagination token fetches the records from the beginning.	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetExternalModels	Gets the details for one or more Amazon SageMaker models that have been imported into the service. This is a paginated API. If you provide a null <code>maxSizePerPage</code> , this actions retrieves a maximum of 10 records per page. If you provide a <code>maxSizePerPage</code> , the value must be between 5 and 10. To get the next page results, provide the pagination token from the <code>GetExternalModelsResult</code> as part of your request. A null pagination token fetches the records from the beginning.	List			
GetModelVersion	Gets a model version.	List			
GetModels	Gets all of the models for the AWS account, or the specified model type, or gets a single model for the specified model type, model ID combination.	List			
GetOutcomes	Gets one or more outcomes. This is a paginated API. If you provide a null <code>maxSizePerPage</code> , this actions retrieves a maximum of 10 records per page. If you provide a <code>maxSizePerPage</code> , the value must be between 50 and 100. To get the next page results, provide the pagination token from the <code>GetOutcomesResult</code> as part of your request. A null pagination token fetches the records from the beginning.	List			
GetPrediction	Evaluates an event against a detector version. If a version ID is not provided, the detector's (ACTIVE) version is used.	Read			
GetRules	Gets all rules available for the specified detector.	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetVariables	Gets all of the variables or the specific variable. This is a paginated API. Providing null <code>maxSizePerPage</code> results in retrieving maximum of 100 records per page. If you provide <code>maxSizePerPage</code> the value must be between 50 and 100. To get the next page result, provide a pagination token from <code>GetVariablesResult</code> as part of your request. Null pagination token fetches the records from the beginning.	List			
PutDetector	Creates or updates a detector.	Write			
PutExternalModel	Creates or updates an Amazon SageMaker model endpoint. You can also use this action to update the configuration of the model endpoint, including the IAM role and/or the mapped variables.	Write			
PutModel	Creates or updates a model.	Write			iam:PassRole
PutOutcome	Creates or updates an outcome.	Write			
UpdateDetectorVersion	Updates a detector version. The <code>detector</code> version attributes that you can update include models, external model endpoints, rules, and description. You can only update a DRAFT detector version.	Write			
UpdateDetectorVersionMetadata	Updates the detector version's <code>description</code> . You can update the metadata for any detector version (DRAFT, ACTIVE, or INACTIVE).	Write			
UpdateDetectorVersionStatus	Updates the detector version's <code>status</code> . You can perform the following promotions or demotions using <code>UpdateDetectorVersionStatus</code> : DRAFT to ACTIVE, ACTIVE to INACTIVE, and INACTIVE to ACTIVE.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UpdateModelVersion	Updates a model version. You can update the description and status attributes using this action.	Write			
UpdateRuleMetadata	Updates a rule's metadata.	Write			
UpdateRuleVersion	Updates a rule version resulting in a new rule version.	Write			
UpdateVariable	Updates a variable.	Write			

Resource Types Defined by Amazon Fraud Detector

Amazon Fraud Detector does not support specifying a resource ARN in the `Resource` element of an IAM policy statement. To allow access to Amazon Fraud Detector, specify “`Resource`”: “`*`” in your policy.

Condition Keys for Amazon Fraud Detector

Fraud Detector has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Amazon FreeRTOS

Amazon FreeRTOS (service prefix: `freertos`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon FreeRTOS \(p. 1176\)](#)
- [Resource Types Defined by Amazon FreeRTOS \(p. 1177\)](#)
- [Condition Keys for Amazon FreeRTOS \(p. 1178\)](#)

Actions Defined by Amazon FreeRTOS

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources (“`*`”) in the `Resource` element of your

policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateSoftwareConfiguration	Creates a software configuration.	Write	configuration* (p. 1178)		
				aws:RequestTag/ \${TagKey} (p. 1178)	
				aws:TagKeys (p. 1178)	
DeleteSoftwareConfiguration	Deletes the software configuration.	Write	configuration* (p. 1178)		
DescribeHardwarePlatform	Describes the hardware platform.	Read			
DescribeSoftwareConfiguration	Describes the software configuration.	Read	configuration* (p. 1178)		
GetSoftwareURL	Get the URL for Amazon FreeRTOS software download.	Read			
GetSoftwareURLForRTOSVersion	Get the URL for Amazon FreeRTOS software download based on the configuration.	Read			
ListFreeRTOSVersions	Lists versions of AmazonFreeRTOS.	List			
ListHardwarePlatforms	Lists the hardware platforms.	List			
ListHardwareVendors	Lists the hardware vendors.	List			
ListSoftwareConfigurations	Lists the software configurations.	List			
UpdateSoftwareConfiguration	Updates the software configuration.	Write	configuration* (p. 1178)		

Resource Types Defined by Amazon FreeRTOS

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 1176\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you

can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
configuration	arn:\${Partition}:freertos:\${Region}:\${Account}:configuration/\${configurationName}	aws:ResourceTag/\${TagKey} (p. 1178)

Condition Keys for Amazon FreeRTOS

Amazon FreeRTOS defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	A tag key that is present in the request that the user makes to Amazon FreeRTOS.	String
aws:ResourceTag/\${TagKey}	The tag key component of a tag attached to an Amazon FreeRTOS resource.	String
aws:TagKeys	The list of all the tag key names associated with the resource in the request.	String

Actions, Resources, and Condition Keys for Amazon FSx

Amazon FSx (service prefix: `fsx`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon FSx \(p. 1178\)](#)
- [Resource Types Defined by Amazon FSx \(p. 1181\)](#)
- [Condition Keys for Amazon FSx \(p. 1182\)](#)

Actions Defined by Amazon FSx

You can specify the following actions in the Action element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CancelDataRepositoryTask	This action cancels a data repository task	Write	task* (p. 1182)		
CreateBackup	This action creates a new backup.	Tagging	backup* (p. 1182)		
			file-system* (p. 1181)		
			aws:RequestTag/\${TagKey} (p. 1182) aws:TagKeys (p. 1182)		
CreateDataRepositoryTask	This action creates a new task.	Tagging	file-system* (p. 1181)		
			task* (p. 1182)		
			aws:RequestTag/\${TagKey} (p. 1182) aws:TagKeys (p. 1182)		
CreateFileSystem	This action creates a new, empty, Amazon FSx file system	Tagging	file-system* (p. 1181)		
			aws:RequestTag/\${TagKey} (p. 1182) aws:TagKeys (p. 1182)		
CreateFileSystem	This action creates a new Amazon FSx file system from an existing backup.	Tagging	backup* (p. 1182)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			file-system* (p. 1181)		
DeleteBackup	This action deletes a backup, deleting its contents. After deletion, the backup no longer exists, and its data is gone.	Write	backup* (p. 1182)		
DeleteFileSystem	This action deletes a file system, deleting its contents.	Write	file-system* (p. 1181)		
DescribeBackups	This action returns the description of specific Amazon FSx backups, if one or more BackupIds are provided for that backup. Otherwise, it returns all backups owned by your AWS account in the AWS Region of the endpoint that you're calling.	Read			
DescribeDataRepositoryTasks	This action returns the descriptions of specific Amazon FSx data repository task, if one or more TaskIds are provided for that data repository task. Otherwise, it returns all data repository task owned by your AWS account in the AWS Region of the endpoint that you're calling.	Read			
DescribeFileSystems	This action returns the description of specific Amazon FSx file systems, if a FileSystemIds value is provided for that file system. Otherwise, it returns descriptions of all file systems owned by your AWS account in the AWS Region of the endpoint that you're calling.	Read			
ListTagsForResource	This action lists tags for an Amazon FSx resource.	Read	backup (p. 1182)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			file-system (p. 1181)		
			task (p. 1182)		
TagResource	This action tags an Amazon FSx resource.	Tagging	backup (p. 1182)		
			file-system (p. 1181)		
			task (p. 1182)		
				aws:TagKeys (p. 1182)	
				aws:RequestTag/ {\$TagKey} (p. 1182)	
UntagResource	This action removes a tag from an Amazon FSx resource.	Tagging	backup (p. 1182)		
			file-system (p. 1181)		
			task (p. 1182)		
				aws:TagKeys (p. 1182)	
UpdateFileSystem	This action updates file system configuration.	Write	file-system* (p. 1181)		

Resource Types Defined by Amazon FSx

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 1178\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
file-system	arn:\${Partition}:fsx:\${Region}: \${Account}:file-system/*	aws:ResourceTag/ {\$TagKey} (p. 1182)

Resource Types	ARN	Condition Keys
backup	arn:\${Partition}:fsx:\${Region}:\${Account}:backup/*	aws:ResourceTag/\${TagKey} (p. 1182)
task	arn:\${Partition}:fsx:\${Region}:\${Account}:task/*	aws:ResourceTag/\${TagKey} (p. 1182)

Condition Keys for Amazon FSx

Amazon FSx defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}		String
aws:ResourceTag/\${TagKey}		String
aws:TagKeys		String

Actions, Resources, and Condition Keys for Amazon GameLift

Amazon GameLift (service prefix: `gamelift`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon GameLift \(p. 1182\)](#)
- [Resource Types Defined by Amazon GameLift \(p. 1189\)](#)
- [Condition Keys for Amazon GameLift \(p. 1190\)](#)

Actions Defined by Amazon GameLift

You can specify the following actions in the Action element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the Resource element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AcceptMatch	Registers player acceptance or rejection of a proposed FlexMatch match.	Write			
CreateAlias	Defines a new alias for a fleet.	Write		aws:RequestTag/\${TagKey} (p. 1190) aws:TagKeys (p. 1190)	
CreateBuild	Creates a new game build using files stored in an Amazon S3 bucket.	Write		aws:RequestTag/\${TagKey} (p. 1190) aws:TagKeys (p. 1190)	
CreateFleet	Creates a new fleet of computing resources to run your game servers.	Write		aws:RequestTag/\${TagKey} (p. 1190) aws:TagKeys (p. 1190)	
CreateGameSession	Starts a new game session on a specified fleet.	Write			
CreateGameSessionQueue	Sets up a new queue for processing new game session placement requests.	Write		aws:RequestTag/\${TagKey} (p. 1190) aws:TagKeys (p. 1190)	
CreateMatchmakingConfiguration	Creates a new FlexMatch matchmaking configuration.	Write		aws:RequestTag/\${TagKey} (p. 1190) aws:TagKeys (p. 1190)	
CreateMatchmakingRuleSet	Creates a new matchmaking rule set for FlexMatch.	Write		aws:RequestTag/\${TagKey} (p. 1190)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:TagKeys (p. 1190)	
CreatePlayerSession	Reserves an available game session slot for a player.	Write			
CreatePlayerSessions	Reserves available game session slots for multiple players.	Write			
CreateScript	Creates a new Realtime Servers script.	Write		aws:RequestTag/\${TagKey} (p. 1190) aws:TagKeys (p. 1190)	
CreateVpcPeeringAuthorization	Allows GameLift to create or deauthorize connection between a GameLift fleet VPC and a VPC on another AWS account.	Write			
CreateVpcPeeringConnection	Establishes a peering connection between your GameLift fleet VPC and a VPC on another account.	Write			
DeleteAlias	Deletes an alias.	Write	alias* (p. 1189)		
DeleteBuild	Deletes a game build.	Write	build* (p. 1189)		
DeleteFleet	Deletes an empty fleet.	Write	fleet* (p. 1189)		
DeleteGameSessionQueue	Deletes an existing game session queue.	Write	gameSessionQueue* (p. 1189)		
DeleteMatchmakingConfiguration	Deletes an existing FlexMatch matchmaking configuration.	Write	matchmakingConfiguration* (p. 1189)		
DeleteMatchmakingRuleSet	Deletes an existing FlexMatch matchmaking rule set.	Write	matchmakingRuleSet* (p. 1189)		
DeleteScalingPolicy	Deletes a set of auto-scaling rules.	Write	fleet* (p. 1189)		
DeleteScript	Deletes a Realtime Servers script.	Write	script* (p. 1189)		
DeleteVpcPeeringAuthorization	Cancels a VPC peering authorization .	Write			
DeleteVpcPeeringConnection	Removes a peering connection between VPCs.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeAlias	Retrieves properties for an alias.	Read	alias* (p. 1189)		
DescribeBuild	Retrieves properties for a game build.	Read	build* (p. 1189)		
DescribeEC2Instances	Retrieves the maximum allowed and current usage for EC2 instance types.	Read			
DescribeFleetAttributes	Retrieves general properties, including status, for fleets.	Read			
DescribeFleetCapacity	Retrieves the current capacity setting for fleets.	Read			
DescribeFleetEvents	Retrieves entries from a fleet's event log.	Read	fleet* (p. 1189)		
DescribeFleetPortConnectors	Retrieves the inbound connection permissions for a fleet.	Read	fleet* (p. 1189)		
DescribeFleetUtilization	Retrieves utilization statistics for fleets.	Read			
DescribeGameSessions	Retrieves properties for game sessions in a fleet, including the protection policy.	Read			
DescribeGameSessionPlacement	Retrieves details of a game session placement request.	Read			
DescribeGameSessionQueues	Retrieves properties for game session queues.	Read			
DescribeGameSessions	Retrieves properties for game sessions in a fleet.	Read			
DescribeInstances	Retrieves information about instances in a fleet.	Read	fleet* (p. 1189)		
DescribeMatchmaking	Retrieves details of matchmaking tickets.	Read			
DescribeMatchmakers	Retrieves properties for FlexMatchmakers.	Read			
DescribeMatchmakingRuleSets	Retrieves properties for FlexMatchmaking rule sets.	Read			
DescribePlayerSessions	Retrieves properties for player sessions in a game session.	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeRuntimeConfiguration	Retrieves the current runtime configuration for a fleet.	Read	fleet* (p. 1189)		
DescribeScalingPolicies	Retrieves all scaling policies that are applied to a fleet.	Read	fleet* (p. 1189)		
DescribeScript	Retrieves properties for a Realtime Servers script.	Read	script* (p. 1189)		
DescribeVpcPeeringAuthorizations	Retrieves valid VPC peering connections.	Read			
DescribeVpcPeeringConnections	Retrieves details on active or pending VPC peering connections.	Read			
GetGameSessionLogs	Retrieves the location of stored logs for a game session.	Read			
GetInstanceAccess	Requests remote access to a specified fleet instance.	Read	fleet* (p. 1189)		
ListAliases	Retrieves all aliases that are defined in the current region.	List			
ListBuilds	Retrieves all game build in the current region.	List			
ListFleets	Retrieves a list of fleet IDs for all fleets in the current region.	List			
ListScripts	Retrieves properties for all Realtime Servers scripts in the current region.	List			
ListTagsForResource	List Tags for GameLift Resource	List	alias (p. 1189) build (p. 1189) fleet (p. 1189) gameSessionQueue (p. 1189) matchmakingConfiguration (p. 1189) matchmakingRuleSet (p. 1189) script (p. 1189)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
PutScalingPolicy	Creates or updates a fleet auto-scaling policy.	Write	fleet* (p. 1189)		
RequestUploadCredentials	Retrieves fresh upload credentials to use when uploading a new game build.	Read	build* (p. 1189)		
ResolveAlias	Retrieves the fleet ID associated with an alias.	Read	alias* (p. 1189)		
SearchGameSessions	Retrieves game sessions that match a set of search criteria.	Read			
StartFleetActions	Resumes auto-scaling activity on a fleet after it was suspended with <code>StopFleetActions()</code> .	Write	fleet* (p. 1189)		
StartGameSessionPlacement	Sends a game session placement request to a game session queue.	Write	gameSessionQueue* (p. 1189)		
StartMatchBackfill	Requests FlexMatch matchmaking to fill available player slots in an existing game session.	Write			
StartMatchmaking	Requests FlexMatch matchmaking for one or a group of players and game session placement for a resulting match.	Write			
StopFleetActions	Suspends auto-scaling activity on a fleet.	Write	fleet* (p. 1189)		
StopGameSessionPlacement	Cancels a game session placement request that is in progress.	Write			
StopMatchmaking	Cancels a matchmaking or match backfill request that is in progress.	Write			
TagResource	Tags GameLift Resources	Tagging	alias (p. 1189)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			matchmakingRuleSet (p. 1189)		
			script (p. 1189)		
				aws:RequestTag/\${TagKey} (p. 1190)	
	Untagging GameLift Resources	Tagging	alias (p. 1189)		
			build (p. 1189)		
			fleet (p. 1189)		
			gameSessionQueue (p. 1189)		
			matchmakingConfiguration (p. 1189)		
			matchmakingRuleSet (p. 1189)		
			script (p. 1189)		
				aws:TagKeys (p. 1190)	
UpdateAlias	Updates the properties of an existing alias.	Write	alias* (p. 1189)		
UpdateBuild	Updates an existing build's metadata.	Write	build* (p. 1189)		
UpdateFleetAttributes	Updates the general properties of an existing fleet.	Write	fleet* (p. 1189)		
UpdateFleetCapacity	Adjusts a fleet's capacity settings.	Write	fleet* (p. 1189)		
UpdateFleetPortSettings	Adjusts a fleet's port settings.	Write	fleet* (p. 1189)		
UpdateGameSession	Updates the properties of an existing game session.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UpdateGameSessionQueue	Updates properties of an existing game session queue.	Write	gameSessionQueue* (p. 1189)		
UpdateMatchmakingConfiguration	Updates properties of an existing Flex Matchmaking configuration.	Write	matchmakingConfiguration* (p. 1189)		
UpdateRuntimeConfiguration	Updates how server processes configuration on instances in an existing fleet.	Write	fleet* (p. 1189)		
UpdateScript	Updates the metadata and content of an existing Realtime Servers script.	Write	script* (p. 1189)		
ValidateMatchmakingRuleSet	Validates the syntax of a Flex Matchmaking rule set.	Read			

Resource Types Defined by Amazon GameLift

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1182\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
alias	<code>arn:\${Partition}:gamelift:\${Region}::alias/\${AliasId}</code>	aws:ResourceTag/\${TagKey} (p. 1190)
build	<code>arn:\${Partition}:gamelift:\${Region}://\${AccountId}:build/\${BuildId}</code>	aws:ResourceTag/\${TagKey} (p. 1190)
script	<code>arn:\${Partition}:gamelift:\${Region}://\${AccountId}:script/\${ScriptId}</code>	aws:ResourceTag/\${TagKey} (p. 1190)
fleet	<code>arn:\${Partition}:gamelift:\${Region}://\${Account}:fleet/\${FleetId}</code>	aws:ResourceTag/\${TagKey} (p. 1190)
gameSessionQueue	<code>arn:\${Partition}:gamelift:\${Region}://\${Account}:gamesessionqueue/\${GameSessionQueueName}</code>	aws:ResourceTag/\${TagKey} (p. 1190)
matchmakingConfiguration	<code>arn:\${Partition}:gamelift:\${Region}://\${Account}:matchmakingconfiguration/\${MatchmakingConfigurationName}</code>	aws:ResourceTag/\${TagKey} (p. 1190)
matchmakingRuleSet	<code>arn:\${Partition}:gamelift:\${Region}://\${Account}:matchmakingruleset/\${MatchmakingRuleSetName}</code>	aws:ResourceTag/\${TagKey} (p. 1190)

Condition Keys for Amazon GameLift

Amazon GameLift defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
<code>aws:RequestTag/\${TagKey}</code>	Filters actions based on the tags that are passed in the request	String
<code>aws:ResourceTag/\${TagKey}</code>	Filters actions based on the tags associated with the resource	String
<code>aws:TagKeys</code>	Filters actions based on the tag keys that are passed in the request	String

Actions, Resources, and Condition Keys for Amazon Glacier

Amazon Glacier (service prefix: `glacier`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Glacier \(p. 1190\)](#)
- [Resource Types Defined by Amazon Glacier \(p. 1193\)](#)
- [Condition Keys for Amazon Glacier \(p. 1193\)](#)

Actions Defined by Amazon Glacier

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AbortMultipartUpload	Aborts a multipart upload identified by the upload ID	Write	vault* (p. 1193)		
AbortVaultLock	Aborts the vault locking process if the vault lock is not in the Locked state	Permissions management	vault* (p. 1193)		
AddTagsToVault	Adds the specified tags to a vault	Tagging	vault* (p. 1193)		
CompleteMultipartUpload	Completes a multipart upload process	Write	vault* (p. 1193)		
CompleteVaultLock	Completes the vault locking process	Permissions management	vault* (p. 1193)		
CreateVault	Creates a new vault with the specified name	Write	vault* (p. 1193)		
DeleteArchive	Deletes an archive from a vault	Write	vault* (p. 1193)		
				glacier:ArchiveAgeInDays (p. 1193)	
DeleteVault	Deletes a vault	Write	vault* (p. 1193)		
DeleteVaultAccessPolicy	Deletes the access policy associated with the specified vault	Permissions management	vault* (p. 1193)		
DeleteVaultNotificationConfiguration	Deletes the notification configuration set for a vault	Write	vault* (p. 1193)		
DescribeJob	Returns information about a job you previously initiated	Read	vault* (p. 1193)		
DescribeVault	Returns information about a vault	Read	vault* (p. 1193)		
GetDataRetrievalPolicy	Returns the current data retrieval policy for the account and region specified in the GET request	Read			
GetJobOutput	Downloads the output of the job you initiated	Read	vault* (p. 1193)		
GetVaultAccessPolicy	Retrieves the access-policy subresource set on the vault	Read	vault* (p. 1193)		
GetVaultLock	Retrieves attributes from the lock-policy subresource set on the specified vault	Read	vault* (p. 1193)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetVaultNotifications	Retrieves the notification-configuration subresource set on the vault	Read	vault* (p. 1193)		
InitiateJob	Initiates a job of the specified type	Write	vault* (p. 1193)		
					glacier:ArchiveAgeInDays (p. 1193)
InitiateMultipartUpload	Initiates a multipart upload	Write	vault* (p. 1193)		
InitiateVaultLock	Initiates the vault locking process	Permissions management	vault* (p. 1193)		
ListJobs	Lists jobs for a vault that are in-progress and jobs that have recently finished	List	vault* (p. 1193)		
ListMultipartUploads	Lists in-progress multipart uploads for the specified vault	List	vault* (p. 1193)		
ListParts	Lists the parts of an archive that have been uploaded in a specific multipart upload	List	vault* (p. 1193)		
ListProvisionedCapacity	This operation lists the provisioned capacity for the specified AWS account.	List			
ListTagsForVault	Lists all the tags attached to a vault	List	vault* (p. 1193)		
ListVaults	Lists all vaults	List			
PurchaseProvisionedCapacity	This operation purchases a provisioned capacity unit for an AWS account.	Write			
RemoveTagsFromVault	Removes one or more tags from the set of tags attached to a vault	Tagging	vault* (p. 1193)		
SetDataRetrievalPolicy	Sets and then enacts a data retrieval policy in the region specified in the PUT request	Permissions management			
SetVaultAccessPolicy	Configures an access policy for a vault and will overwrite an existing policy	Permissions management	vault* (p. 1193)		
SetVaultNotifications	Configures vault notifications	Write	vault* (p. 1193)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UploadArchive	Adds an archive to a vault	Write	vault* (p. 1193)		
UploadMultipartPart	Uploads a part of an archive	Write	vault* (p. 1193)		

Resource Types Defined by Amazon Glacier

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 1190\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
vault	arn:\${Partition}:glacier:\${Region}: \${Account}:vaults/\${VaultName}	

Condition Keys for Amazon Glacier

Amazon Glacier defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
glacier:ArchiveAgeInDays	How long an archive has been stored in the vault, in days.	String
glacier:ResourceTag/	A customer-defined tag.	String

Actions, Resources, and Condition Keys for AWS Global Accelerator

AWS Global Accelerator (service prefix: `globalaccelerator`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).

- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Global Accelerator \(p. 1194\)](#)
- [Resource Types Defined by AWS Global Accelerator \(p. 1195\)](#)
- [Condition Keys for AWS Global Accelerator \(p. 1195\)](#)

Actions Defined by AWS Global Accelerator

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateAccelerator	Create an accelerator.	Write			
CreateEndpointGroup	Add an endpoint group.	Write	listener* (p. 1195)		
CreateListener	Add a listener.	Write	accelerator* (p. 1195)		
DeleteAccelerator	Delete the accelerator.	Write	accelerator* (p. 1195)		
DeleteEndpointGroup	Delete the endpoint group.	Write	endpointgroup* (p. 1195)		
DeleteListener	Delete the listener.	Write	listener* (p. 1195)		
DescribeAccelerator	Describe the accelerator.	Read	accelerator* (p. 1195)		
DescribeAcceleratorAttributes	Describe the accelerator attributes	Read	accelerator* (p. 1195)		
DescribeEndpointGroup	Describe the endpoint group.	Read	endpointgroup* (p. 1195)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeListener	Describe the listener.	Read	listener* (p. 1195)		
ListAccelerators	List the accelerators.	List			
ListEndpointGroups	List the endpoint groups.	List	listener* (p. 1195)		
ListListeners	List the listeners.	List	accelerator* (p. 1195)		
UpdateAccelerator	Update the accelerator.	Write	accelerator* (p. 1195)		
UpdateAcceleratorAttributes	Update the accelerator attributes	Write	accelerator* (p. 1195)		
UpdateEndpointGroup	Update the endpoint group.	Write	endpointgroup* (p. 1195)		
UpdateListener	Update the listener.	Write	listener* (p. 1195)		

Resource Types Defined by AWS Global Accelerator

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions Table \(p. 1194\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
accelerator	<code>arn:\${Partition}:globalaccelerator::\${Account}:accelerator/\${AcceleratorId}</code>	
listener	<code>arn:\${Partition}:globalaccelerator::\${Account}:accelerator/\${AcceleratorId}/listener/\${ListenerId}</code>	
endpointgroup	<code>arn:\${Partition}:globalaccelerator::\${Account}:accelerator/\${AcceleratorId}/listener/\${ListenerId}/endpoint-group/\${EndpointGroupId}</code>	

Condition Keys for AWS Global Accelerator

GlobalAccelerator has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for AWS Glue

AWS Glue (service prefix: `glue`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Glue \(p. 1196\)](#)
- [Resource Types Defined by AWS Glue \(p. 1204\)](#)
- [Condition Keys for AWS Glue \(p. 1205\)](#)

Actions Defined by AWS Glue

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
BatchCreatePartition	Grants permission to create one or more partitions	Write			
BatchDeleteConnection	Grants permission to delete one or more connections	Write			
BatchDeletePartition	Grants permission to delete one or more partitions	Write			
BatchDeleteTable	Grants permission to delete one or more tables	Write			
BatchDeleteTableVersion	Grants permission to delete one or more versions of a table	Read			
BatchGetCrawlers	Grants permission to retrieve one or more crawlers	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
BatchGetDevEndpoints	Grants permission to retrieve one or more development endpoints	Read			
BatchGetJobs	Grants permission to retrieve one or more jobs	Read			
BatchGetPartitions	Grants permission to retrieve one or more partitions	Read			
BatchGetTriggers	Grants permission to retrieve one or more triggers	Read			
BatchGetWorkflows	Grants permission to retrieve one or more workflows	Read			
BatchStopJobRun	Grants permission to stop one or more job runs for a job	Write			
CancelMLTaskRun	Grants permission to stop a running ML Task Run	Write			
CreateClassifier	Grants permission to create a classifier	Write			
CreateConnection	Grants permission to create a connection	Write			
CreateCrawler	Grants permission to create a crawler	Write		aws:RequestTag/\${TagKey} (p. 1205) aws:TagKeys (p. 1205)	
CreateDatabase	Grants permission to create a database	Write			
CreateDevEndpoint	Grants permission to create a development endpoint	Write		aws:RequestTag/\${TagKey} (p. 1205) aws:TagKeys (p. 1205)	
CreateJob	Grants permission to create a job	Write		aws:RequestTag/\${TagKey} (p. 1205) aws:TagKeys (p. 1205)	
CreateMLTransform	Grants permission to create an ML Transform	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreatePartition	Grants permission to create a partition	Write			
CreateScript	Grants permission to create a script	Write			
CreateSecurityConfiguration	Grants permission to create a security configuration	Write			
 CreateTable	Grants permission to create a table	Write			
CreateTrigger	Grants permission to create a trigger	Write		aws:RequestTag/\${TagKey} (p. 1205) aws:TagKeys (p. 1205)	
CreateUserDefinedFunction	Grants permission to create a function definition	Write			
CreateWorkflow	Grants permission to create a workflow	Write		aws:RequestTag/\${TagKey} (p. 1205) aws:TagKeys (p. 1205)	
DeleteClassifier	Grants permission to delete a classifier	Write			
DeleteConnection	Grants permission to delete a connection	Write			
DeleteCrawler	Grants permission to delete a crawler	Write			
DeleteDatabase	Grants permission to delete a database	Write			
DeleteDevEndpoint	Grants permission to delete a development endpoint	Write			
DeleteJob	Grants permission to delete a job	Write			
DeleteMLTransform	Grants permission to delete an ML Transform	Write			
DeletePartition	Grants permission to delete a partition	Write			
DeleteResourcePolicy	Grants permission to delete a resource policy	Write	catalog* (p. 1204)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteSecurityConfigurations	Grants permission to delete a security configuration	Write			
DeleteTable	Grants permission to delete a table	Write			
DeleteTableVersions	Grants permission to delete a version of a table	Read			
DeleteTrigger	Grants permission to delete a trigger	Write			
DeleteUserDefinedFunctions	Grants permission to delete a function definition	Write			
DeleteWorkflow	Grants permission to delete a workflow	Write			
GetCatalogImportStatus	Grants permission to retrieve the Catalog import status	Read			
GetClassifier	Grants permission to retrieve a classifier	Read			
GetClassifiers	Grants permission to list all classifiers	Read			
GetConnection	Grants permission to retrieve a connection	Read			
GetConnections	Grants permission to retrieve a list of connections	Read			
GetCrawler	Grants permission to retrieve a crawler	Read			
GetCrawlerMetrics	Grants permission to retrieve metrics about crawlers	Read			
GetCrawlers	Grants permission to retrieve all crawlers	Read			
GetDataCatalogEncryptionSettings	Grants permission to retrieve Catalog encryption settings	Read			
GetDatabase	Grants permission to retrieve a database	Read			
GetDatabases	Grants permission to retrieve all databases	Read			
GetDataflowGraph	Grants permission to transform a script into a directed acyclic graph (DAG)	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetDevEndpoint	Grants permission to retrieve a development endpoint	Read			
GetDevEndpoints	Grants permission to retrieve all development endpoints	Read			
GetJob	Grants permission to retrieve a job	Read			
GetJobBookmark	Grants permission to retrieve a job bookmark	Read			
GetJobRun	Grants permission to retrieve a job run	Read			
GetJobRuns	Grants permission to retrieve all job runs of a job	Read			
GetJobs	Grants permission to retrieve all current jobs	Read			
GetMLTaskRun	Grants permission to retrieve an ML Task Run	Read			
GetMLTaskRuns	Grants permission to retrieve all ML Task Runs	List			
GetMLTransform	Grants permission to retrieve an ML Transform	Read			
GetMLTransforms	Grants permission to retrieve all ML Transforms	List			
GetMapping	Grants permission to create a mapping	Write			
GetPartition	Grants permission to retrieve a partition	Read			
GetPartitions	Grants permission to retrieve the partitions of a table	Read			
GetPlan	Grants permission to retrieve a mapping for a script	Read			
GetResourcePolicy	Grants permission to retrieve a resource policy	Read	catalog* (p. 1204)		
GetSecurityConfig	Grants permission to retrieve a security configuration	Read			
GetSecurityConfig	Grants permission to retrieve one or more security configurations	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetTable	Grants permission to retrieve a table	Read			
GetTableVersion	Grants permission to retrieve a version of a table	Read			
GetTableVersions	Grants permission to retrieve a list of versions of a table	Read			
GetTables	Grants permission to retrieve the tables in a database	Read			
GetTags	Grants permission to retrieve all tags associated with a resource	Read	crawler (p. 1205)		
			devendpoint (p. 1205)		
			job (p. 1205)		
			trigger (p. 1205)		
			workflow (p. 1205)		
GetTrigger	Grants permission to retrieve a trigger	Read			
GetTriggers	Grants permission to retrieve the triggers associated with a job	Read			
 GetUserDefinedFunction	Grants permission to retrieve a function definition.	Read			
 GetUserDefinedFunctions	Grants permission to retrieve multiple function definitions	Read			
GetWorkflow	Grants permission to retrieve a workflow	Read			
GetWorkflowRun	Grants permission to retrieve a workflow run	Read			
GetWorkflowRunProperties	Grants permission to retrieve workflow run properties	Read			
GetWorkflowRuns	Grants permission to retrieve all runs of a workflow	Read			
ImportCatalogToAthena	Grants permission to import an Athena data catalog into AWS Glue	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListCrawlers	Grants permission to retrieve all crawlers	List			
ListDevEndpoints	Grants permission to retrieve all development endpoints	List			
ListJobs	Grants permission to retrieve all current jobs	List			
ListTriggers	Grants permission to retrieve all triggers	List			
ListWorkflows	Grants permission to retrieve all workflows	List			
PutDataCatalogEncryptionOption	Grants permission to update catalog encryption settings	Write			
PutResourcePolicy	Grants permission to update a resource policy	Write	catalog* (p. 1204)		
PutWorkflowRunProperties	Grants permission to update workflow run properties	Write			
ResetJobBookmark	Grants permission to reset a job bookmark	Write			
SearchTables	Grants permission to retrieve the tables in the catalog	Read			
StartCrawler	Grants permission to start a crawler	Write			
StartCrawlerSchedule	Grants permission to change the schedule state of a crawler to SCHEDULED	Write			
StartExportLabels	Grants permission to start an Export Labels ML Task Run	Write			
StartImportLabels	Grants permission to start an Import Labels ML Task Run	Write			
StartJobRun	Grants permission to start running a job	Write			
StartMLEvaluation	Grants permission to start an Evaluation ML Task Run	Write			
StartMLLabeling	Grants permission to start a Labeling Set Generation ML Task Run	Write			
StartTrigger	Grants permission to start a trigger	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
StartWorkflowRun	Grants permission to start running a workflow	Write			
StopCrawler	Grants permission to stop a running crawler	Write			
StopCrawlerSchedule	Grants permission to set the schedule state of a crawler to NOT_SCHEDULED	Write			
StopTrigger	Grants permission to stop a trigger	Write			
TagResource	Grants permission to add tags to a resource	Tagging	crawler (p. 1205)		
devendpoint (p. 1205)					
job (p. 1205)					
trigger (p. 1205)					
workflow (p. 1205)					
	aws:TagKeys (p. 1205)				
	aws:RequestTag/\${TagKey} (p. 1205)				
UntagResource	Grants permission to remove tags associated with a resource	Tagging	crawler (p. 1205)		
devendpoint (p. 1205)					
job (p. 1205)					
trigger (p. 1205)					
workflow (p. 1205)					
	aws:TagKeys (p. 1205)				
UpdateClassifier	Grants permission to update a classifier	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UpdateConnection	Grants permission to update a connection	Write			
UpdateCrawler	Grants permission to update a crawler	Write			
UpdateCrawlerSchedule	Grants permission to update the schedule of a crawler	Write			
UpdateDatabase	Grants permission to update a database	Write			
UpdateDevEndpoint	Grants permission to update a development endpoint	Write			
UpdateJob	Grants permission to update a job	Write			
UpdateMLTransform	Grants permission to update an ML Transform	Write			
UpdatePartition	Grants permission to update a partition	Write			
UpdateTable	Grants permission to update a table	Write			
UpdateTrigger	Grants permission to update a trigger	Write			
UpdateUserDefinedFunction	Grants permission to update a function definition	Write			
UpdateWorkflow	Grants permission to update a workflow	Write			
UseMLTransforms	Grants permission to use an ML Transform from within a Glue ETL Script	Write			

Resource Types Defined by AWS Glue

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 1196\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
catalog	arn:\${Partition}:glue:\${Region}:\${Account}:catalog/\${CatalogName}	

Resource Types	ARN	Condition Keys
database	arn:\${Partition}:glue:\${Region}: \${Account}:database/\${DatabaseName}	
table	arn:\${Partition}:glue:\${Region}: \${Account}:table/\${TableName}	
partition	arn:\${Partition}:glue:\${Region}: \${Account}:partition/\${PartitionName}	
tableversion	arn:\${Partition}:glue:\${Region}: \${Account}:tableVersion/\${TableVersionName}	
connection	arn:\${Partition}:glue:\${Region}: \${Account}:connection/\${ConnectionName}	
userdefinedfunction	arn:\${Partition}:glue:\${Region}: \${Account}:userDefinedFunction/ \${UserDefinedFunctionName}	
devendpoint	arn:\${Partition}:glue:\${Region}: \${Account}:devendpoint/\${DevEndpointName}	aws:ResourceTag/ \${TagKey} (p. 1205)
job	arn:\${Partition}:glue:\${Region}: \${Account}:job/\${JobName}	aws:ResourceTag/ \${TagKey} (p. 1205)
trigger	arn:\${Partition}:glue:\${Region}: \${Account}:trigger/\${TriggerName}	aws:ResourceTag/ \${TagKey} (p. 1205)
crawler	arn:\${Partition}:glue:\${Region}: \${Account}:crawler/\${CrawlerName}	aws:ResourceTag/ \${TagKey} (p. 1205)
workflow	arn:\${Partition}:glue:\${Region}: \${Account}:workflow/\${WorkflowName}	aws:ResourceTag/ \${TagKey} (p. 1205)

Condition Keys for AWS Glue

AWS Glue defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/ \${TagKey}	Filters actions based on the presence of tag key-value pairs in the request	String
aws:ResourceTag/ \${TagKey}	Filters actions based on tag key-value pairs attached to the resource	String
aws:TagKeys	Filters actions based on the presence of tag keys in the request	String

Actions, Resources, and Condition Keys for AWS Ground Station

AWS Ground Station (service prefix: `groundstation`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Ground Station \(p. 1206\)](#)
- [Resource Types Defined by AWS Ground Station \(p. 1209\)](#)
- [Condition Keys for AWS Ground Station \(p. 1209\)](#)

Actions Defined by AWS Ground Station

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CancelContact	Grants permission to cancel a contact	Write	Contact* (p. 1209)		
CreateConfig	Grants permission to create a configuration	Write		aws:RequestTag/\${TagKey} (p. 1209) aws:TagKeys (p. 1210)	
CreateDataflowEndpointGroup	Grants permission to create a dataflow endpoint group	Write		aws:RequestTag/\${TagKey} (p. 1209) aws:TagKeys (p. 1210)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateMissionProfile	Grants permission to create a mission profile	Write		aws:RequestTag/ \${TagKey} (p. 1209) aws:TagKeys (p. 1210)	
DeleteConfig	Grants permission to delete a config	Write	Config* (p. 1209)		
DeleteDataflowEndpointGroup	Grants permission to delete a dataflow endpoint group	Write	DataflowEndpointGroup* (p. 1209)		
DeleteMissionProfile	Grants permission to delete a mission profile	Write	MissionProfile* (p. 1209)		
DescribeContact	Grants permission to describe a contact	Read	Contact* (p. 1209)		
GetConfig	Grants permission to return a configuration	Read	Config* (p. 1209)		
GetDataflowEndpointGroup	Grants permission to return a dataflow endpoint group	Read	DataflowEndpointGroup* (p. 1209)		
GetMinuteUsage	Grants permission to return minutes usage	Read			
GetMissionProfile	Grants permission to retrieve a mission profile	Read	MissionProfile* (p. 1209)		
GetSatellite	Grants permission to return information about a satellite	Read	Satellite* (p. 1209)		
ListConfigs	Grants permission to return a list of past configurations	List			
ListContacts	Grants permission to return a list of contacts	List			
ListDataflowEndpointGroups	Grants permission to list dataflow endpoint groups	List			
ListGroundStations	Grants permission to list ground stations	List			
ListMissionProfiles	Grants permission to return a list of mission profiles	List			
ListSatellites	Grants permission to list satellites	List			
ListTagsForResource	Grants permission to list tags for a resource	Read	Config (p. 1209)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			Contact (p. 1209)		
			DataflowEndpointGroup (p. 1209)		
			MissionProfile (p. 1209)		
ReserveContact	Grants permission to reserve a contact	Write		aws:RequestTag/\${TagKey} (p. 1209) aws:TagKeys (p. 1210)	
TagResource	Grants permission to assign a resource tag	Tagging	Config (p. 1209)		
Contact (p. 1209)					
DataflowEndpointGroup (p. 1209)					
MissionProfile (p. 1209)					
	aws:TagKeys (p. 1210) aws:RequestTag/\${TagKey} (p. 1209)				
UntagResource	Grants permission to deassign a resource tag	Tagging	Config (p. 1209)		
Contact (p. 1209)					
DataflowEndpointGroup (p. 1209)					
MissionProfile (p. 1209)					
	aws:TagKeys (p. 1210)				
UpdateConfig	Grants permission to update a configuration	Write	Config* (p. 1209)		
	Grants permission to update a mission profile	Write	MissionProfile* (p. 1209)		

Resource Types Defined by AWS Ground Station

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 1206\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
Config	arn:\${Partition}:groundstation:\${Region}: \${Account}:config/\${configType}/\${configId}	aws:ResourceTag/\${TagKey} (p. 1210) groundstation:configId (p. 1210) groundstation:configType (p. 1210)
Contact	arn:\${Partition}:groundstation:\${Region}: \${Account}:contact/\${contactId}	aws:ResourceTag/\${TagKey} (p. 1210) groundstation:contactId (p. 1210)
DataflowEndpointGroup	arn:\${Partition}:groundstation:\${Region}: \${Account}:dataflow-endpoint-group/ \${dataflowEndpointGroupId}	aws:ResourceTag/\${TagKey} (p. 1210) groundstation:dataflowEndpointGroup (p. 1210)
GroundStationResource	arn:\${Partition}:groundstation:\${Region}: \${Account}:groundstation:\${groundStationId}	groundstation:groundStationId (p. 1210)
MissionProfile	arn:\${Partition}:groundstation: \${Region}: \${Account}:mission-profile/ \${missionProfileId}	aws:ResourceTag/\${TagKey} (p. 1210) groundstation:missionProfileId (p. 1210)
Satellite	arn:\${Partition}:groundstation:\${Region}: \${Account}:satellite/\${satelliteId}	groundstation:satelliteId (p. 1210)

Condition Keys for AWS Ground Station

AWS Ground Station defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Filters access by a key that is present in the request the user makes to the Ground Station service.	String

Condition Keys	Description	Type
aws:ResourceTag/\${TagKey}	Filters access by a tag key and value pair.	String
aws:TagKeys	Filters access by the list of all the tag key names present in the request the user makes to the Ground Station service.	String
groundstation:configId	Filters access by the ID of a config	String
groundstation:configType	Filters access by the type of a config	String
groundstation:contactId	Filters access by the ID of a contact	String
groundstation:dataflowEndpointGroupId	Filters access by the ID of a dataflow endpoint group	String
groundstation:groundStationId	Filters access by the ID of a ground station	String
groundstation:missionProfileId	Filters access by the ID of a mission profile	String
groundstation:satelliteId	Filters access by the ID of a satellite	String

Actions, Resources, and Condition Keys for Amazon GroundTruth Labeling

Amazon GroundTruth Labeling (service prefix: `groundtruthlabeling`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- View a list of the [API operations available for this service](#).

Topics

- [Actions Defined by Amazon GroundTruth Labeling \(p. 1210\)](#)
- [Resource Types Defined by Amazon GroundTruth Labeling \(p. 1211\)](#)
- [Condition Keys for Amazon GroundTruth Labeling \(p. 1211\)](#)

Actions Defined by Amazon GroundTruth Labeling

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in

a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeConsoleJobs [permission only]	Get status of GroundTruthLabeling Jobs	Read			
ListDatasetObjects [permission only]	Paginated list api to list dataset objects in a manifest file	Read			
RunFilterOrSample [permission only]	Filter records from a manifest file using S3 select. Get Sample entries based on random sampling.	Write			
RunGenerateManifestFile [permission only]	List a S3 prefix and create manifest files from objects in there.	Write			

Resource Types Defined by Amazon GroundTruth Labeling

Amazon GroundTruth Labeling does not support specifying a resource ARN in the Resource element of an IAM policy statement. To allow access to Amazon GroundTruth Labeling, specify “Resource”: “*” in your policy.

Condition Keys for Amazon GroundTruth Labeling

GroundTruth Labeling has no service-specific context keys that can be used in the Condition element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Amazon GuardDuty

Amazon GuardDuty (service prefix: `guardduty`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon GuardDuty \(p. 1212\)](#)

- [Resource Types Defined by Amazon GuardDuty \(p. 1217\)](#)
- [Condition Keys for Amazon GuardDuty \(p. 1218\)](#)

Actions Defined by Amazon GuardDuty

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AcceptInvitation	Grants permission to accept invitations to become a GuardDuty member account.	Write	detector* (p. 1217)		
ArchiveFindings	Grants permission to archive GuardDuty findings.	Write	detector* (p. 1217)		
CreateDetector	Grants permission to create a detector.	Write		aws:RequestTag/\${TagKey} (p. 1218) aws:TagKeys (p. 1218)	
CreateFilter	Grants permission to create GuardDuty filters. A filter defines finding attributes and conditions used to filter findings.	Write	detector* (p. 1217)		
				aws:RequestTag/\${TagKey} (p. 1218) aws:TagKeys (p. 1218)	
CreateIPSet	Grants permission to create an IPSet.	Write	detector* (p. 1217)		
				aws:RequestTag/\${TagKey} (p. 1218) aws:TagKeys (p. 1218)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateMembers	Grants permission to create GuardDuty member accounts. The account used to create a member becomes the GuardDuty master account.	Write	detector* (p. 1217)		
CreatePublishingDestination	Grants permission to create a publishing destination.	Write	detector* (p. 1217)		s3:GetObject s3>ListBucket
CreateSampleFindings	Grants permission to create sample findings.	Write	detector* (p. 1217)		
CreateThreatIntelSet	Grants permission to create GuardDuty ThreatIntelSets. A ThreatIntelSet consists of known malicious IP addresses used by GuardDuty to generate findings.	Write	detector* (p. 1217)		
			aws:RequestTag/\${TagKey} (p. 1218)	aws:TagKeys (p. 1218)	
DeclineInvitation	Grants permission to decline invitations to become a GuardDuty member account.	Write			
DeleteDetector	Grants permission to delete GuardDuty detectors.	Write	detector* (p. 1217)		
DeleteFilter	Grants permission to delete GuardDuty filters.	Write	detector* (p. 1217)		
			filter* (p. 1217)		
DeleteIPSet	Grants permission to delete GuardDuty IPSets.	Write	detector* (p. 1217)		
			ipset* (p. 1217)		
DeleteInvitations	Grants permission to delete invitations to become a GuardDuty member account.	Write			
DeleteMembers	Grants permission to delete GuardDuty member accounts.	Write	detector* (p. 1217)		
DeletePublishingDestination	Grants permission to delete a publishing destination.	Write	detector* (p. 1217)		
			publishingdestination* (p. 1218)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteThreatIntelGuardDuty	Grants permission to delete GuardDuty ThreatIntelSets.	Write	detector* (p. 1217)		
	threatintelset* (p. 1217)				
DescribePublishingDetailsAbout	Grants permission to retrieve details about a publishing destination.	Read	detector* (p. 1217)		
	publishingdestination* (p. 1218)				
DisassociateFromMasterAccount	Grants permission to disassociate a GuardDuty member account from its GuardDuty master account.	Write	detector* (p. 1217)		
DisassociateMembers	Grants permission to disassociate GuardDuty member accounts from their master GuardDuty account.	Write	detector* (p. 1217)		
GetDetector	Grants permission to retrieve GuardDuty detectors.	Read	detector* (p. 1217)		
GetFilter	Grants permission to retrieve GuardDuty filters.	Read	detector* (p. 1217)		
	filter* (p. 1217)				
GetFindings	Grants permission to retrieve GuardDuty findings.	Read	detector* (p. 1217)		
GetFindingsStatistics	Grants permission to retrieve statistics of GuardDuty finding statistics.	Read	detector* (p. 1217)		
GetIPSet	Grants permission to retrieve GuardDuty IP Sets.	Read	detector* (p. 1217)		
	ipset* (p. 1217)				
GetInvitationsCount	Grants permission to retrieve the count of all GuardDuty invitations sent to a specified account. The count does not include an accepted invitation.	Read			
GetMasterAccount	Grants permission to retrieve details of the GuardDuty master account associated with a member account.	Read	detector* (p. 1217)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetMembers	Grants permission to retrieve the member accounts associated with a master account.	Read	detector* (p. 1217)		
GetThreatIntelSet	Grants permission to retrieve GuardDuty ThreatIntelSets.	Read	detector* (p. 1217)		
			threatintelset* (p. 1217)		
InviteMembers	Grants permission to invite other AWS accounts to enable GuardDuty and become GuardDuty member accounts.	Write	detector* (p. 1217)		
ListDetectors	Grants permission to retrieve a list of GuardDuty detectors.	List			
ListFilters	Grants permission to retrieve a list of GuardDuty filters.	List	detector* (p. 1217)		
ListFindings	Grants permission to retrieve a list of GuardDuty findings.	List	detector* (p. 1217)		
ListInvitations	Grants permission to retrieve a lists of all of the GuardDuty membership invitations that were sent to an AWS account.	List			
ListMembers	Grants permission to retrieve a lsit of GuardDuty member accounts associated with a master account.	List	detector* (p. 1217)		
ListPublishingDestinations	Grants permission to retrieve a list of publishing destinations.	List	detector* (p. 1217)		
ListTagsForResource	Grants permission to retrieve a list of tags associated with a GuardDuty resource.	List	detector (p. 1217)		
			filter (p. 1217)		
			ipset (p. 1217)		
			threatintelset (p. 1217)		
ListThreatIntelSets	Grants permission to retrieve a list of GuardDuty ThreatIntelSets.	List	detector* (p. 1217)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
StartMonitoringMembers	Grants permission to a master account to monitor findings from GuardDuty member accounts. Use this after disabling monitoring of member accounts using the StopMonitoringMembers operation.	Write	detector* (p. 1217)		
StopMonitoringMembers	Grants permission to disable monitoring findings from member accounts.	Write	detector* (p. 1217)		
TagResource	Grants permission to add tags to a GuardDuty resource. There is a limit of 50 tags per resource.	Write	detector (p. 1217)		
filter (p. 1217)					
ipset (p. 1217)					
threatintelset (p. 1217)					
	aws:RequestTag/\${TagKey} (p. 1218)				
UnarchiveFinding	Grants permission to unarchive GuardDuty findings.	Write	detector* (p. 1217)		
UntagResource	Grants permission to remove tags from a GuardDuty resource.	Write	detector (p. 1217)		
filter (p. 1217)					
ipset (p. 1217)					
threatintelset (p. 1217)					
	aws:TagKeys (p. 1218)				
UpdateDetector	Grants permission to update GuardDuty detectors.	Write	detector* (p. 1217)		
UpdateFilter	Grants permission to update GuardDuty filters.	Write	detector* (p. 1217)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			filter* (p. 1217)		
UpdateFindingsFeedback	Grants permission to update findings feedback to mark GuardDuty findings as useful or not useful.	Write	detector* (p. 1217)		
UpdateIPSet	Grants permission to update GuardDuty IPSets.	Write	detector* (p. 1217)		
			ipset* (p. 1217)		
UpdatePublishingDestination	Grants permission to update a publishing destination .	Write	detector* (p. 1217)		s3:GetObject s3>ListBucket
			publishingdestination* (p. 1218)		
UpdateThreatIntelSet	Grants permission to update the GuardDuty ThreatIntelSets.	Write	detector* (p. 1217)		
			threatintelset* (p. 1217)		

Resource Types Defined by Amazon GuardDuty

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 1212\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
detector	arn:\${Partition}:guardduty:\${Region}: \${Account}:detector/\${DetectorId}	aws:ResourceTag/ \${TagKey} (p. 1218)
filter	arn:\${Partition}:guardduty:\${Region}: \${Account}:detector/\${DetectorId}/filter/ \${FilterName}	aws:ResourceTag/ \${TagKey} (p. 1218)
ipset	arn:\${Partition}:guardduty:\${Region}: \${Account}:detector/\${DetectorId}/ipset/ \${IPSetId}	aws:ResourceTag/ \${TagKey} (p. 1218)
threatintelset	arn:\${Partition}:guardduty:\${Region}: \${Account}:detector/\${DetectorId}/ threatintelset/\${ThreatIntelSetId}	aws:ResourceTag/ \${TagKey} (p. 1218)

Resource Types	ARN	Condition Keys
publishingdestination	<code>arn:\${Partition}:guardduty:\${Region}: \${Account}:detector/\${DetectorId}/ threatintelset/\${PublishingDestinationId}</code>	

Condition Keys for Amazon GuardDuty

Amazon GuardDuty defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Filters actions based on the presence of tag key-value pairs in the request	String
aws:ResourceTag/\${TagKey}	Filters actions based on tag key-value pairs attached to the resource	String
aws:TagKeys	Filters actions based on the presence of tag keys in the request	String

Actions, Resources, and Condition Keys for AWS Health APIs and Notifications

AWS Health APIs and Notifications (service prefix: `health`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Health APIs and Notifications \(p. 1218\)](#)
- [Resource Types Defined by AWS Health APIs and Notifications \(p. 1220\)](#)
- [Condition Keys for AWS Health APIs and Notifications \(p. 1220\)](#)

Actions Defined by AWS Health APIs and Notifications

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases,

a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the Resource element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeAffectedAccounts	Gets a list of accounts that have been affected by the specified events in organization.	Read			organizations>ListAccounts
DescribeAffectedEntities	Gets a list of entities that have been affected by the specified events.	Read	event* (p. 1220)		
				health:eventTypeCode (p. 1221)	
DescribeAffectedOrganizations	Gets a list of entities that have been affected by the specified events and accounts in organization.	Read			organizations>ListAccounts
DescribeEntityAggregates	Returns the number of entities that are affected by each of the specified events.	Read			
DescribeEventAggregates	Returns the number of events of each event type (issue, scheduled change, and account notification).	Read			
DescribeEventDetails	Returns detailed information about one or more specified events.	Read	event* (p. 1220)		
				health:eventTypeCode (p. 1221)	
DescribeEventDetailsForOrganization	Returns detailed information about one or more specified events for provided accounts in organization.	Read			organizations>ListAccounts

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeEventTypes	Returns the event types that meet the specified filter criteria.	Read			
DescribeEvents	Returns information about events that meet the specified filter criteria.	Read			
DescribeEventsForOrganization	Returns information about events that meet the specified filter criteria in organization.	Read			organizations>ListAccounts
DescribeHealthServiceFeaturesForOrganization	Returns the status of enabling or disabling the Organizational View feature	Permissions management			organizations>ListAccounts
DisableHealthServiceFeaturesForOrganization	Disables the Organizational View feature	Permissions management			organizations>DisableAWSOrganizationsFeature organizations>ListAccounts
EnableHealthServiceFeaturesForOrganization	Enables the Organizational View feature	Permissions management			iam>CreateServiceLinkedRole organizations>EnableAWSOrganizationsFeature organizations>ListAccounts

Resource Types Defined by AWS Health APIs and Notifications

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 1218\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
event	arn:\${Partition}:health:*::event/\${Service}/\${EventTypeCode}/*	

Condition Keys for AWS Health APIs and Notifications

AWS Health APIs and Notifications defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
health:eventTypeCode	The type of event.	String
health:service	The service of the event.	String

Actions, Resources, and Condition Keys for IAM Access Analyzer

IAM Access Analyzer (service prefix: `access-analyzer`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by IAM Access Analyzer \(p. 1221\)](#)
- [Resource Types Defined by IAM Access Analyzer \(p. 1223\)](#)
- [Condition Keys for IAM Access Analyzer \(p. 1224\)](#)

Actions Defined by IAM Access Analyzer

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateAnalyzer	Grants permission to create an analyzer.	Write	Analyzer* (p. 1223)		aws:RequestTag/\${TagKey} (p. 1224) aws:TagKeys (p. 1224)

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateArchiveRule	Grants permission to create an archive rule for the specified analyzer.	Write	Analyzer* (p. 1223)		
			ArchiveRule* (p. 1224)		
DeleteAnalyzer	Grants permission to delete the specified analyzer.	Write	Analyzer* (p. 1223)		
				aws:RequestTag/\${TagKey} (p. 1224) aws:TagKeys (p. 1224)	
DeleteArchiveRule	Grants permission to delete archive rules for the specified analyzer.	Write	Analyzer* (p. 1223)		
			ArchiveRule* (p. 1224)		
GetAnalyzedResource	Grants permission to retrieve information about an analyzed resource.	Read			
GetAnalyzer	Grants permission to retrieve information about analyzers.	Read	Analyzer* (p. 1223)		
				aws:RequestTag/\${TagKey} (p. 1224) aws:TagKeys (p. 1224)	
GetArchiveRule	Grants permission to retrieve information about archive rules for the specified analyzer.	Read	Analyzer* (p. 1223)		
			ArchiveRule* (p. 1224)		
GetFinding	Grants permission to retrieve findings.	Read			
ListAnalyzedResources	Grants permission to retrieve a list of resources that have been analyzed.	List			
ListAnalyzers	Grants permission to retrieves a list of analyzers.	List			
ListArchiveRules	Grants permission to retrieve a list of archive rules from an analyzer.	List	Analyzer* (p. 1223)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListFindings	Grants permission to retrieve a list of findings from an analyzer.	List			
ListTagsForResource	Grants permission to retrieve a list of tags applied to a resource.	List	Analyzer (p. 1223)		
StartResourceScan	Grants permission to start a scan of the policies applied to a resource.	Write			
TagResource	Grants permission to add a tag to a resource.	Tagging	Analyzer (p. 1223)		
				aws:RequestTag/\${TagKey} (p. 1224)	
				aws:TagKeys (p. 1224)	
UntagResource	Grants permission to remove a tag from a resource.	Tagging	Analyzer (p. 1223)		
				aws:RequestTag/\${TagKey} (p. 1224)	
				aws:TagKeys (p. 1224)	
UpdateArchiveRule	Grants permission to modify an archive rule.	Write	Analyzer* (p. 1223)		
				ArchiveRule* (p. 1224)	
UpdateFindings	Grants permission to modify findings.	Write			

Resource Types Defined by IAM Access Analyzer

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1221\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
Analyzer	<code>arn:\${Partition}:access-analyzer:\${Region}:\${Account}:analyzer/\${analyzerName}</code>	aws:ResourceTag/\${TagKey} (p. 1224)

Resource Types	ARN	Condition Keys
ArchiveRule	arn:\${Partition}:access-analyzer:\${Region}:\${Account}:analyzer/\${analyzerName}/archive-rule/\${ruleName}	

Condition Keys for IAM Access Analyzer

IAM Access Analyzer defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Filters actions based on the presence of tag key-value pairs in the request	String
aws:ResourceTag/\${TagKey}	Filters actions based on tag key-value pairs attached to the resource	String
aws:TagKeys	Filters actions based on the presence of tag keys in the request	String

Actions, Resources, and Condition Keys for Identity And Access Management

Identity And Access Management (service prefix: `iam`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Identity And Access Management \(p. 1224\)](#)
- [Resource Types Defined by Identity And Access Management \(p. 1237\)](#)
- [Condition Keys for Identity And Access Management \(p. 1238\)](#)

Actions Defined by Identity And Access Management

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the Resource element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AddClientIDToOpenIDProvider	Grants permission to add a new client ID (<code>clientID</code>) to the list of registered IDs for the specified IAM OpenID Connect (OIDC) provider resource	Write	oidc-provider* (p. 1238)		
AddRoleToInstanceProfile	Grants permission to add an IAM user to the specified instance profile	Write	instance-profile* (p. 1238)		
AddUserToGroup	Grants permission to add an IAM user to the specified IAM group	Write	group* (p. 1238)		
AttachGroupPolicy	Grants permission to attach a managed policy to the specified IAM group	Permissions management	group* (p. 1238)		
				iam:PolicyARN (p. 1238)	
AttachRolePolicy	Grants permission to attach a managed policy to the specified IAM role	Permissions management	role* (p. 1238)		
				iam:PolicyARN (p. 1238)	
				iam:PermissionsBoundary (p. 1238)	
AttachUserPolicy	Grants permission to attach a managed policy to the specified IAM user	Permissions management	user* (p. 1238)		
				iam:PolicyARN (p. 1238)	
				iam:PermissionsBoundary (p. 1238)	
ChangePassword	Grants permission for an IAM user to change their own password	Write	user* (p. 1238)		
CreateAccessKey	Grants permission to create access key and secret access key for the specified IAM user	Write	user* (p. 1238)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateAccountAlias	Grants permission to create an alias for your AWS account	Write			
CreateGroup	Grants permission to create a new group	Write	group* (p. 1238)		
CreateInstanceProfile	Grants permission to create a new instance profile	Write	instance-profile* (p. 1238)		
CreateLoginProfile	Grants permission to create a password for the specified IAM user	Write	user* (p. 1238)		
CreateOpenIDConnectProvider	Grants permission to create an IAM resource that describes an identity provider (IdP) that supports OpenID Connect (OIDC)	Write	oidc-provider* (p. 1238)		
CreatePolicy	Grants permission to create a new managed policy	Permissions management	policy* (p. 1238)		
CreatePolicyVersion	Grants permission to create a new version of the specified managed policy	Permissions management	policy* (p. 1238)		
CreateRole	Grants permission to create a new role	Write	role* (p. 1238)		
			iam:PermissionsBoundary (p. 1238)		
CreateSAMLProvider	Grants permission to create an IAM resource that describes an identity provider (IdP) that supports SAML 2.0	Write	saml-provider* (p. 1238)		
CreateServiceLinkedRole	Grants permission to create an IAM role that allows an AWS service to perform actions on your behalf	Write	role* (p. 1238)		
			iam:AWSServiceName (p. 1238)		
CreateServiceSpecificCredential	Grants permission to create a new service-specific credential for an IAM user	Write	user* (p. 1238)		
CreateUser	Grants permission to create a new IAM user	Write	user* (p. 1238)		
			iam:PermissionsBoundary (p. 1238)		
CreateVirtualMFADevice	Grants permission to create a new virtual MFA device	Write	mfa* (p. 1238)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeactivateMFADevice	Grants permission to deactivate the specified MFA device and remove its association with the IAM user for which it was originally enabled	Write	user* (p. 1238)		
DeleteAccessKey	Grants permission to delete the access key pair that is associated with the specified IAM user	Write	user* (p. 1238)		
DeleteAccountAlias	Grants permission to delete the specified AWS account alias	Write			
DeleteAccountPasswordPolicy	Grants permission to delete the password policy for the AWS account		Permissions management		
DeleteGroup	Grants permission to delete the specified IAM group	Write	group* (p. 1238)		
DeleteGroupPolicy	Grants permission to delete the specified inline policy from its group		Permissions management	group* (p. 1238)	
DeleteInstanceProfile	Grants permission to delete the specified instance profile	Write		instance-profile* (p. 1238)	
DeleteLoginProfile	Grants permission to delete the password for the specified IAM user	Write	user* (p. 1238)		
DeleteOpenIDCorporation	Grants permission to delete an OpenID Connect identity provider (IdP) resource object in IAM	Write		oidc-provider* (p. 1238)	
DeletePolicy	Grants permission to delete the specified managed policy and remove it from any IAM entities (users, groups, or roles) to which it is attached		Permissions management	policy* (p. 1238)	
DeletePolicyVersion	Grants permission to delete a version from the specified managed policy		Permissions management	policy* (p. 1238)	
DeleteRole	Grants permission to delete the specified role	Write		role* (p. 1238)	
DeleteRolePermissionsBoundary	Grants permission to remove the permissions boundary from a role		Permissions management	role* (p. 1238)	
					iam:PermissionsBoundary (p. 1238)

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteRolePolicy	Grants permission to delete the specified inline policy from the specified role	Permissions management	role* (p. 1238)		
	iam:PermissionsBoundary (p. 1238)				
DeleteSAMLProvider	Grants permission to delete a SAML provider resource in IAM	Write	saml-provider* (p. 1238)		
DeleteSSHPublicKey	Grants permission to delete the specified SSH public key	Write	user* (p. 1238)		
DeleteServerCertificate	Grants permission to delete the specified server certificate	Write	server-certificate* (p. 1238)		
DeleteServiceLinkedRole	Grants permission to delete the role that is linked to a specific AWS service, if the service is no longer using it	Write	role* (p. 1238)		
DeleteServiceSpecificCredential	Grants permission to delete the specified service-specific credential for an IAM user	Write	user* (p. 1238)		
DeleteSigningCertificate	Grants permission to delete the signing certificate that is associated with the specified IAM user	Write	user* (p. 1238)		
DeleteUser	Grants permission to delete the specified IAM user	Write	user* (p. 1238)		
DeleteUserPermissionsBoundary	Grants permission to remove the permissions boundary from the specified IAM user	Permissions management	user* (p. 1238)		
iam:PermissionsBoundary (p. 1238)					
DeleteUserPolicy	Grants permission to delete the specified inline policy from an IAM user	Permissions management	user* (p. 1238)		
iam:PermissionsBoundary (p. 1238)					
DeleteVirtualMFADevice	Grants permission to delete a virtual MFA device	Write	mfa (p. 1238)		
sms-mfa (p. 1238)					
DetachGroupPolicy	Grants permission to detach a managed policy from the specified IAM group	Permissions management	group* (p. 1238)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				iam:PolicyARN (p. 1238)	
DetachRolePolicy	Grants permission to detach a managed policy from the specified role	Permissions management (p. 1238)	role*		
				iam:PolicyARN (p. 1238)	iam:PermissionsBoundary (p. 1238)
DetachUserPolicy	Grants permission to detach a managed policy from the specified IAM user	Permissions management (p. 1238)	user*		
				iam:PolicyARN (p. 1238)	iam:PermissionsBoundary (p. 1238)
EnableMFADevice	Grants permission to enable an MFA device and associate it with the specified IAM user	Write	user* (p. 1238)		
GenerateCredentialReport	Grants permission to generate a credential report for the AWS account	Read			
GenerateOrganizationAccessReport	Grants permission to generate an access report for an AWS Organizations entity	Read	access-report* (p. 1237)		organizations:DescribePolicies organizations>ListChildren organizations>ListParents organizations>ListPolicies organizations>ListRoots organizations>ListTargets
				iam:OrganizationsPolicyId (p. 1238)	
GenerateServiceLastAccessReport	Grants permission to generate a service last access data report for an IAM resource	Read			
GetAccessKeyLastUsed	Grants permission to retrieve information about when the specified access key was last used	Read	user* (p. 1238)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetAccountAuthorizationInformation	Grants permission to retrieve information about all IAM users, groups, roles, and policies in your AWS account, including their relationships to one another	Read			
GetAccountPasswordPolicy	Grants permission to retrieve the password policy for the AWS account	Read			
GetAccountSummary	Grants permission to retrieve information about IAM entity usage and IAM quotas in the AWS account	List			
GetContextKeysForListOfAllContextKeys	Grants permission to retrieve a list of all context keys that are referenced in the specified policy	Read			
GetContextKeysForListOfAllPolicies	Grants permission to retrieve a list of all context keys that are referenced in all IAM policies that are attached to the specified IAM identity (user, group, or role)	Read	group (p. 1238)		
			role (p. 1238)		
			user (p. 1238)		
GetCredentialReport	Grants permission to retrieve a credential report for the AWS account	Read			
GetGroup	Grants permission to retrieve a list of IAM users in the specified IAM group	Read	group* (p. 1238)		
GetGroupPolicy	Grants permission to retrieve an inline policy document that is embedded in the specified IAM group	Read	group* (p. 1238)		
GetInstanceProfile	Grants permission to retrieve information about the specified instance profile, including the instance profile's path, GUID, ARN, and role	Read	instance-profile* (p. 1238)		
GetLoginProfile	Grants permission to retrieve the user name and password creation date for the specified IAM user	List	user* (p. 1238)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetOpenIDConnectInformation	Grants permission to retrieve information about the specified OpenID Connect (OIDC) provider resource in IAM	Read	oidc-provider* (p. 1238)		
GetOrganizationsAWSOrganizations	Grants permission to retrieve an AWS Organizations access report	Read			
GetPolicy	Grants permission to retrieve information about the specified managed policy, including the policy's default version and the total number of identities to which the policy is attached	Read	policy* (p. 1238)		
GetPolicyVersion	Grants permission to retrieve information about a version of the specified managed policy, including the policy document	Read	policy* (p. 1238)		
GetRole	Grants permission to retrieve information about the specified role, including the role's path, GUID, ARN, and the role's trust policy	Read	role* (p. 1238)		
GetRolePolicy	Grants permission to retrieve an inline policy document that is embedded with the specified IAM role	Read	role* (p. 1238)		
GetSAMLProvider	Grants permission to retrieve the SAML provider metadocument that was uploaded when the IAM SAML provider resource was created or updated	Read	saml-provider* (p. 1238)		
GetSSHPublicKey	Grants permission to retrieve the specified SSH public key, including metadata about the key	Read	user* (p. 1238)		
GetServerCertificate	Grants permission to retrieve information about the specified server certificate stored in IAM	Read	server-certificate* (p. 1238)		
GetServiceLastAccessedData	Grants permission to retrieve information about the service last accessed data report	Read			
GetServiceLastAccessedDataWithEntities	Grants permission to retrieve information about the entities from the service last accessed data report	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetServiceLinkedRoleDeletionStatus	Grants permission to retrieve an IAM service-linked role deletion status	Read	role* (p. 1238)		
 GetUser	Grants permission to retrieve information about the specified IAM user, including the user's creation date, path, unique ID, and ARN	Read	user* (p. 1238)		
 GetUserPolicy	Grants permission to retrieve an inline policy document that is embedded in the specified IAM user	Read	user* (p. 1238)		
 ListAccessKeys	Grants permission to list information about the access key IDs that are associated with the specified IAM user	List	user* (p. 1238)		
 ListAccountAliases	Grants permission to list the account alias that is associated with the AWS account	List			
 ListAttachedGroupPolicies	Grants permission to list all managed policies that are attached to the specified IAM group	List	group* (p. 1238)		
 ListAttachedRolePolicies	Grants permission to list all managed policies that are attached to the specified IAM role	List	role* (p. 1238)		
 ListAttachedUserPolicies	Grants permission to list all managed policies that are attached to the specified IAM user	List	user* (p. 1238)		
 ListEntitiesForPolicy	Grants permission to list all IAM identities to which the specified managed policy is attached	List	policy* (p. 1238)		
 ListGroupPolicies	Grants permission to list the names of the inline policies that are embedded in the specified IAM group	List	group* (p. 1238)		
 ListGroups	Grants permission to list the IAM groups that have the specified path prefix	List			
 ListGroupsForUser	Grants permission to list the IAM groups that the specified IAM user belongs to	List	user* (p. 1238)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListInstanceProfiles	Grants permission to list the instance profiles that have the specified path prefix	List	instance-profile* (p. 1238)		
ListInstanceProfilesForRole	Grants permission to list the instance profiles that have the specified associated IAM role	List	role* (p. 1238)		
ListMFADevices	Grants permission to list the MFA devices for an IAM user	List	user (p. 1238)		
ListOpenIDConnectInformation	Grants permission to list information about the IAM OpenID Connect (OIDC) provider resource objects that are defined in the AWS account	List			
ListPolicies	Grants permission to list all managed policies	List			
ListPoliciesGrantingPermissions	Grants permission to list information about the policies that grant an entity access to a specific service	List			
ListPolicyVersions	Grants permission to list information about the versions of the specified managed policy, including the version that is currently set as the policy's default version	List	policy* (p. 1238)		
ListRolePolicies	Grants permission to list the names of the inline policies that are embedded in the specified IAM role	List	role* (p. 1238)		
ListRoleTags	Grants permission to list the tags that are attached to the specified IAM role.	List	role* (p. 1238)		
ListRoles	Grants permission to list the IAM roles that have the specified path prefix	List			
ListSAMLProvider	Grants permission to list the SAML provider resources in IAM	List			
ListSSHPublicKeys	Grants permission to list information about the SSH public keys that are associated with the specified IAM user	List	user* (p. 1238)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListServerCertificates	Grants permission to list the server certificates that have the specified path prefix	List			
ListServiceSpecificCredentials	Grants permission to list the service specific credentials that are associated with the specified IAM user	List	user* (p. 1238)		
ListSigningCertificates	Grants permission to list information about the signing certificates that are associated with the specified IAM user	List	user* (p. 1238)		
ListUserPolicies	Grants permission to list the names of the inline policies that are embedded in the specified IAM user	List	user* (p. 1238)		
ListUserTags	Grants permission to list the tags that are attached to the specified IAM user.	List	user* (p. 1238)		
ListUsers	Grants permission to list the IAM users that have the specified path prefix	List			
ListVirtualMFADevices	Grants permission to list virtual MFA devices by assignment status	List			
PassRole [permission only]	Grants permission to pass a role to a service	Write	role* (p. 1238)		
				iam:PassedToService (p. 1238)	
PutGroupPolicy	Grants permission to create or update an inline policy document that is embedded in the specified IAM group	Permissions management	group* (p. 1238)		
PutRolePermissionsBoundary	Grants permission to set a managed policy as a permissions boundary for a role	Permissions management	role* (p. 1238)		
				iam:PermissionsBoundary (p. 1238)	
PutRolePolicy	Grants permission to create or update an inline policy document that is embedded in the specified IAM role	Permissions management	role* (p. 1238)		
				iam:PermissionsBoundary (p. 1238)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
PutUserPermissionsBoundary	Grants permission to set a managed policy as a permissions boundary for an IAM user	Permissions management	user* (p. 1238)		
				iam:PermissionsBoundary (p. 1238)	
PutUserPolicy	Grants permission to create or update an inline policy document that is embedded in the specified IAM user	Permissions management	user* (p. 1238)		
				iam:PermissionsBoundary (p. 1238)	
RemoveClientIDFromOpenIDConnectProvider	Grants permission to remove the client ID (audience) from the list of client IDs in the specified IAM OpenID Connect (OIDC) provider resource	Write	oidc-provider* (p. 1238)		
RemoveRoleFromInstanceProfile	Grants permission to remove an IAM role from the specified EC2 instance profile	Write	instance-profile* (p. 1238)		
RemoveUserFromGroup	Grants permission to remove an IAM user from the specified group	Write	group* (p. 1238)		
ResetServiceSpecificPassword	Grants permission to reset the password for an existing service-specific credential for an IAM user	Write	user* (p. 1238)		
ResyncMFADevice	Grants permission to synchronize the specified MFA device with its IAM entity (user or role)	Write	user* (p. 1238)		
SetDefaultPolicyVersion	Grants permission to set the version of the specified policy as the policy's default version	Permissions management	policy* (p. 1238)		
SetSecurityTokenGlobalEndpoint	Grants permission to set the STS token version	Write			
SimulateCustomPolicy	Grants permission to simulate whether an identity-based policy or resource-based policy provides permissions for specific API operations and resources	Read			
SimulatePrincipalPolicy	Grants permission to simulate whether an identity-based policy that is attached to a specified IAM entity (user or role) provides permissions for specific API operations and resources	Read	group (p. 1238)		
			role (p. 1238)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			user (p. 1238)		
TagRole	Grants permission to add tags to an IAM role.	Tagging	role* (p. 1238)		
TagUser	Grants permission to add tags to an IAM user.	Tagging	user* (p. 1238)		
UntagRole	Grants permission to remove the specified tags from the role.	Tagging	role* (p. 1238)		
UntagUser	Grants permission to remove the specified tags from the user.	Tagging	user* (p. 1238)		
UpdateAccessKey	Grants permission to update the status of the specified access key as Active or Inactive	Write	user* (p. 1238)		
UpdateAccountPasswordPolicy	Grants permission to update the password policy settings for the AWS account	Write			
UpdateAssumeRolePolicy	Grants permission to update the policy that grants an IAM entity permission to assume a role	Permissions management	role* (p. 1238)		
UpdateGroup	Grants permission to update the name or path of the specified IAM group	Write	group* (p. 1238)		
UpdateLoginProfile	Grants permission to change the password for the specified IAM user	Write	user* (p. 1238)		
UpdateOpenIDConnectProvider	Grants permission to update the entire list of server certificates that are associated with an OpenID Connect (OIDC) provider resource	Write	oidc-provider* (p. 1238)		
UpdateRole	Grants permission to update the description or maximum session duration setting of a role	Write	role* (p. 1238)		
UpdateRoleDescription	Grants permission to update only the description of a role	Write	role* (p. 1238)		
UpdateSAMLProvider	Grants permission to update the metadata document for an existing SAML provider resource	Write	saml-provider* (p. 1238)		
UpdateSSHPublicKey	Grants permission to update the status of an IAM user's SSH public key to active or inactive	Write	user* (p. 1238)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UpdateServerCertificate	Grants permission to update the name or the path of the specified server certificate stored in IAM	Write	server-certificate* (p. 1238)		
UpdateServiceSpecificCredential	Grants permission to update the status of a service-specific credential to active or inactive for an IAM user	Write	user* (p. 1238)		
UpdateSigningCertificate	Grants permission to update the status of the specified user signing certificate to active or disabled	Write	user* (p. 1238)		
UpdateUser	Grants permission to update the name or the path of the specified IAM user	Write	user* (p. 1238)		
UploadSSHPublicKey	Grants permission to upload an SSH public key and associate it with the specified IAM user	Write	user* (p. 1238)		
UploadServerCertificate	Grants permission to upload a server certificate entity for the AWS account	Write	server-certificate* (p. 1238)		
UploadSigningCertificate	Grants permission to upload an X.509 signing certificate and associate it with the specified IAM user	Write	user* (p. 1238)		

Resource Types Defined by Identity And Access Management

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1224\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
access-report	<code>arn:\${Partition}:iam:::\${Account}:access-report/\${EntityPath}</code>	
assumed-role	<code>arn:\${Partition}:iam:::\${Account}:assumed-role/\${RoleName}/\${RoleSessionName}</code>	
federated-user	<code>arn:\${Partition}:iam:::\${Account}:federated-user/\${UserName}</code>	

Resource Types	ARN	Condition Keys
group	arn:\${Partition}:iam::\${Account}:group/\${GroupNameWithPath}	
instance-profile	arn:\${Partition}:iam::\${Account}:instance-profile/\${InstanceProfileNameWithPath}	
mfa	arn:\${Partition}:iam::\${Account}:mfa/\${Path}/\${MfaTokenId}	
oidc-provider	arn:\${Partition}:iam::\${Account}:oidc-provider/\${OidcProviderName}	
policy	arn:\${Partition}:iam::\${Account}:policy/\${PolicyNameWithPath}	
role	arn:\${Partition}:iam::\${Account}:role/\${RoleNameWithPath}	iam:ResourceTag/\${TagKey} (p. 1239)
saml-provider	arn:\${Partition}:iam::\${Account}:saml-provider/\${SamlProviderName}	
server-certificate	arn:\${Partition}:iam::\${Account}:server-certificate/\${CertificateNameWithPath}	
sms-mfa	arn:\${Partition}:iam::\${Account}:sms-mfa/\${MfaTokenIdWithPath}	
user	arn:\${Partition}:iam::\${Account}:user/\${UserNameWithPath}	iam:ResourceTag/\${TagKey} (p. 1239)

Condition Keys for Identity And Access Management

Identity And Access Management defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
iam:AWSServiceName	Filters access by the AWS service to which this role is attached	String
iam:OrganizationsPolicyId	Filters access by the ID of an AWS Organizations policy	String
iam:PassedToService	Filters access by the AWS service to which this role is passed	String
iam:PermissionsBoundary	Filters access if the specified policy is set as the permissions boundary on the IAM entity (user or role)	String
iam:PolicyARN	Filters access by the ARN of an IAM policy	ARN

Condition Keys	Description	Type
iam:ResourceTag/ \${TagKey}	Filters access by the tags attached to an IAM entity (user or role).	String

Actions, Resources, and Condition Keys for AWS Import Export Disk Service

AWS Import Export Disk Service (service prefix: `importexport`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Import Export Disk Service \(p. 1239\)](#)
- [Resource Types Defined by AWS Import Export Disk Service \(p. 1240\)](#)
- [Condition Keys for AWS Import Export Disk Service \(p. 1240\)](#)

Actions Defined by AWS Import Export Disk Service

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CancelJob	This action cancels a specified job. Only the job owner can cancel it. The action fails if the job has already started or is complete.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateJob	This action initiates the process of scheduling an upload or download of your data.	Write			
GetShippingLabel	This action generates a pre-paid shipping label that you will use to ship your device to AWS for processing.	Read			
GetStatus	This action returns information about a job, including where the job is in the processing pipeline, the status of the results, and the signature value associated with the job.	Read			
ListJobs	This action returns the jobs associated with the requester.	List			
UpdateJob	You use this action to change the parameters specified in the original manifest file by supplying a new manifest file.	Write			

Resource Types Defined by AWS Import Export Disk Service

AWS Import Export Disk Service does not support specifying a resource ARN in the Resource element of an IAM policy statement. To allow access to AWS Import Export Disk Service, specify "Resource": "*" in your policy.

Condition Keys for AWS Import Export Disk Service

Import/Export has no service-specific context keys that can be used in the Condition element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Amazon Inspector

Amazon Inspector (service prefix: `inspector`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Inspector \(p. 1241\)](#)
- [Resource Types Defined by Amazon Inspector \(p. 1244\)](#)
- [Condition Keys for Amazon Inspector \(p. 1244\)](#)

Actions Defined by Amazon Inspector

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AddAttributesToFindings	Assigns attributes (key and value pairs) to the findings that are specified by the ARNs of the findings.	Write			
CreateAssessmentUsing	Creates a new assessment target using the ARN of the resource group that is generated by <code>CreateResourceGroup</code> .	Write			
CreateAssessmentFor	Creates an assessment template for the assessment target that is specified by the ARN of the assessment target.	Write			
CreateResourceGroup	Creates a resource group using the specified set of tags (key and value pairs) that are used to select the EC2 instances to be included in an Amazon Inspector assessment target.	Write			
DeleteAssessmentRun	Deletes the assessment run that is specified by the ARN of the assessment run.	Write			
DeleteAssessmentTarget	Deletes the assessment target that is specified by the ARN of the assessment target.	Write			
DeleteAssessmentTemplate	Deletes the assessment template that is specified by the ARN of the assessment template.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeAssessmentRuns	Describes the assessment runs that are specified by the ARNs of the assessment runs.	Read			
DescribeAssessmentTargets	Describes the assessment targets that are specified by the ARNs of the assessment targets.	Read			
DescribeAssessmentTemplates	Describes the assessment templates that are specified by the ARNs of the assessment templates.	Read			
DescribeCrossAccountAccessRole	Describes the IAM role that enables Amazon Inspector to access your AWS account.	Read			
DescribeFindings	Describes the findings that are specified by the ARNs of the findings.	Read			
DescribeResourceGroups	Describes the resource groups that are specified by the ARNs of the resource groups.	Read			
DescribeRulesPackages	Describes the rules packages that are specified by the ARNs of the rules packages.	Read			
GetTelemetryMetrics	Information about the data that is collected for the specified assessment run.	Read			
ListAssessmentRuns	Lists the agents of the assessment runs that are specified by the ARNs of the assessment runs.	List			
ListAssessmentRuns	Lists the assessment runs that correspond to the assessment templates that are specified by the ARNs of the assessment templates.	List			
ListAssessmentTargets	Lists the ARNs of the assessment targets within this AWS account.	List			
ListAssessmentTemplates	Lists the assessment templates that correspond to the assessment targets that are specified by the ARNs of the assessment targets.	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListEventSubscriptionsForAssessmentTemplate	Lists all the event subscriptions for the assessment template that is specified by the ARN of the assessment template.	List			
ListFindings	Lists findings that are generated by the assessment runs that are specified by the ARNs of the assessment runs.	List			
ListRulesPackages	Lists all available Amazon Inspector rules packages.	List			
ListTagsForResource	Lists all tags associated with an assessment template.	List			
PreviewAgents	Previews the agents installed on the EC2 instances that are part of the specified assessment target.	Read			
RegisterCrossAccountRole	Registers the IAM role that Amazon Inspector uses to list your EC2 instances at the start of the assessment run or when you call the PreviewAgents action.	Write			
RemoveAttributesFromFindings	Removes entire attributes (key and value pairs) from the findings that are specified by the ARNs of the findings where an attribute with the specified key exists.	Write			
SetTagsForResource	Sets tags (key and value pairs) to the assessment template that is specified by the ARN of the assessment template.	Tagging			
StartAssessmentRun	Starts the assessment run specified by the ARN of the assessment template.	Write			
StopAssessmentRun	Stops the assessment run that is specified by the ARN of the assessment run.	Write			
SubscribeToEvent	Enables the process of sending Amazon Simple Notification Service (SNS) notifications about a specified event to a specified SNS topic.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UnsubscribeFromTopic	Disables the process of sending Amazon Simple Notification Service (SNS) notifications about a specified event to a specified SNS topic.	Write			
UpdateAssessmentTarget	Updates the assessment target that is specified by the ARN of the assessment target.	Write			

Resource Types Defined by Amazon Inspector

Amazon Inspector does not support specifying a resource ARN in the `Resource` element of an IAM policy statement. To allow access to Amazon Inspector, specify “`Resource`”: “`*`” in your policy.

Condition Keys for Amazon Inspector

Inspector has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for AWS IoT

AWS IoT (service prefix: `iot`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS IoT \(p. 1244\)](#)
- [Resource Types Defined by AWS IoT \(p. 1260\)](#)
- [Condition Keys for AWS IoT \(p. 1261\)](#)

Actions Defined by AWS IoT

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources (“`*`”) in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in

a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AcceptCertificateTransfer	Accepts a pending certificate transfer.	Write			
AddThingToBillingGroup	Adds a thing to the specified billing group.	Write	billinggroup* (p. 1260)		
			thing* (p. 1260)		
AddThingToThingGroup	Adds a thing to the specified thing group.	Write	thing* (p. 1260)		
			thinggroup* (p. 1260)		
AssociateTargetsWithJob	Associates a group with a continuous job.	Write	job* (p. 1260)		
			thing* (p. 1260)		
			thinggroup* (p. 1260)		
AttachPolicy	Attaches a policy to the specified target.	Permissions management	cert (p. 1261)		
			thinggroup (p. 1260)		
AttachPrincipalPolicy	Attaches the specified policy to the specified principal (certificate or other credential).	Permissions management	cert (p. 1261)		
AttachSecurityProfile	Associates a Device Defender security profile with a thing group or with this account.	Write	securityprofile* (p. 1261)		
			thinggroup (p. 1260)		
AttachThingPrincipal	Attaches the specified principal to the specified thing.	Write			
CancelAuditTask	Cancels an audit that is in progress. The audit can be either scheduled or on-demand.	Write			
CancelCertificateTransfer	Cancels a pending transfer for the specified certificate.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CancelJob	Cancels a job.	Write	job* (p. 1260)		
CancelJobExecution	Cancels a job execution on a particular device.	Write	job* (p. 1260)		
			thing* (p. 1260)		
ClearDefaultAuthorizer	Clears the default authorizer.	Write			
CloseTunnel	Closes a tunnel.	Write	tunnel* (p. 1260)		
				iot:Delete (p. 1262)	
Connect	Connect as the specified client	Write	client* (p. 1260)		
CreateAuthorizer	Creates an authorizer.	Write	authorizer* (p. 1261)		
CreateBillingGroup	Creates a billing group.	Tagging	billinggroup* (p. 1260)		
				aws:RequestTag/ \${TagKey} (p. 1261)	
CreateCertificate Using	Creates an X.509 certificate using the specified certificate signing request.	Write			
				aws:RequestTag/ \${TagKey} (p. 1261)	
CreateDynamicThingGroup	Creates a Dynamic Thing Group	Tagging	dynamicthinggroup* (p. 1260)		
				aws:RequestTag/ \${TagKey} (p. 1261)	
CreateJob	Creates a job.	Write	job* (p. 1260)		
			thing* (p. 1260)		
			thinggroup* (p. 1260)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:RequestTag/ \${TagKey} (p. 1261) aws:TagKeys (p. 1261)	
CreateKeysAndCertificates	Creates a 2048 bit RSA key pair and issues an X.509 certificate using the issued public key.	Write			
CreateOTAUpdate	Creates an OTA update job.	Write	otaupdate* (p. 1261)		
				aws:RequestTag/ \${TagKey} (p. 1261) aws:TagKeys (p. 1261)	
CreatePolicy	Creates an AWS IoT policy.	Write			
CreatePolicyVersion	Creates a new version of the specified AWS IoT policy.	Write	policy* (p. 1261)		
CreateProvisioningClaim	Creates a provisioning claim.	Write	provisioningtemplate* (p. 1261)		
CreateProvisioningTemplate	Creates a fleet provisioning template.	Write	provisioningtemplate* (p. 1261)		
CreateProvisioningTemplateVersion	Creates a new version of a fleet provisioning template.	Write	provisioningtemplate* (p. 1261)		
CreateRoleAlias	Creates a role alias.	Write	role* (p. 1261)		
			rolealias* (p. 1260)		
CreateScheduledAudit	Creates a scheduled audit that is run at a specified time interval.	Tagging	scheduledaudit* (p. 1261)		
				aws:RequestTag/ \${TagKey} (p. 1261) aws:TagKeys (p. 1261)	
CreateSecurityProfile	Creates a Device Defender security profile.	Tagging	securityprofile* (p. 1261)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:RequestTag/ \${TagKey} (p. 1261) aws:TagKeys (p. 1261)	
CreateStream	Creates a new AWS IoT stream	Write	stream*(p. 1261)		
				aws:RequestTag/ \${TagKey} (p. 1261) aws:TagKeys (p. 1261)	
CreateThing	Creates a thing in the thing registry.	Write	thing*(p. 1260)		
			billinggroup (p. 1260)		
CreateThingGroup	Creates a thing group.	Tagging	thinggroup*(p. 1260)		
				aws:RequestTag/ \${TagKey} (p. 1261) aws:TagKeys (p. 1261)	
CreateThingType	Creates a new thing type.	Tagging	thingtype*(p. 1260)		
				aws:RequestTag/ \${TagKey} (p. 1261) aws:TagKeys (p. 1261)	
CreateTopicRule	Creates a rule.	Write	rule*(p. 1261)		
				aws:RequestTag/ \${TagKey} (p. 1261) aws:TagKeys (p. 1261)	
	Deletes the audit configuration associated with the account.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteAuthorizer	Deletes the specified authorizer.	Write	authorizer* (p. 1261)		
DeleteBillingGroup	Deletes the specified billing group.	Tagging	billinggroup* (p. 1260)		
DeleteCACertificate	Deletes a registered CA certificate.	Write	cacert* (p. 1261)		
DeleteCertificate	Deletes the specified certificate.	Write	cert* (p. 1261)		
DeleteDynamicThingGroup	Deletes the specified Dynamic Thing Group.	Tagging	dynamictinggroup* (p. 1260)		
DeleteJob	Deletes a job and its related job executions.	Write	job* (p. 1260)		
DeleteJobExecution	Deletes a job execution.	Write	job* (p. 1260)		
			thing* (p. 1260)		
DeleteOTAUpdate	Deletes an OTA update job.	Write	otaupdate* (p. 1261)		
DeletePolicy	Deletes the specified policy.	Write	policy* (p. 1261)		
DeletePolicyVersion	Deletes the specified version of the specified policy.	Write	policy* (p. 1261)		
DeleteProvisioningTemplate	Deletes a fleet provisioning template.	Write	provisioningtemplate* (p. 1261)		
DeleteProvisioningTemplateVersion	Deletes a fleet provisioning template version.	Write	provisioningtemplate* (p. 1261)		
DeleteRegistrationCode	Deletes a CA certificate registration code.	Write			
DeleteRoleAlias	Deletes the specified role alias.	Write	rolealias* (p. 1260)		
DeleteScheduledAudit	Deletes a scheduled audit.	Write	scheduledaudit* (p. 1261)		
DeleteSecurityProfile	Deletes a Device Defender security profile.	Write	securityprofile* (p. 1261)		
DeleteStream	Deletes a specified stream.	Write	stream* (p. 1261)		
DeleteThing	Deletes the specified thing.	Write	thing* (p. 1260)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteThingGroup	Deletes the specified thing group.	Tagging	thinggroup* (p. 1260)		
DeleteThingShadow	Deletes the specified thing shadow.	Write	thing* (p. 1260)		
DeleteThingType	Deletes the specified thing type.	Tagging	thingtype* (p. 1260)		
DeleteTopicRule	Deletes the specified rule.	Write	rule* (p. 1261)		
DeleteV2LoggingLevel	Deletes the specified v2 logging level.	Write			
DeprecateThingType	Deprecates the specified thing type.	Write	thingtype* (p. 1260)		
DescribeAccountAuditConfigurations	Gets information about audit configurations for the account.	Read			
DescribeAuditTask	Gets information about a Device Defender audit.	Read			
DescribeAuthorizer	Describes an authorizer.	Read	authorizer* (p. 1261)		
DescribeBillingGroup	Gets information about the specified billing group.	Read	billinggroup* (p. 1260)		
DescribeCACertificate	Describes a registered CA certificate.	Read	cacert* (p. 1261)		
DescribeCertificate	Gets information about the specified certificate.	Read	cert* (p. 1261)		
DescribeDefaultAuthorizer	Describes the default authorizer.	Read			
DescribeEndpoint	Returns a unique endpoint specific to the AWS account making the call.	Read			
DescribeEventConfigurations	Returns account event configurations.	Read			
DescribeIndex	Gets information about the specified index.	Read	index* (p. 1260)		
DescribeJob	Describes a job.	Read	job* (p. 1260)		
DescribeJobExecution	Describes a job execution.	Read	job (p. 1260)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			thing (p. 1260)		
DescribeProvisioningTemplate	Returns information about a provisioning template.	Read	provisioningtemplate*(p. 1261)		
DescribeProvisioningTemplateVersion	Returns information about a provisioning template version.	Read	provisioningtemplate*(p. 1261)		
DescribeRoleAlias	Describes a role alias.	Read	rolealias*(p. 1260)		
DescribeScheduledAudit	Gets information about a scheduled audit.	Read	scheduledaudit*(p. 1261)		
DescribeSecurityProfile	Gets information about a Device Defender security profile.	Read	securityprofile*(p. 1261)		
DescribeStream	Gets information about the specified stream.	Read	stream*(p. 1261)		
DescribeThing	Gets information about the specified thing.	Read	thing*(p. 1260)		
DescribeThingGroup	Gets information about the specified thing group.	Read	thinggroup*(p. 1260)		
DescribeThingRegistrationTask	Gets information about the bulk thing registration task.	Read			
DescribeThingType	Gets information about the specified thing type.	Read	thingtype*(p. 1260)		
DescribeTunnel	Describes a tunnel.	Read	tunnel*(p. 1260)		
DetachPolicy	Detaches a policy from the specified target.	Permissions management	cert (p. 1261)		
			thinggroup (p. 1260)		
DetachPrincipalPolicy	Removes the specified policy from the specified certificate.	Permissions management	cert (p. 1261)		
DetachSecurityProfile	Disassociates a Device Defender security profile from a thing group or from this account.	Write	securityprofile*(p. 1261)		
DetachThingPrincipal	Detaches the specified principal from the specified thing.		thinggroup (p. 1260)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DisableTopicRule	Disables the specified rule.	Write	rule* (p. 1261)		
EnableTopicRule	Enables the specified rule.	Write	rule* (p. 1261)		
GetCardinality	Get cardinality for IoT fleet index	Read	index* (p. 1260)		
GetEffectivePolicies	Gets effective policies.	Read	cert (p. 1261)		
GetIndexingConfiguration	Gets current fleet indexing configuration	Read			
GetJobDocument	Gets a job document.	Read	job* (p. 1260)		
GetLoggingOptions	Gets the logging options.	Read			
GetOTAUpdate	Gets the information about the OTA update job.	Read	otaupdate* (p. 1261)		
GetPendingJobExpiry	Gets the list of all jobs for a thing that are not in a terminal state.	Read	thing* (p. 1260)		
GetPercentiles	Get percentiles for IoT fleet index	Read	index* (p. 1260)		
GetPolicy	Gets information about the specified policy with the policy document of the default version.	Read	policy* (p. 1261)		
GetPolicyVersion	Gets information about the specified policy version.	Read	policy* (p. 1261)		
GetRegistrationCode	Gets a registration code used to register a CA certificate with AWS IoT.	Read			
GetStatistics	Get statistics for IoT fleet index	Read	index* (p. 1260)		
GetThingShadow	Gets the thing shadow.	Read	thing* (p. 1260)		
GetTopicRule	Gets information about the specified rule.	Read	rule* (p. 1261)		
GetV2LoggingOptions	Gets v2 logging options.	Read			
ListActiveViolations	Lists the active violations for a given Device Defender security profile or Thing.	List	securityprofile (p. 1261)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			thing (p. 1260)		
ListAttachedPolicies	Lists the policies attached to the specified thing group.	List			
ListAuditFindings	Lists the findings (results) of a Device Defender audit or of the audits performed during a specified time period.	List			
ListAuditTasks	Lists the Device Defender audits that have been performed during a given time period.	List			
ListAuthorizers	Lists the authorizers registered in your account.	List			
ListBillingGroups	Lists all billing groups.	List			
ListCACertificates	Lists the CA certificates registered for your AWS account.	List			
ListCertificates	Lists your certificates.	List			
ListCertificatesBy	List the device certificates signed by the specified CA certificate.	List			
ListIndices	Lists all indices for fleet index	List			
ListJobExecutionsForJob	Lists the job executions for a job.	List	job* (p. 1260)		
ListJobExecutions	Lists the job executions for the specified thing.	List	thing* (p. 1260)		
ListJobs	Lists jobs.	List			
ListOTAUpdates	Lists OTA update jobs in the account.	List			
ListOutgoingCertificates	Lists certificates that are being transferred but not yet accepted.	List			
ListPolicies	Lists your policies.	List			
ListPolicyPrincipals	Lists the principals associated with the specified policy.	List			
ListPolicyVersions	Lists the versions of the specified policy, and identifies the default version.	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions	
ListPrincipalPolicies	Lists the policies attached to the specified principal. If you use an Amazon Cognito identity, the ID needs to be in Amazon Cognito Identity format.	List				
ListPrincipalThings	Lists the things associated with the specified principal.	List				
ListProvisioningTemplateVersions	A list of fleet provisioning template versions.	List	provisioningtemplate* (p. 1261)			
ListProvisioningTemplates	Lists the fleet provisioning templates in your AWS account.	List				
ListRoleAliases	Lists role aliases.	List				
ListScheduledAudits	Lists all of your scheduled audits.	List				
ListSecurityProfiles	Lists the Device Defender security profiles you have created.	List				
ListSecurityProfileTargets	Lists the Device Defender security profiles attached to a target.	List	thinggroup (p. 1260)			
ListStreams	Lists the streams in your account.	List				
ListTagsForResource	Lists all tags for a given resource.	List	billinggroup (p. 1260)			
				dynamicthinggroup (p. 1260)		
				job (p. 1260)		
				otaupdate (p. 1261)		
				rule (p. 1261)		
				scheduledaudit (p. 1261)		
				securityprofile (p. 1261)		
				stream (p. 1261)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			thinggroup (p. 1260)		
			thingtype (p. 1260)		
ListTargetsForPolicy	List targets for the specified policy.	List	policy*	(p. 1261)	
ListTargetsForSecurityProfile	Lists the targets associated with Device Defender security profile.	List	securityprofile*	(p. 1261)	
ListThingGroups	Lists all thing groups.	List			
ListThingGroupsForThing	List thing groups to which the specified thing belongs.	List	thing*	(p. 1260)	
ListThingPrincipals	Lists the principals associated with the specified thing.	List			
ListThingRegistrationTasks	Lists information about bulk thing registration tasks.	List			
ListThingRegistrationTaskTasks	Lists bulk thing registration task tasks.	List			
ListThingTypes	Lists all thing types.	List			
ListThings	Lists all things.	List			
ListThingsInBillingGroup	Lists all things in the specified billing group.	List	billinggroup*	(p. 1260)	
ListThingsInThingGroup	Lists all things in the specified thing group.	List	thinggroup*	(p. 1260)	
ListTopicRules	Lists the rules for the specific topic.	List			
ListTunnels	Lists tunnels.	List			
ListV2LoggingLevels	Lists the v2 logging levels.	List			
ListViolationEvents	Lists the Device Defender security profile violations discovered during the given time period.	List	securityprofile	(p. 1261)	
			thing	(p. 1260)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
OpenTunnel	Opens a tunnel.	Write		aws:RequestTag/\${TagKey} (p. 1261) aws:TagKeys (p. 1261) iot:ThingGroupArn (p. 1262) iot:TunnelDestinationService (p. 1262)	
Publish	Publish to the specified topic.	Write	topic* (p. 1260)		
Receive	Receive from the specified topic.	Write	topic* (p. 1260)		
RegisterCACertificate	Registers a CA certificate with AWS IoT.	Write			
RegisterCertificate	Registers a device certificate with AWS IoT.	Write			
RegisterThing	Registers your thing.	Write			
RejectCertificateTransfer	Rejects a pending certificate transfer.	Write	cert* (p. 1261)		
RemoveThingFromBillingGroup	Removes thing from the specified billing group.	Write	billinggroup* (p. 1260)		
				thing* (p. 1260)	
RemoveThingFromThingGroup	Removes thing from the specified thing group.	Write	thing* (p. 1260)		
				thinggroup* (p. 1260)	
ReplaceTopicRule	Replaces the specified rule.	Write	rule* (p. 1261)		
SearchIndex	Search IoT fleet index	Read	index* (p. 1260)		
SetDefaultAuthorizer	Sets the default authorizer. This will be used if a websocket connection is made without specifying an authorizer.	Permissions management	authorizer* (p. 1261)		
SetDefaultPolicyVersion	Sets the specified version of the specified policy as the policy's default (operative) version.	Permissions management	policy* (p. 1261)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
SetLoggingOptions	Sets the logging options.	Write			
SetV2LoggingLevel	Sets the v2 logging level.	Write			
SetV2LoggingOptions	Sets the v2 logging options.	Write			
StartNextPendingJobExecution	Gets and starts the next pending job execution for a thing.	Write	thing* (p. 1260)		
StartOnDemandAuditTask	Starts an on-demand Device Defender audit.	Write			
StartThingRegistrationTask	Starts a bulk thing registration task.	Write			
StopThingRegistrationTask	Stops a bulk thing registration task.	Write			
Subscribe	Subscribe to the specified TopicFilter.	Write	topicfilter* (p. 1260)		
TagResource	Tag a specified resource	Tagging	billinggroup (p. 1260)		
			dynamicthinggroup (p. 1260)		
			job (p. 1260)		
			otaupdate (p. 1261)		
			rule (p. 1261)		
			scheduledaudit (p. 1261)		
			securityprofile (p. 1261)		
			stream (p. 1261)		
			thinggroup (p. 1260)		
			thingtype (p. 1260)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:RequestTag/ \${TagKey} (p. 1261) aws:TagKeys (p. 1261)	
TestAuthorization	Test the policies evaluation for group policies	Read	cert (p. 1261)		
TestInvokeAuthorizer	Invoke the specified custom authorizer for testing purposes.	Read	authorizer* (p. 1261)		
TransferCertificate	Transfers the specified certificate to the specified AWS account.	Write	cert* (p. 1261)		
UntagResource	Untag a specified resource	Tagging	billinggroup (p. 1260)		
			dynamicthinggroup (p. 1260)		
			job (p. 1260)		
			otaupdate (p. 1261)		
			rule (p. 1261)		
			scheduledaudit (p. 1261)		
			securityprofile (p. 1261)		
			stream (p. 1261)		
			thinggroup (p. 1260)		
			thingtype (p. 1260)		
				aws:TagKeys (p. 1261)	
UpdateAccountAudit	Configures or reconfigures the Device Defender audit settings for this account.	Write			
UpdateAuthorizer	Updates an authorizer	Write	authorizer* (p. 1261)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UpdateBillingGroup	Updates information associated with the specified billing group.	Write	billinggroup* (p. 1260)		
UpdateCACertificate	Updates a registered CA certificate.	Write	cacert* (p. 1261)		
UpdateCertificate	Updates the status of the specified certificate. This operation is idempotent.	Write	cert* (p. 1261)		
UpdateDynamicThingGroup	Updates a Dynamic Thing Group	Write	dynamicthinggroup* (p. 1260)		
UpdateEventConfigurations	Updates event configurations.	Write			
UpdateIndexingConfiguration	Updates fleet indexing configuration	Write			
UpdateJob	Updates a job.	Write	job* (p. 1260)		
UpdateJobExecution	Updates a job execution.	Write	thing* (p. 1260)		
UpdateProvisioningTemplate	Updates a fleet provisioning template	Write	provisioningtemplate* (p. 1261)		
UpdateRoleAlias	Updates the role alias	Write	rolealias* (p. 1260)		
			role (p. 1261)		
UpdateScheduledAudit	Updates a scheduled audit, including what checks are performed and how often the audit takes place.	Write	scheduledaudit* (p. 1261)		
UpdateSecurityProfile	Updates a Device Defender security profile.	Write	securityprofile* (p. 1261)		
UpdateStream	Updates the data for a stream.	Write	stream* (p. 1261)		
UpdateThing	Updates information associated with the specified thing.	Write	thing* (p. 1260)		
UpdateThingGroup	Updates information associated with the specified thing group.	Write	thinggroup* (p. 1260)		
UpdateThingGroupsForThing	Updates the thing groups to which the thing belongs.	Write	thing* (p. 1260)		
			thinggroup (p. 1260)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UpdateThingShadow	Updates the thing shadow.	Write	thing* (p. 1260)		
ValidateSecurityProfile	Validates a Device Defender security profile behaviors specification.	Read			

Resource Types Defined by AWS IoT

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1244\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
client	<code>arn:\${Partition}:iot:\${Region}: \${Account}:client/\${ClientId}</code>	
index	<code>arn:\${Partition}:iot:\${Region}: \${Account}:index/\${IndexName}</code>	
job	<code>arn:\${Partition}:iot:\${Region}: \${Account}:job/\${JobId}</code>	aws:ResourceTag/ \${TagKey} (p. 1261)
tunnel	<code>arn:\${Partition}:iot:\${Region}: \${Account}:tunnel/\${TunnelId}</code>	aws:ResourceTag/ \${TagKey} (p. 1261)
thing	<code>arn:\${Partition}:iot:\${Region}: \${Account}:thing/\${ThingName}</code>	
thinggroup	<code>arn:\${Partition}:iot:\${Region}: \${Account}:thinggroup/\${ThingGroupName}</code>	aws:ResourceTag/ \${TagKey} (p. 1261)
billinggroup	<code>arn:\${Partition}:iot:\${Region}: \${Account}:billinggroup/\${BillingGroupName}</code>	aws:ResourceTag/ \${TagKey} (p. 1261)
dynamicthinggroup	<code>arn:\${Partition}:iot:\${Region}: \${Account}:thinggroup/\${ThingGroupName}</code>	aws:ResourceTag/ \${TagKey} (p. 1261)
thingtype	<code>arn:\${Partition}:iot:\${Region}: \${Account}:thingtype/\${ThingTypeName}</code>	aws:ResourceTag/ \${TagKey} (p. 1261)
topic	<code>arn:\${Partition}:iot:\${Region}: \${Account}:topic/\${TopicName}</code>	
topicfilter	<code>arn:\${Partition}:iot:\${Region}: \${Account}:topicfilter/\${TopicFilter}</code>	
rolealias	<code>arn:\${Partition}:iot:\${Region}: \${Account}:rolealias/\${RoleAlias}</code>	

Resource Types	ARN	Condition Keys
role	arn:\${Partition}:iam::\${Account}:role/\${Role}	
authorizer	arn:\${Partition}:iot:\${Region}: \${Account}:authorizer/\${AuthorizerName}	
policy	arn:\${Partition}:iot:\${Region}: \${Account}:policy/\${PolicyName}	
cert	arn:\${Partition}:iot:\${Region}: \${Account}:cert/\${Certificate}	
cacert	arn:\${Partition}:iot:\${Region}: \${Account}:cacert/\${CACertificate}	
stream	arn:\${Partition}:iot:\${Region}: \${Account}:stream/\${streamId}	aws:ResourceTag/ \${TagKey} (p. 1261)
otaupdate	arn:\${Partition}:iot:\${Region}: \${Account}:otaupdate/\${otaUpdateId}	aws:ResourceTag/ \${TagKey} (p. 1261)
scheduledaudit	arn:\${Partition}:iot:\${Region}: \${Account}:scheduledaudit/\${ScheduleName}	aws:ResourceTag/ \${TagKey} (p. 1261)
securityprofile	arn:\${Partition}:iot:\${Region}: \${Account}:securityprofile/ \${SecurityProfileName}	aws:ResourceTag/ \${TagKey} (p. 1261)
rule	arn:\${Partition}:iot:\${Region}: \${Account}:rule/\${ruleName}	aws:ResourceTag/ \${TagKey} (p. 1261)
provisioningtemplate	arn:\${Partition}:iot:\${Region}: \${Account}:provisioningtemplate/ \${provisioningTemplate}	

Condition Keys for AWS IoT

AWS IoT defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/ \${TagKey}	A tag key that is present in the request that the user makes to IoT.	String
aws:ResourceTag/ \${TagKey}	The tag key component of a tag attached to an IoT resource.	String
aws:TagKeys	The list of all the tag key names associated with the resource in the request.	String

Condition Keys	Description	Type
iot:Delete	The flag indicating whether or not to also delete an IoT Tunnel immediately	Bool
iot:ThingGroupArn	The list of all IoT Thing Group ARNs that the destination IoT Thing belongs to for an IoT Tunnel	String
iot:TunnelDestinationService	The list of all destination services for an IoT Tunnel	String

Actions, Resources, and Condition Keys for AWS IoT 1-Click

AWS IoT 1-Click (service prefix: `iot1click`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS IoT 1-Click \(p. 1262\)](#)
- [Resource Types Defined by AWS IoT 1-Click \(p. 1264\)](#)
- [Condition Keys for AWS IoT 1-Click \(p. 1265\)](#)

Actions Defined by AWS IoT 1-Click

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AssociateDeviceWithPlacement	Associate a device to a placement	Write	project* (p. 1265)		
ClaimDevicesByClaimCode	Claim a batch of devices with a claim code.	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreatePlacement	Create a new placement in a project	Write	project* (p. 1265)		
CreateProject	Create a new project	Write	project* (p. 1265)		
				aws:RequestTag/ \${TagKey} (p. 1265)	
				aws:TagKeys (p. 1265)	
DeletePlacement	Delete a placement from a project	Write	project* (p. 1265)		
DeleteProject	Delete a project	Write	project* (p. 1265)		
DescribeDevice	Describe a device	Read	device* (p. 1265)		
DescribePlacement	Describe a placement	Read	project* (p. 1265)		
DescribeProject	Describe a project	Read	project* (p. 1265)		
DisassociateDeviceFromPlacement	Disassociate a device from a placement	Write	project* (p. 1265)		
FinalizeDeviceClaim	Finalize a device claim	Read	device* (p. 1265)		
				aws:RequestTag/ \${TagKey} (p. 1265)	
				aws:TagKeys (p. 1265)	
GetDeviceMethods	Get available methods of a device	Read	device* (p. 1265)		
GetDevicesInPlacement	Get devices associated to a placement	Read	project* (p. 1265)		
InitiateDeviceClaim	Initialize a device claim	Read	device* (p. 1265)		
InvokeDeviceMethod	Invoke a device method	Write	device* (p. 1265)		
ListDeviceEvents	List past events published by a device	Read	device* (p. 1265)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListDevices	List all devices	List			
ListPlacements	List placements in a project	Read	project* (p. 1265)		
ListProjects	List all projects	List			
ListTagsForResource	Lists the tags (metadata) which you have assigned to the resource.	List	device (p. 1265) project (p. 1265)		
TagResource	Adds to or modifies the tags of the given resource. Tags are metadata which can be used to manage a resource.	Write	device (p. 1265) project (p. 1265) aws:RequestTag/\${TagKey} (p. 1265) aws:TagKeys (p. 1265)		
UnclaimDevice	Unclaim a device	Read	device* (p. 1265)		
UntagResource	Removes the given tags (metadata) from the resource.	Write	device (p. 1265) project (p. 1265) aws:TagKeys (p. 1265)		
UpdateDeviceState	Update device state	Write	device* (p. 1265)		
UpdatePlacement	Update a placement	Write	project* (p. 1265)		
UpdateProject	Update a project	Write	project* (p. 1265)		

Resource Types Defined by AWS IoT 1-Click

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1262\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
device	arn:\${Partition}:iot1click:\${Region}:\${Account}:devices/\${DeviceId}	aws:ResourceTag/\${TagKey} (p. 1265)
project	arn:\${Partition}:iot1click:\${Region}:\${Account}:projects/\${ProjectName}	aws:ResourceTag/\${TagKey} (p. 1265)

Condition Keys for AWS IoT 1-Click

AWS IoT 1-Click defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	A tag key that is present in the request that the user makes to IoT 1-Click.	String
aws:ResourceTag/\${TagKey}	The preface string for a tag key and value pair attached to an IoT 1-Click resource.	String
aws:TagKeys	The list of all the tag key names associated with the IoT 1-Click resource in the request.	String

Actions, Resources, and Condition Keys for AWS IoT Analytics

AWS IoT Analytics (service prefix: `iotanalytics`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS IoT Analytics \(p. 1265\)](#)
- [Resource Types Defined by AWS IoT Analytics \(p. 1269\)](#)
- [Condition Keys for AWS IoT Analytics \(p. 1270\)](#)

Actions Defined by AWS IoT Analytics

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
BatchPutMessage	Puts a batch of messages into the specified channel.	Write	channel* (p. 1269)		
CancelPipelineReprocessing	Cancels reprocessing for the specified pipeline.	Write	pipeline* (p. 1269)		
CreateChannel	Creates a channel.	Write	channel* (p. 1269)		
				aws:RequestTag/\${TagKey} (p. 1270) aws:TagKeys (p. 1270)	
CreateDataset	Creates a dataset.	Write	dataset* (p. 1269)		
				aws:RequestTag/\${TagKey} (p. 1270) aws:TagKeys (p. 1270)	
CreateDatasetContent	Generates content of the specified dataset (by executing the dataset actions).	Write	dataset* (p. 1269)		
CreateDatastore	Creates a datastore.	Write	datastore* (p. 1269)		
				aws:RequestTag/\${TagKey} (p. 1270) aws:TagKeys (p. 1270)	
CreatePipeline	Creates a pipeline.	Write	pipeline* (p. 1269)		
				aws:RequestTag/\${TagKey} (p. 1270)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:TagKeys (p. 1270)	
DeleteChannel	Deletes the specified channel.	Write	channel* (p. 1269)		
DeleteDataset	Deletes the specified dataset.	Write	dataset* (p. 1269)		
DeleteDatasetContent	Deletes the content of the specified dataset.	Write	dataset* (p. 1269)		
DeleteDatastore	Deletes the specified datastore.	Write	datastore* (p. 1269)		
DeletePipeline	Deletes the specified pipeline.	Write	pipeline* (p. 1269)		
DescribeChannel	Describes the specified channel.	Read	channel* (p. 1269)		
DescribeDataset	Describes the specified dataset.	Read	dataset* (p. 1269)		
DescribeDatastore	Describes the specified datastore.	Read	datastore* (p. 1269)		
DescribeLoggingOptions	Describes logging options for the account.	Read			
DescribePipeline	Describes the specified pipeline.	Read	pipeline* (p. 1269)		
GetDatasetContent	Gets the content of the specified dataset.	Read	dataset* (p. 1269)		
ListChannels	Lists the channels for the account.	List			
ListDatasets	Lists the datasets for the account.	List			
ListDatastores	Lists the datastores for the account.	List			
ListPipelines	Lists the pipelines for the account.	List			
ListTagsForResource	Lists the tags (metadata) which you have assigned to the resource.	Read	channel (p. 1269)		
			dataset (p. 1269)		
			datastore (p. 1269)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			pipeline (p. 1269)		
PutLoggingOptions	Puts logging options for the account.	Write			
RunPipelineActivity	Runs the specified pipeline activity.	Read			
SampleChannelData	Samples the specified channel's data.	Read	channel* (p. 1269)		
StartPipelineReprocessing	Starts reprocessing for the specified pipeline.	Write	pipeline* (p. 1269)		
TagResource	Adds to or modifies the tags of the given resource. Tags are metadata which can be used to manage a resource.	Tagging	channel (p. 1269)		
			dataset (p. 1269)		
			datastore (p. 1269)		
			pipeline (p. 1269)		
			aws:RequestTag/ {\$TagKey} (p. 1270)		
			aws:TagKeys (p. 1270)		
UntagResource	Removes the given tags (metadata) from the resource.	Tagging	channel (p. 1269)		
			dataset (p. 1269)		
			datastore (p. 1269)		
			pipeline (p. 1269)		
			aws:RequestTag/ {\$TagKey} (p. 1270)		
			aws:TagKeys (p. 1270)		
UpdateChannel	Updates the specified channel.	Write	channel* (p. 1269)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UpdateDataset	Updates the specified dataset.	Write	dataset* (p. 1269)		
UpdateDatastore	Updates the specified datastore.	Write	datastore* (p. 1269)		
UpdatePipeline	Updates the specified pipeline.	Write	pipeline* (p. 1269)		

Resource Types Defined by AWS IoT Analytics

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 1265\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
channel	arn:\${Partition}:iotanalytics:\${Region}: \${Account}:channel/\${ChannelName}	aws:RequestTag/\${TagKey} (p. 1270) aws:TagKeys (p. 1270) iotanalytics:ResourceTag/\${TagKey} (p. 1270)
dataset	arn:\${Partition}:iotanalytics:\${Region}: \${Account}:dataset/\${DatasetName}	aws:RequestTag/\${TagKey} (p. 1270) aws:TagKeys (p. 1270) iotanalytics:ResourceTag/\${TagKey} (p. 1270)
datastore	arn:\${Partition}:iotanalytics:\${Region}: \${Account}:datastore/\${DatastoreName}	aws:RequestTag/\${TagKey} (p. 1270) aws:TagKeys (p. 1270) iotanalytics:ResourceTag/\${TagKey} (p. 1270)
pipeline	arn:\${Partition}:iotanalytics:\${Region}: \${Account}:pipeline/\${PipelineName}	aws:RequestTag/\${TagKey} (p. 1270) aws:TagKeys (p. 1270) iotanalytics:ResourceTag/\${TagKey} (p. 1270)

Condition Keys for AWS IoT Analytics

AWS IoT Analytics defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
<code>aws:RequestTag/\${TagKey}</code>	A tag key that is present in the request that the user makes to IoT Analytics.	String
<code>aws:TagKeys</code>	The list of all the tag key names associated with the IoT Analytics resource in the request.	String
<code>iotanalytics:ResourceTag/\${TagKey}</code>	The preface string for a tag key and value pair attached to an IoT Analytics resource.	String

Actions, Resources, and Condition Keys for AWS IoT Events

AWS IoT Events (service prefix: `iotevents`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS IoT Events \(p. 1270\)](#)
- [Resource Types Defined by AWS IoT Events \(p. 1272\)](#)
- [Condition Keys for AWS IoT Events \(p. 1273\)](#)

Actions Defined by AWS IoT Events

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
BatchPutMessage	Sends a set of messages to the AWS IoT Events system.	Write	input* (p. 1273)		
BatchUpdateDetector	Update an detector within the AWS IoT Events system.	Write	input* (p. 1273)		
CreateDetectorModel	Creates a detector model.	Write	detectorModel* (p. 1273)		
				aws:RequestTag/\${TagKey} (p. 1273) aws:TagKeys (p. 1273)	
CreateInput	Creates an input.	Write	input* (p. 1273)		
				aws:RequestTag/\${TagKey} (p. 1273) aws:TagKeys (p. 1273)	
DeleteDetectorModel	Deletes a detector model.	Write	detectorModel* (p. 1273)		
DeleteInput	Deletes an input.	Write	input* (p. 1273)		
DescribeDetector	Returns information about the specified detector (instance).	Read	detectorModel* (p. 1273)		
DescribeDetectorModel	Describes a detector model.	Read	detectorModel* (p. 1273)		
DescribeInput	Describes an input.	Read	input* (p. 1273)		
DescribeLoggingOptions	Retrieves the current settings of the AWS IoT Events logging options.	Read			
ListDetectorModelDetectors	Lists all the versions of a detector model. Only the metadata associated with each detector model version is returned.	List	detectorModel* (p. 1273)		
ListDetectorModels	Lists the detector models you have created. Only the metadata	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
	associated with each detector model is returned.				
ListDetectors	Lists detectors (the instances of a detector model).	List	detectorModel* (p. 1273)		
ListInputs	Lists the inputs you have created.	List			
ListTagsForResource	Lists the tags (metadata) which you have assigned to the resource.	Read	detectorModel (p. 1273)		
PutLoggingOptionsEvent	Sets or updates the AWS IoT Events logging options.	Write			
TagResource	Adds to or modifies the tags of the given resource. Tags are metadata which can be used to manage a resource.	Tagging	detectorModel (p. 1273)		
input (p. 1273)					
aws:RequestTag / \${TagKey} (p. 1273)					
UntagResource	Removes the given tags (metadata) from the resource.	Tagging	detectorModel (p. 1273)		
input (p. 1273)					
aws:TagKeys (p. 1273)					
UpdateDetectorModel	Updates a detector model.	Write	detectorModel* (p. 1273)		
UpdateInput	Updates an input.	Write	input* (p. 1273)		
UpdateInputRouting	Updates input routing.	Write	input* (p. 1273)		

Resource Types Defined by AWS IoT Events

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1270\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you

can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
detectorModel	arn:\${Partition}:iotevents:\${Region}:\${Account}:detectorModel/\${DetectorModelName}	aws:ResourceTag/\${TagKey} (p. 1273)
input	arn:\${Partition}:iotevents:\${Region}:\${Account}:input/\${inputName}	aws:ResourceTag/\${TagKey} (p. 1273)

Condition Keys for AWS IoT Events

AWS IoT Events defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	A tag key that is present in the request that the user makes to IoT Events.	String
aws:ResourceTag/\${TagKey}	The tag key by which a tag value is attached to an IoT Events resource.	String
aws:TagKeys	The list of all the tag key names associated with the IoT Events resource in the request.	String

Actions, Resources, and Condition Keys for AWS IoT Greengrass

AWS IoT Greengrass (service prefix: `greengrass`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS IoT Greengrass \(p. 1273\)](#)
- [Resource Types Defined by AWS IoT Greengrass \(p. 1283\)](#)
- [Condition Keys for AWS IoT Greengrass \(p. 1285\)](#)

Actions Defined by AWS IoT Greengrass

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually

allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the **Resource** element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AssociateRoleToGroup	Grants permission to associate a role with a group. The role's permissions must allow Greengrass core Lambda functions and connectors to perform actions in other AWS services.	Write	group* (p. 1283)		
AssociateServiceRoleWithYourAccount	Grants permission to associate a service role with your account. AWS IoT Greengrass uses this role to access your Lambda functions and AWS IoT resources.	Permissions management			
CreateConnectorDefinition	Grants permission to create a connector definition.	Write	aws:RequestTag/\${TagKey} (p. 1285) aws:TagKeys (p. 1285)		
CreateConnectorDefinitionVersion	Grants permission to create a version of an existing connector definition.	Write	connectorDefinition* (p. 1284)		
CreateCoreDefinition	Grants permission to create a core definition.	Write	aws:RequestTag/\${TagKey} (p. 1285) aws:TagKeys (p. 1285)		
CreateCoreDefinitionVersion	Grants permission to create a version of an existing core definition. Greengrass groups must each contain exactly one Greengrass core.	Write	coreDefinition* (p. 1284)		
CreateDeployment	Grants permission to create a deployment.	Write	group* (p. 1283)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateDeviceDefinition	Grants permission to create a device definition.	Write		aws:RequestTag/ \${TagKey} (p. 1285) aws:TagKeys (p. 1285)	
CreateDeviceDefinitionVersion	Grants permission to create a version of an existing device definition.	Write	deviceDefinition* (p. 1284)		
CreateFunctionDefinitionLambda	Grants permission to create a Lambda function definition to be used in a group that contains a list of Lambda functions and their configurations.	Write		aws:RequestTag/ \${TagKey} (p. 1285) aws:TagKeys (p. 1285)	
CreateFunctionDefinitionVersion	Grants permission to create a version of an existing Lambda function definition.	Write	functionDefinition* (p. 1284)		
CreateGroup	Grants permission to create a group.	Write		aws:RequestTag/ \${TagKey} (p. 1285) aws:TagKeys (p. 1285)	
CreateGroupCertificateAuthority	Grants permission to create a CA for a group, or rotate the existing CA.	Write	group* (p. 1283)		
CreateGroupVersion	Grants permission to create a version of a group that has already been defined.	Write	group* (p. 1283)		
CreateLoggerDefinition	Grants permission to create a logger definition.	Write		aws:RequestTag/ \${TagKey} (p. 1285) aws:TagKeys (p. 1285)	
CreateLoggerDefinitionVersion	Grants permission to create a version of an existing logger definition.	Write	loggerDefinition* (p. 1284)		
CreateResourceDefinition	Grants permission to create a resource definition that contains a list of resources to be used in a group.	Write		aws:RequestTag/ \${TagKey} (p. 1285) aws:TagKeys (p. 1285)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateResourceDefinition	Grants permission to create a new resource definition.	Write	resourceDefinition* (p. 1284)		
CreateSoftwareUpdateJob	Grants permission to create an AWS Lambda job that will trigger your Greengrass cores to update the software they are running.	Write			
CreateSubscriptionDefinition	Grants permission to create a new subscription definition.	Write		aws:RequestTag/ \${TagKey} (p. 1285)	
CreateSubscriptionDefinition	Grants permission to create a new subscription definition.	Write	subscriptionDefinition* (p. 1284)		
DeleteConnectorDefinition	Grants permission to delete a connector definition.	Write	connectorDefinition* (p. 1284)		
DeleteCoreDefinition	Grants permission to delete a core definition. Deleting a definition that is currently in use in a deployment affects future deployments.	Write	coreDefinition* (p. 1284)		
DeleteDeviceDefinition	Grants permission to delete a device definition. Deleting a definition that is currently in use in a deployment affects future deployments.	Write	deviceDefinition* (p. 1284)		
DeleteFunctionDefinition	Grants permission to delete a Lambda function definition. Deleting a definition that is currently in use in a deployment affects future deployments.	Write	functionDefinition* (p. 1284)		
DeleteGroup	Grants permission to delete a group that is not currently in use in a deployment.	Write	group* (p. 1283)		
DeleteLoggerDefinition	Grants permission to delete a logger definition. Deleting a definition that is currently in use in a deployment affects future deployments.	Write	loggerDefinition* (p. 1284)		
DeleteResourceDefinition	Grants permission to delete a resource definition.	Write	resourceDefinition* (p. 1284)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteSubscription	Grants permission to delete a subscription definition . Deleting a definition that is currently in use in a deployment affects future deployments.	Write	subscriptionDefinition* (p. 1284)		
DisassociateRoleFromGroup	Grants permission to disassociate the role from a group.	Write	group* (p. 1283)		
DisassociateServiceRoleFromAccount	Grants permission to disassociate the service role from an account. Without a service role, deployments will not work.	Write			
GetAssociatedRole	Grants permission to retrieve the role associated with a group.	Read	group* (p. 1283)		
GetBulkDeploymentStatus	Grants permission to return the status of a bulk deployment.	Read	bulkDeployment* (p. 1283)		
GetConnectivityInfo	Grants permission to retrieve the connectivity information for a core.	Read	connectivityInfo* (p. 1283)		
GetConnectorDefinition	Grants permission to retrieve information about a connector definition.	Read	connectorDefinition* (p. 1284)		
GetConnectorDefinitionInformation	Grants permission to retrieve information about a connector definition version.	Read	connectorDefinition* (p. 1284)		
GetCoreDefinition	Grants permission to retrieve information about a core definition.		coreDefinition* (p. 1284)		
GetCoreDefinitionInformation	Grants permission to retrieve information about a core definition version.	Read	coreDefinition* (p. 1284)		
GetDeploymentStatus	Grants permission to return the status of a deployment.		deployment* (p. 1283) group* (p. 1283)		
GetDeviceDefinition	Grants permission to retrieve information about a device definition.	Read	deviceDefinition* (p. 1284)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetDeviceDefinition	Grants permission to retrieve information about a device definition version.	Read	deviceDefinition* (p. 1284)		
	deviceDefinitionVersion* (p. 1284)				
GetFunctionDefinition	Grants permission to retrieve information about a Lambda function definition, such as its creation time and latest version.	Read	functionDefinition* (p. 1284)		
GetFunctionDefinition	Grants permission to retrieve information about a Lambda function definition version, such as which Lambda functions are included in the version and their configurations.	Read	functionDefinition* (p. 1284)		
	functionDefinitionVersion* (p. 1284)				
GetGroup	Grants permission to retrieve information about a group.	Read	group* (p. 1283)		
GetGroupCertificate	Grants permission to return the public key of the CA associated with a group.	Read	certificateAuthority* (p. 1283)		
	group* (p. 1283)				
GetGroupCertificate	Grants permission to retrieve the configuration for the CA used by a group.	Read	group* (p. 1283)		
GetGroupVersion	Grants permission to retrieve information about a group version.	Read	group* (p. 1283)		
	groupVersion* (p. 1283)				
GetLoggerDefinition	Grants permission to retrieve information about a logger definition.	Read	loggerDefinition* (p. 1284)		
GetLoggerDefinition	Grants permission to retrieve information about a logger definition version.	Read	loggerDefinition* (p. 1284)		
	loggerDefinitionVersion* (p. 1284)				
GetResourceDefinition	Grants permission to retrieve information about a resource definition, such as its creation time and latest version.	Read	resourceDefinition* (p. 1284)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetResourceDefinitionInformation	Grants permission to retrieve information about a resource definition version, such as which resources are included in the version.	Read	resourceDefinition* (p. 1284)		
			resourceDefinitionVersion* (p. 1284)		
GetServiceRoleForService	Grants permission to retrieve the service role that is attached to an account.	Read			
GetSubscriptionDefinitionInformation	Grants permission to retrieve information about a subscription definition.	Read	subscriptionDefinition* (p. 1284)		
GetSubscriptionDefinitionVersionInformation	Grants permission to retrieve information about a subscription definition version.	Read	subscriptionDefinition* (p. 1284)		
subscriptionDefinitionVersion* (p. 1284)					
ListBulkDeploymentInformation	Grants permission to retrieve a paginated list of the deployments that have been started in a bulk deployment operation and their current deployment status.	List	bulkDeployment* (p. 1283)		
ListBulkDeployments	Grants permission to retrieve a list of bulk deployments.	List			
ListConnectorDefinitionInformation	Grants permission to list the versions of a connector definition.	List	connectorDefinition* (p. 1284)		
ListConnectorDefinitions	Grants permission to retrieve a list of connector definitions.	List			
ListCoreDefinitionVersions	Grants permission to list the versions of a core definition.	List	coreDefinition* (p. 1284)		
ListCoreDefinitions	Grants permission to retrieve a list of core definitions.	List			
ListDeployments	Grants permission to retrieve a list of all deployments for a group.	List	group* (p. 1283)		
ListDeviceDefinitionInformation	Grants permission to list the versions of a device definition.	List	deviceDefinition* (p. 1284)		
ListDeviceDefinitions	Grants permission to retrieve a list of device definitions.	List			
ListFunctionDefinitionVersions	Grants permission to list the versions of a Lambda function definition.	List	functionDefinition* (p. 1284)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions	
ListFunctionDefinitions	Grants permission to retrieve a list of Lambda function definitions.	List				
ListGroupCertificates	Grants permission to retrieve a list of current CAs for a group.	List	group* (p. 1283)			
ListGroupVersions	Grants permission to list the versions of a group.	List	group* (p. 1283)			
ListGroups	Grants permission to retrieve a list of groups.	List				
ListLoggerDefinitions	Grants permission to list the versions of a logger definition.	List	loggerDefinition* (p. 1284)			
ListLoggerDefinitionVersions	Grants permission to retrieve a list of logger definitions.	List				
ListResourceDefinitionVersions	Grants permission to list the versions of a resource definition.	List	resourceDefinition* (p. 1284)			
ListResourceDefinitionVersionsForSubscription	Grants permission to retrieve a list of resource definitions.	List				
ListSubscriptionDefinitionVersions	Grants permission to list the versions of a subscription definition.	List	subscriptionDefinition* (p. 1284)			
ListSubscriptionDefinitionVersionsForResource	Grants permission to retrieve a list of subscription definitions.	List				
ListTagsForResource	Grants permission to list the tags for a resource.	List	bulkDeployment (p. 1283)			
				connectorDefinition (p. 1284)		
				coreDefinition (p. 1284)		
				deviceDefinition (p. 1284)		
				functionDefinition (p. 1284)		
				group (p. 1283)		
				loggerDefinition (p. 1284)		
				resourceDefinition (p. 1284)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			subscriptionDefinition (p. 1284)		
				aws:RequestTag/ \${TagKey} (p. 1285)	
				aws:TagKeys (p. 1285)	
ResetDeployment	Grants permission to reset a group's deployments.	Write	group* (p. 1283)		
StartBulkDeployment	Grants permission to deploy multiple groups in one operation.	Write		aws:RequestTag/ \${TagKey} (p. 1285)	
				aws:TagKeys (p. 1285)	
StopBulkDeployment	Grants permission to stop the execution of a bulk deployment.	Write	bulkDeployment* (p. 1283)		
TagResource	Grants permission to add tags to a resource.	Tagging	bulkDeployment (p. 1283)		
			connectorDefinition (p. 1284)		
			coreDefinition (p. 1284)		
			deviceDefinition (p. 1284)		
			functionDefinition (p. 1284)		
			group (p. 1283)		
			loggerDefinition (p. 1284)		
			resourceDefinition (p. 1284)		
			subscriptionDefinition (p. 1284)		
				aws:RequestTag/ \${TagKey} (p. 1285)	
				aws:TagKeys (p. 1285)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UntagResource	Grants permission to remove tags from a resource.	Tagging	bulkDeployment (p. 1283)		
			connectorDefinition (p. 1284)		
			coreDefinition (p. 1284)		
			deviceDefinition (p. 1284)		
			functionDefinition (p. 1284)		
			group (p. 1283)		
			loggerDefinition (p. 1284)		
			resourceDefinition (p. 1284)		
			subscriptionDefinition (p. 1284)		
UpdateConnectivityInfo	Grants permission to update the connectivity information for a Greengrass core. Any devices that belong to the group that has this core will receive this information in order to find the location of the core and connect to it.	Write	connectivityInfo* (p. 1283)		
UpdateConnectorDefinition	Grants permission to update a connector definition.	Write	connectorDefinition* (p. 1284)		
UpdateCoreDefinition	Grants permission to update a core definition.	Write	coreDefinition* (p. 1284)		
UpdateDeviceDefinition	Grants permission to update a device definition.	Write	deviceDefinition* (p. 1284)		
UpdateFunctionDefinition	Grants permission to update a Lambda function definition.	Write	functionDefinition* (p. 1284)		
UpdateGroup	Grants permission to update a group.	Write	group* (p. 1283)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UpdateGroupCertificateExpiration	Grants permission to update the certificate expiration time for a group.	Write	group* (p. 1283)		
UpdateLoggerDefinition	Grants permission to update a logger definition.	Write	loggerDefinition* (p. 1284)		
UpdateResourceDefinition	Grants permission to update a resource definition.	Write	resourceDefinition* (p. 1284)		
UpdateSubscriptionDefinition	Grants permission to update a subscription definition.	Write	subscriptionDefinition* (p. 1284)		

Resource Types Defined by AWS IoT Greengrass

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 1273\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
connectivityInfo	arn:\${Partition}:greengrass:\${Region}: \${Account}:/greengrass/things/\${ThingName}/connectivityInfo	
artifact	arn:\${Partition}:greengrass:\${Region}: \${Account}:/greengrass/groups/\${GroupId}/deployments/\${DeploymentId}/artifacts/lambda/\${ArtifactId}	
certificateAuthority	arn:\${Partition}:greengrass:\${Region}: \${Account}:/greengrass/groups/\${GroupId}/certificateauthorities/\${CertificateAuthorityId}	
deployment	arn:\${Partition}:greengrass:\${Region}: \${Account}:/greengrass/groups/\${GroupId}/deployments/\${DeploymentId}	
bulkDeployment	arn:\${Partition}:greengrass:\${Region}: \${Account}:/greengrass/bulk/deployments/\${BulkDeploymentId}	aws:ResourceTag/\${TagKey} (p. 1285)
group	arn:\${Partition}:greengrass:\${Region}: \${Account}:/greengrass/groups/\${GroupId}	aws:ResourceTag/\${TagKey} (p. 1285)
groupVersion	arn:\${Partition}:greengrass:\${Region}: \${Account}:/greengrass/groups/\${GroupId}/versions/\${VersionId}	

Resource Types	ARN	Condition Keys
coreDefinition	arn:\${Partition}:greengrass:\${Region}: \${Account}:/greengrass/definition/cores/\${CoreDefinitionId}	aws:ResourceTag/\${TagKey} (p. 1285)
coreDefinitionVersion	arn:\${Partition}:greengrass:\${Region}: \${Account}:/greengrass/definition/cores/\${CoreDefinitionId}/versions/\${VersionId}	
deviceDefinition	arn:\${Partition}:greengrass:\${Region}: \${Account}:/greengrass/definition/devices/\${DeviceDefinitionId}	aws:ResourceTag/\${TagKey} (p. 1285)
deviceDefinitionVersion	arn:\${Partition}:greengrass:\${Region}: \${Account}:/greengrass/definition/devices/\${DeviceDefinitionId}/versions/\${VersionId}	
functionDefinition	arn:\${Partition}:greengrass:\${Region}: \${Account}:/greengrass/definition/functions/\${FunctionDefinitionId}	aws:ResourceTag/\${TagKey} (p. 1285)
functionDefinitionVersion	arn:\${Partition}:greengrass:\${Region}: \${Account}:/greengrass/definition/functions/\${FunctionDefinitionId}/versions/\${VersionId}	
subscriptionDefinition	arn:\${Partition}:greengrass:\${Region}: \${Account}:/greengrass/definition/subscriptions/\${SubscriptionDefinitionId}	aws:ResourceTag/\${TagKey} (p. 1285)
subscriptionDefinitionVersion	arn:\${Partition}:greengrass:\${Region}: \${Account}:/greengrass/definition/subscriptions/\${SubscriptionDefinitionId}/versions/\${VersionId}	
loggerDefinition	arn:\${Partition}:greengrass:\${Region}: \${Account}:/greengrass/definition/loggers/\${LoggerDefinitionId}	aws:ResourceTag/\${TagKey} (p. 1285)
loggerDefinitionVersion	arn:\${Partition}:greengrass:\${Region}: \${Account}:/greengrass/definition/loggers/\${LoggerDefinitionId}/versions/\${VersionId}	
resourceDefinition	arn:\${Partition}:greengrass:\${Region}: \${Account}:/greengrass/definition/resources/\${ResourceDefinitionId}	aws:ResourceTag/\${TagKey} (p. 1285)
resourceDefinitionVersion	arn:\${Partition}:greengrass:\${Region}: \${Account}:/greengrass/definition/resources/\${ResourceDefinitionId}/versions/\${VersionId}	
connectorDefinition	arn:\${Partition}:greengrass:\${Region}: \${Account}:/greengrass/definition/connectors/\${ConnectorDefinitionId}	aws:ResourceTag/\${TagKey} (p. 1285)

Resource Types	ARN	Condition Keys
connectorDefinition	<code>arn:\${Partition}:greengrass:\${Region}: \${Account}:/greengrass/definition/ connectors/\${ConnectorDefinitionId}/ versions/\${VersionId}</code>	

Condition Keys for AWS IoT Greengrass

AWS IoT Greengrass defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:CurrentTime	Filters access by checking date/time conditions for the current date and time.	Date
aws:EpochTime	Filters access by checking date/time conditions for the current date and time in epoch or Unix time.	Date
aws:MultiFactorAuthAge	Filters access by checking how long ago (in seconds) the security credentials validated by multi-factor authentication (MFA) in the request were issued using MFA.	Numeric
aws:MultiFactorAuthPreferred	Filters access by checking whether multi-factor authentication (MFA) was used to validate the temporary security credentials that made the current request.	Boolean
aws:RequestTag/\${TagKey}	Filters create requests based on the allowed set of values for each of the mandatory tags.	String
aws:ResourceTag/\${TagKey}	Filters actions based on the tag value associated with the resource.	String
aws:SecureTransport	Filters access by checking whether the request was sent using SSL.	Boolean
aws:TagKeys	Filters create requests based on the presence of mandatory tags in the request.	String
aws:UserAgent	Filters access by the requester's client application.	String

Actions, Resources, and Condition Keys for AWS IoT SiteWise

AWS IoT SiteWise (service prefix: `iotsitewise`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).

- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS IoT SiteWise \(p. 1286\)](#)
- [Resource Types Defined by AWS IoT SiteWise \(p. 1291\)](#)
- [Condition Keys for AWS IoT SiteWise \(p. 1292\)](#)

Actions Defined by AWS IoT SiteWise

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AssociateAssets	Associate child assets to the parent via specified model hierarchy.	Write	asset* (p. 1291)		
AssociateViewEntityWithGroup	Associates the specified entities within a view for the specified group.	Write	group* (p. 1292)		
			view* (p. 1292)		
			asset (p. 1291)		
BatchAssociateProjectsToAssets	Grants permission to associate assets to a specified project.	Write	project* (p. 1292)		
BatchDisassociateAssetsFromProject	Grants permission to disassociate assets from a specified project.	Write	project* (p. 1292)		
BatchPutAssetProperties	Puts batch of property values for the specified properties.	Write	asset* (p. 1291)		
CreateAccessPolicy	Grants permission to create access policy for a specified portal or a project.	Permissions management	portal (p. 1292)		
			project (p. 1292)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateAsset	Creates an asset.	Write	asset-model* (p. 1292)		
CreateAssetModel	Creates an asset model.	Write			
CreateAssetTemplate	Creates an asset template.	Write			
CreateDashboard	Grants permission to create a dashboard within a specified project.	Write	project* (p. 1292)		
CreateGroup	Creates a group.	Write			
CreateMeasurementDataStore	Registers a measurement data store.	Write			
CreateMetricType	Creates a metric type.	Write			
CreatePortal	Grants permission to create a portal.	Write			sso:CreateManagedApplication
CreateProject	Grants permission to create a project within a specified portal.	Write	portal* (p. 1292)		
createView	Creates a view.	Write			
DeleteAccessPolicy	Grants permission to delete a specified access policy.	Permissions	access-policy* (p. 1292)		
DeleteAsset	Deletes the specified asset.	Write	asset* (p. 1291)		
DeleteAssetModel	Deletes the specified asset model.	Write	asset-model* (p. 1292)		
DeleteAssetTemplate	Deletes the specified asset template.	Write	asset-template* (p. 1292)		
DeleteDashboard	Grants permission to delete a specified dashboard.	Write	dashboard* (p. 1292)		
DeleteGroup	Deletes the specified group.	Write	group* (p. 1292)		
DeleteMeasurementDataStore	Deregisters the specified measurement data store.	Write	measurement-data-store* (p. 1292)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteMetricType	Deletes the specified metric type.	Write	metric-type* (p. 1292)		
DeletePortal	Grants permission to delete a specified portal.	Write	portal* (p. 1292)		sso:DeleteManagedApplication
DeleteProject	Grants permission to delete a specified project.	Write	project* (p. 1292)		
DeleteView	Deletes the specified view.	Write	view* (p. 1292)		
DeregisterViewEntries	Deregisters the specified assets and groups from the specified view.	Write	view* (p. 1292)		
			asset (p. 1291)		
			group (p. 1292)		
DescribeAccessPolicy	Grants permission to describe a specified access policy.	Permissions management	access-policy* (p. 1292)		
DescribeAsset	Describes the specified asset.		asset* (p. 1291)		
DescribeAssetModel	Describes the specified asset model.	Read	asset-model* (p. 1292)		
DescribeAssetProperty	Describes the specified asset property.	Read	asset* (p. 1291)		
DescribeAssetTemplate	Describes the specified asset template.	Read	asset-template* (p. 1292)		
DescribeAssets	Describes the specified assets.	Read	asset* (p. 1291)		
DescribeDashboard	Grants permission to describe a specified dashboard.	Read	dashboard* (p. 1292)		
DescribeGroups	Describes the groups for the account.	Read	group* (p. 1292)		
DescribeLoggingOptions	Describes the logging options for the account.	Read			
DescribeMeasurementDataStores	Describes the specified measurement data stores.	Read	measurement-data-store* (p. 1292)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeMetricType	Describes the metric types for the account.	Read	metric-type* (p. 1292)		
DescribePortal	Grants permission to describe a specified portal.	Read	portal* (p. 1292)		
DescribeProject	Grants permission to describe a specified project.	Read	project* (p. 1292)		
DescribeViews	Describes the specified views.	Read	view* (p. 1292)		
DisassociateAssets	Disassociate child assets from the parent for specified model hierarchy.	Write	asset* (p. 1291)		
DisassociateViewEntities	Disassociates the specified entities within a view for the specified group.	Write	group* (p. 1292) view* (p. 1292) asset (p. 1291)		
GetAssetPropertyValue	Fetches the aggregated property values for the specified property.	Read	asset* (p. 1291)		
GetAssetPropertyForTime	Fetches the latest property value for the specified property.	Read	asset* (p. 1291)		
GetAssetPropertyHistoryForTime	Fetches the property value history for the specified property.	Read	asset* (p. 1291)		
GetMeasurementForTime	Fetches the measurement data for the specified measurement and time interval.	Read	measurement* (p. 1292)		
GetMetricData	Fetches the metric data for the specified metric and time interval.	Read	metric* (p. 1292)		
ListAccessPolicies	Grants permission to list access policies for a specified portal or a project.	Permissions management	portal (p. 1292) project (p. 1292)		
ListAssetModels	Lists the asset models for the account.	List			
ListAssetTemplates	Lists the asset templates for the account.	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListAssets	Lists the assets for the account.	List	asset-model (p. 1292)		
ListAssociatedAssets	Lists the assets associated to the parent via specified model hierarchy.	List	asset* (p. 1291)		
ListDashboards	Grants permission to list dashboards within a specified project.	List	project* (p. 1292)		
ListGroups	Lists the groups for the account.	List			
ListMeasurementStores	Lists the measurement data stores for the account.	List			
ListMeasurementStreams	Lists the measurement data streams for the specified measurement data store.	List	measurement-data-store* (p. 1292)		
ListMetricTypes	Lists the metric types for the account.	List			
ListPortals	Grants permission to list portals in the account.	List			
ListProjectAssets	Grants permission to list assets associated with a specified project.	List	project* (p. 1292)		
ListProjects	Grants permission to list projects within a specified portal.	List	portal* (p. 1292)		
ListViewEntities	Lists the assets and groups for the specified view.	List	view* (p. 1292)		
			asset (p. 1291)		
			group (p. 1292)		
ListViews	Lists the views for the account.	List			
PutLoggingOptions	Sets the logging options.	Write			
RegisterViewEntities	Registers the specified assets and groups with the specified view.	Write	view* (p. 1292)		
			asset (p. 1291)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			group (p. 1292)		
UpdateAccessPolicy	Grants permission to update a specified access policy.	Permissions management	access-policy* (p. 1292)		
UpdateAsset	Updates the specified asset.	Write	asset* (p. 1291)		
UpdateAssetModel	Updates the specified asset model.	Write	asset-model* (p. 1292)		
UpdateAssetProperty	Updates the specified asset property.	Write	asset* (p. 1291)		
UpdateAssetTemplate	Updates the specified asset template.	Write	asset-template* (p. 1292)		
UpdateDashboard	Grants permission to update a specified dashboard.	Write	dashboard* (p. 1292)		
UpdateGroup	Updates the specified group.	Write	group* (p. 1292)		
UpdateMeasurementStore	Updates metadata about the measurement data store.	Write	measurement-data-store* (p. 1292)		
UpdatePortal	Grants permission to update a specified portal.	Write	portal* (p. 1292)		
UpdateProject	Grants permission to update a specified project.	Write	project* (p. 1292)		
UpdateView	Updates the specified view.	Write	view* (p. 1292)		

Resource Types Defined by AWS IoT SiteWise

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1286\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
asset	arn:\${Partition}:iotsitewise:\${Region}: \${Account}:asset/\${AssetId}	

Resource Types	ARN	Condition Keys
asset-template	arn:\${Partition}:iotsitewise:\${Region}: \${Account}:asset-template/\${AssetTemplateId}	
asset-model	arn:\${Partition}:iotsitewise:\${Region}: \${Account}:asset-model/\${AssetModelId}	
gateway	arn:\${Partition}:iotsitewise:\${Region}: \${Account}:gateway/\${GatewayId}	
group	arn:\${Partition}:iotsitewise:\${Region}: \${Account}:group/\${GroupId}	
measurement	arn:\${Partition}:iotsitewise:\${Region}: \${Account}:measurement/\${MeasurementId}	
measurement-data-store	arn:\${Partition}:iotsitewise:\${Region}: \${Account}:measurement-data-store/ \${MeasurementDatastoreId}	
metric	arn:\${Partition}:iotsitewise:\${Region}: \${Account}:metric/\${MetricId}	
metric-type	arn:\${Partition}:iotsitewise:\${Region}: \${Account}:metric-type/\${MetricTypeId}	
view	arn:\${Partition}:iotsitewise:\${Region}: \${Account}:view/\${ViewId}	
portal	arn:\${Partition}:iotsitewise:\${Region}: \${Account}:portal/\${PortalId}	
project	arn:\${Partition}:iotsitewise:\${Region}: \${Account}:project/\${ProjectId}	
dashboard	arn:\${Partition}:iotsitewise:\${Region}: \${Account}:dashboard/\${DashboardId}	
access-policy	arn:\${Partition}:iotsitewise:\${Region}: \${Account}:access-policy/\${AccessPolicyId}	

Condition Keys for AWS IoT SiteWise

AWS IoT SiteWise defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
iotsitewise:assetHierarchy	String of asset IDs in the asset hierarchy separated by forward slash.	String

Condition Keys	Description	Type
iotsitewise:childAssetId	ID of an asset being associated as a child to another asset.	String
iotsitewise:group	Group ID.	String
iotsitewise:portal	Portal ID.	String
iotsitewise:project	Project ID.	String
iotsitewise:propertyId	Property ID.	String
iotsitewise:user	User ID.	String

Actions, Resources, and Condition Keys for AWS IoT Things Graph

AWS IoT Things Graph (service prefix: `iotthingsgraph`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS IoT Things Graph \(p. 1293\)](#)
- [Resource Types Defined by AWS IoT Things Graph \(p. 1298\)](#)
- [Condition Keys for AWS IoT Things Graph \(p. 1298\)](#)

Actions Defined by AWS IoT Things Graph

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AssociateEntityToThing	Associates a device with a concrete thing that is in the user's registry. A thing can be associated with only one device at a time. If you associate a thing with a new device id, its previous association will be removed.	Write			iot:DescribeThing iot:DescribeThingGroup
CreateFlowTemplate	Creates a workflow template. Workflows can be created only in the user's namespace. (The public namespace contains only entities.) The workflow can contain only entities in the specified namespace. The workflow is validated against the entities in the latest version of the user's namespace unless another namespace version is specified in the request.	Write			
CreateSystemInstance	Creates an instance of a system with specified configurations and Things.	Tagging		aws:RequestTag/\${TagKey} (p. 1298) aws:TagKeys (p. 1298)	
CreateSystemTemplate	Creates a system. The system is validated against the entities in the latest version of the user's namespace unless another namespace version is specified in the request.	Write			
DeleteFlowTemplate	Deletes a workflow. Any new system or system instance that contains this workflow will fail to update or deploy. Existing system instances that contain the workflow will continue to run (since they use a snapshot of the workflow taken at the time of deploying the system instance).	Write	Workflow* (p. 1298)		
DeleteNamespace	Deletes the specified namespace. This action deletes all of the entities in the namespace. Delete the systems and flows in the namespace before performing this action.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteSystemInstance	Deletes a system instance. Only instances that have never been deployed, or that have been undeployed from the target can be deleted. Users can create a new system instance that has the same ID as a deleted system instance.	Write	SystemInstance* (p. 1298)		
DeleteSystemTemplate	Deletes a system. New system instances can't contain the system after its deletion. Existing system instances that contain the system will continue to work because they use a snapshot of the system that is taken when it is deployed.	Write	System* (p. 1298)		
DeploySystemInstance	Deploys the system instance to the target specified in CreateSystemInstance .	Write	SystemInstance* (p. 1298)		
DeprecateFlowTemplate	Deprecates the specified workflow. This action marks the workflow for deletion. Deprecated flows can't be deployed, but existing system instances that use the flow will continue to run.	Write	Workflow* (p. 1298)		
DeprecateSystemTemplate	Deprecates the specified system.	Write	System* (p. 1298)		
DescribeNamespace	Gets the latest version of the user's namespace and the public version that it is tracking.	Read			
DissociateEntityFromThing	Dissociates a device entity from a Thing . The action takes only the type of the entity that you need to dissociate because only one entity of a particular type can be associated with a thing.	Write			iot:DescribeThing iot:DescribeThingGroup
GetEntities	Gets descriptions of the specified entities. Uses the latest version of the user's namespace by default.	Read			
GetFlowTemplate	Gets the latest version of the DefinitionDocument and FlowTemplateSummary for the specified workflow.	Read	Workflow* (p. 1298)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetFlowTemplate	Gets revisions of the specified Workflow . Only the last 100 revisions are stored. If the workflow has been deprecated, this action will return revisions that occurred before the deprecation. This action won't work for workflows that have been deleted.	Read	Workflow* (p. 1298)		
GetNamespaceDeletionStatus	Gets the status of a namespace deletion task .	Read			
GetSystemInstance	Gets a system instance.	Read	SystemInstance* (p. 1298)		
GetSystemTemplate	Gets a system.	Read	System* (p. 1298)		
GetSystemTemplateRevisions	Gets revisions made to the specified system template. Only the previous 100 revisions are stored. If the system has been deprecated, this action will return the revisions that occurred before its deprecation. This action won't work with systems that have been deleted.	Read	System* (p. 1298)		
GetUploadStatus	Gets the status of the specified upload.	Read			
ListFlowExecution	Lists details of a single workflow execution .	List			
ListTagsForResource	Lists all tags for a given resource	List	SystemInstance (p. 1298)		
SearchEntities	Searches for entities of the specified type. You can search for entities in your namespace and the public namespace that you're tracking.	Read			
SearchFlowExecution	Searches for workflow executions of a system instance	Read	SystemInstance* (p. 1298)		
SearchFlowTemplate	Searches for summary information about workflows.	Read			
SearchSystemInstances	Searches for system instances in the user's account.	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
SearchSystemTemplate	Searches for summary information about systems in the user's account. You can filter by the ID of a workflow to return only systems that use the specified workflow.	Read			
SearchThings	Searches for things associated with the specified entity. You can search by both device and device model.	Read			
TagResource	Tag a specified resource	Tagging	SystemInstance (p. 1298)		
				aws:RequestTag/\${TagKey} (p. 1298)	aws:TagKeys (p. 1298)
UndeploySystemTemplate	Removes the system instance and associated triggers from the target.	Write	SystemInstance* (p. 1298)		
UntagResource	Untag a specified resource	Tagging	SystemInstance (p. 1298)		
				aws:TagKeys (p. 1298)	
UpdateFlowTemplate	Updates the specified workflow. All deployed systems and system instances that use the workflow will see the changes in the flow when it is redeployed. The workflow can contain only entities in the specified namespace.	Write	Workflow* (p. 1298)		
UpdateSystemTemplate	Updates the specified system. You don't need to run this action after updating a workflow. Any system instance that uses the system will see the changes in the system when it is redeployed.	Write	System* (p. 1298)		
UploadEntityDefinitions	Asynchronously uploads one or more entity definitions to the user's namespace.	Write			

Resource Types Defined by AWS IoT Things Graph

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 1293\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
Workflow	arn:\${Partition}:iotthingsgraph:\${Region}: \${Account}:Workflow/\${NamespacePath}	
System	arn:\${Partition}:iotthingsgraph:\${Region}: \${Account}:System/\${NamespacePath}	
SystemInstance	arn:\${Partition}:iotthingsgraph:\${Region}: \${Account}:Deployment/\${NamespacePath}	aws:ResourceTag/\${TagKey} (p. 1298)

Condition Keys for AWS IoT Things Graph

AWS IoT Things Graph defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Filters access by a key that is present in the request the user makes to the thingsgraph service.	String
aws:ResourceTag/\${TagKey}	Filters access by a tag key and value pair.	String
aws:TagKeys	Filters access by the list of all the tag key names present in the request the user makes to the thingsgraph service.	String

Actions, Resources, and Condition Keys for AWS IQ

AWS IQ (service prefix: `iq`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).

Topics

- [Actions Defined by AWS IQ \(p. 1299\)](#)
- [Resource Types Defined by AWS IQ \(p. 1299\)](#)

- [Condition Keys for AWS IQ \(p. 1299\)](#)

Actions Defined by AWS IQ

AWS IQ has no API operations that can be used in the `Actions` element of an IAM policy statement. To allow access to AWS IQ, specify “`Action`”: “`iq:*`” in your policy.

Resource Types Defined by AWS IQ

AWS IQ does not support specifying a resource ARN in the `Resource` element of an IAM policy statement. To allow access to AWS IQ, specify “`Resource`”: “`*`” in your policy.

Condition Keys for AWS IQ

IQ has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for AWS IQ Permissions

AWS IQ Permissions (service prefix: `iq-permission`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).

Topics

- [Actions Defined by AWS IQ Permissions \(p. 1299\)](#)
- [Resource Types Defined by AWS IQ Permissions \(p. 1299\)](#)
- [Condition Keys for AWS IQ Permissions \(p. 1299\)](#)

Actions Defined by AWS IQ Permissions

AWS IQ Permissions has no API operations that can be used in the `Actions` element of an IAM policy statement. To allow access to AWS IQ Permissions, specify “`Action`”: “`iq-permission:*`” in your policy.

Resource Types Defined by AWS IQ Permissions

AWS IQ Permissions does not support specifying a resource ARN in the `Resource` element of an IAM policy statement. To allow access to AWS IQ Permissions, specify “`Resource`”: “`*`” in your policy.

Condition Keys for AWS IQ Permissions

IQ Permission has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Amazon Kendra

Amazon Kendra (service prefix: `kendra`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).

- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Kendra \(p. 1300\)](#)
- [Resource Types Defined by Amazon Kendra \(p. 1302\)](#)
- [Condition Keys for Amazon Kendra \(p. 1302\)](#)

Actions Defined by Amazon Kendra

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
BatchDeleteDocument	Batch Delete document	Write	index* (p. 1302)		
BatchPutDocument	Batch put document	Write	index* (p. 1302)		
CreateDataSource	Create a data source	Write	index* (p. 1302)		
CreateFaq	Create an Faq	Write	index* (p. 1302)		
CreateIndex	Create an Index	Write			
DeleteFaq	Delete an Faq	Write	faq* (p. 1302)		
			index* (p. 1302)		
DeleteIndex	Delete an Index	Write	index* (p. 1302)		
DescribeDataSource	Describe a data source	Read	data-source* (p. 1302)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			index* (p. 1302)		
DescribeFaq	Describe an Faq	Read	faq* (p. 1302)		
			index* (p. 1302)		
DescribeIndex	Describe an Index	Read	index* (p. 1302)		
ListDataSourceSyncJobs	Get Data Source sync job history	List	data-source* (p. 1302)		
			index* (p. 1302)		
ListDataSources	List the data sources	List	index* (p. 1302)		
ListFaqs	List the Faqs	List	index* (p. 1302)		
ListIndices	List the indexes	List			
Query	Query documents and Faqs	Read	index* (p. 1302)		
StartDataSourceSyncJob	Start Data Source sync job	Write	data-source* (p. 1302)		
			index* (p. 1302)		
StopDataSourceSyncJob	Stop Data Source sync job	Write	data-source* (p. 1302)		
			index* (p. 1302)		
SubmitFeedback	Send feedback about a query results	Write	index* (p. 1302)		
UpdateDataSource	Update a data source	Write	data-source* (p. 1302)		
			index* (p. 1302)		
UpdateIndex	Update an Index	Write	index* (p. 1302)		

Resource Types Defined by Amazon Kendra

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1300\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
<code>index</code>	<code>arn:\${Partition}:kendra:\${Region}: \${Account}:index/\${IndexId}</code>	
<code>data-source</code>	<code>arn:\${Partition}:kendra:\${Region}: \${Account}:index/\${IndexId}/data-source/ \${DataSourceId}</code>	
<code>faq</code>	<code>arn:\${Partition}:kendra:\${Region}: \${Account}:index/\${IndexId}/faq/\${FaqId}</code>	

Condition Keys for Amazon Kendra

Kendra has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for AWS Key Management Service

AWS Key Management Service (service prefix: `kms`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Key Management Service \(p. 1302\)](#)
- [Resource Types Defined by AWS Key Management Service \(p. 1312\)](#)
- [Condition Keys for AWS Key Management Service \(p. 1312\)](#)

Actions Defined by AWS Key Management Service

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your

policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CancelKeyDeletion	Controls permission to cancel the scheduled deletion of a customer master key.	Write	key* (p. 1312)		
				kms:CallerAccount (p. 1313)	
				kms:ViaService (p. 1314)	
ConnectCustomKeyStores	Controls permission to connect a custom key store to its associated AWS CloudHSM cluster.	Write			
CreateAlias	Controls permission to create an alias for a customer master key (CMK). Aliases are optional friendly names that you can associate with customer master keys.	Write	alias* (p. 1312)		
			key* (p. 1312)		
				kms:CallerAccount (p. 1313)	kms:ViaService (p. 1314)
CreateCustomKeyStores	Controls permission to create a custom key store that is associated with an AWS CloudHSM cluster that you own and manage.	Write			cloudhsm:DescribeClusters
CreateGrant	Controls permission to add a grant to a customer master key. You can use grants to add permissions without changing the key policy or IAM policy.	Permissions management	key* (p. 1312)		
				kms:CallerAccount (p. 1313)	
				kms:GrantConstraintType (p. 1313)	
				kms:GrantIsForAWSResource (p. 1313)	
				kms:ViaService (p. 1314)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateKey	Controls permission to create a customer master key that can be used to protect data keys and other sensitive information.	Write		kms:BypassPolicyLockoutSafetyCheck (p. 1313) kms:CustomerMasterKeySpec (p. 1313) kms:CustomerMasterKeyUsage (p. 1313) kms:KeyOrigin (p. 1313)	
Decrypt	Controls permission to decrypt ciphertext that was encrypted under a customer master key.	Write	key* (p. 1312)	kms:CallerAccount (p. 1313) kms:EncryptionAlgorithm (p. 1313) kms:EncryptionContextKeys (p. 1313) kms:ViaService (p. 1314)	
DeleteAlias	Controls permission to delete an alias. Aliases are optional friendly names that you can associate with customer master keys.	Write	alias* (p. 1312)		
			key* (p. 1312)	kms:CallerAccount (p. 1313) kms:ViaService (p. 1314)	
DeleteCustomKeyStore	Controls permission to delete a custom key store.	Write			
DeleteImportedKeyMaterial	Controls permission to delete cryptographic material that you imported into a customer master key. This action makes the key unusable.	Write	key* (p. 1312)		
			kms:CallerAccount (p. 1313) kms:ViaService (p. 1314)		
DescribeCustomKeyStores	Controls permission to view detailed information about custom key stores in the account and region.	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeKey	Controls permission to view detailed information about a customer master key.	Read	key* (p. 1312)		
				kms:CallerAccount (p. 1313)	kms:ViaService (p. 1314)
DisableKey	Controls permission to disable a customer master key, which prevents it from being used in cryptographic operations.	Write	key* (p. 1312)		
				kms:CallerAccount (p. 1313)	kms:ViaService (p. 1314)
DisableKeyRotation	Controls permission to disable automatic rotation of a customer managed customer master key.	Write	key* (p. 1312)		
				kms:CallerAccount (p. 1313)	kms:ViaService (p. 1314)
DisconnectCustomStore	Controls permission to disconnect the custom key store from its associated AWS CloudHSM cluster.	Write			
EnableKey	Controls permission to change the state of a customer master key (CMK) to enabled. This allows the CMK to be used in cryptographic operations.	Write	key* (p. 1312)		
				kms:CallerAccount (p. 1313)	kms:ViaService (p. 1314)
EnableKeyRotation	Controls permission to enable automatic rotation of the cryptographic material in a customer master key.	Write	key* (p. 1312)		
				kms:CallerAccount (p. 1313)	kms:ViaService (p. 1314)
Encrypt	Controls permission to use the specified customer master key to encrypt data and data keys.	Write	key* (p. 1312)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				kms:CallerAccount (p. 1313) kms:EncryptionAlgorithm (p. 1313) kms:EncryptionContextKeys (p. 1313) kms:ViaService (p. 1314)	
GenerateDataKey	Controls permission to use the customer master key to generate data keys. You can use the data keys to encrypt data outside of AWS KMS.	Write	key*	(p. 1312)	
				kms:CallerAccount (p. 1313) kms:EncryptionAlgorithm (p. 1313) kms:EncryptionContextKeys (p. 1313) kms:ViaService (p. 1314)	
GenerateDataKeyPair	Controls permission to use the customer master key to generate data key pairs.	Write	key*	(p. 1312)	
				kms:CallerAccount (p. 1313) kms:DataKeyValuePairSpec (p. 1313) kms:EncryptionAlgorithm (p. 1313) kms:EncryptionContextKeys (p. 1313) kms:ViaService (p. 1314)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GenerateDataKey	Controls permission to use the customer master key to generate data key pairs. Unlike the GenerateDataKeyPair operation, this operation returns an encrypted private key without a plaintext copy.	Write	key* (p. 1312)		kms:CallerAccount (p. 1313) kms:DataKeyPairSpec (p. 1313) kms:EncryptionAlgorithm (p. 1313) kms:EncryptionContextKeys (p. 1313) kms:ViaService (p. 1314)
GenerateDataKey	Controls permission to use the customer master key to generate a data key. Unlike the GenerateDataKey operation, this operation returns an encrypted data key without a plaintext version of the data key.	Write	key* (p. 1312)		kms:CallerAccount (p. 1313) kms:EncryptionAlgorithm (p. 1313) kms:EncryptionContextKeys (p. 1313) kms:ViaService (p. 1314)
GenerateRandom	Controls permission to get a cryptographically secure random byte string from AWS KMS.	Write			
GetKeyPolicy	Controls permission to view the key policy for the specified customer master key.	Read	key* (p. 1312)		kms:CallerAccount (p. 1313) kms:ViaService (p. 1314)
GetKeyRotationStatus	Controls permission to determine whether automatic key rotation is enabled on the customer master key.	Read	key* (p. 1312)		kms:CallerAccount (p. 1313) kms:ViaService (p. 1314)

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetParametersForImport	Controls permission to get data that is required to import cryptographic material into a customer managed key, including a public key and import token.	Read	key* (p. 1312)		
				kms:CallerAccount (p. 1313)	kms:ViaService (p. 1314)
GetPublicKey	Controls permission to download the public key of an asymmetric customer master key.	Read	key* (p. 1312)		
				kms:CallerAccount (p. 1313)	kms:ViaService (p. 1314)
ImportKeyMaterial	Controls permission to import cryptographic material into a customer master key.	Write	key* (p. 1312)		
				kms:CallerAccount (p. 1313)	kms:ExpirationModel (p. 1313)
				kms:ValidTo (p. 1314)	kms:ViaService (p. 1314)
ListAliases	Controls permission to view the aliases that are defined in the account. Aliases are optional friendly names that you can associate with customer master keys.	List			
ListGrants	Controls permission to view all grants for a customer master key.	List	key* (p. 1312)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				kms:CallerAccount (p. 1313) kms:GrantIsForAWSResource (p. 1313) kms:ViaService (p. 1314)	
ListKeyPolicies	Controls permission to view the names of key policies for a customer master key.	List	key* (p. 1312)		
				kms:CallerAccount (p. 1313)	kms:ViaService (p. 1314)
ListKeys	Controls permission to view the key ID and Amazon Resource Name (ARN) of all customer master keys in the account.	List			
ListResourceTags	Controls permission to view all tags that are attached to a customer master key.	Read	key* (p. 1312)		
				kms:CallerAccount (p. 1313)	kms:ViaService (p. 1314)
ListRetirableGrants	Controls permission to view grants in which the specified principal is the retiring principal. Other principals might be able to retire the grant and this principal might be able to retire other grants.	List	key* (p. 1312)		
PutKeyPolicy	Controls permission to replace the key policy for the specified customer master key.	Permissions management	key* (p. 1312)		
			kms:BypassPolicyLockoutSafetyCheck (p. 1313)	kms:CallerAccount (p. 1313)	
ReEncryptFrom	Controls permission to decrypt data as part of the process that decrypts and reencrypts the data within AWS KMS.	Write	key* (p. 1312)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				kms:CallerAccount (p. 1313) kms:EncryptionAlgorithm (p. 1313) kms:EncryptionContextKeys (p. 1313) kms:ReEncryptOnSameKey (p. 1314) kms:ViaService (p. 1314)	
ReEncryptTo	Controls permission to encrypt data as part of the process that decrypts and reencrypts the data within AWS KMS.	Write	key* (p. 1312)		
				kms:CallerAccount (p. 1313) kms:EncryptionAlgorithm (p. 1313) kms:EncryptionContextKeys (p. 1313) kms:ReEncryptOnSameKey (p. 1314) kms:ViaService (p. 1314)	
RetireGrant	Controls permission to retire a grant. The RetireGrant operation is typically called by the grant user after they complete the tasks that the grant allowed them to perform.	Permissions management	key* management (p. 1312)		
RevokeGrant	Controls permission to revoke a grant, which denies permission for all operations that depend on the grant.	Permissions management	key* management (p. 1312)		
				kms:CallerAccount (p. 1313) kms:GrantIsForAWSResource (p. 1313) kms:ViaService (p. 1314)	
ScheduleKeyDelete	Controls permission to schedule deletion of a customer master key.	Write	key* (p. 1312)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions	
				kms:CallerAccount (p. 1313) kms:ViaService (p. 1314)		
Sign	Controls permission to produce a digital signature for a message.	Write	key* (p. 1312)			
TagResource	Controls permission to create or update tags that are attached to a customer master key.	Tagging	key* (p. 1312)			
						kms:CallerAccount (p. 1313) kms:ViaService (p. 1314)
UntagResource	Controls permission to delete tags that are attached to a customer master key.	Tagging	key* (p. 1312)			
						kms:CallerAccount (p. 1313) kms:ViaService (p. 1314)
UpdateAlias	Controls permission to associate an alias with a different customer master key. An alias is an optional friendly name that you can associate with a customer master key.	Write	alias* (p. 1312)			
				key* (p. 1312)		
						kms:CallerAccount (p. 1313) kms:ViaService (p. 1314)
UpdateCustomKeyStores	Controls permission to change the properties of a custom key store.	Write				

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UpdateKeyDescription	Controls permission to delete or change the description of a customer master key.	Write	key* (p. 1312)		
				kms:CallerAccount (p. 1313) kms:ViaService (p. 1314)	
Verify	Controls permission to use the specified customer master key to verify digital signatures.	Write	key* (p. 1312)		
				kms:CallerAccount (p. 1313) kms:MessageType (p. 1313) kms:SigningAlgorithm (p. 1314) kms:ViaService (p. 1314)	

Resource Types Defined by AWS Key Management Service

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1302\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
alias	<code>arn:\${Partition}:kms:\${Region}: \${Account}:alias/\${Alias}</code>	
key	<code>arn:\${Partition}:kms:\${Region}: \${Account}:key/\${KeyId}</code>	

Condition Keys for AWS Key Management Service

AWS Key Management Service defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
kms:BypassPolicyLockoutSafetyCheck	Controls access to the CreateKey and PutKeyPolicy operations based on the value of the BypassPolicyLockoutSafetyCheck parameter in the request.	Bool
kms:CallerAccount	Controls access to specified AWS KMS operations based on the AWS account ID of the caller. You can use this condition key to allow or deny access to all IAM users and roles in an AWS account in a single policy statement.	String
kms:CustomerMasterKeySpec	Controls access to an API operation based on the CustomerMasterKeySpec property of the CMK that is created by or used in the operation. Use it to qualify authorization of the CreateKey operation or any operation that is authorized for a CMK resource.	String
kms:CustomerMasterKeyUsageProperty	Controls access to an API operation based on the KeyUsage property of the CMK created by or used in the operation. Use it to qualify authorization of the CreateKey operation or any operation that is authorized for a CMK resource.	String
kms:DataKeyPairSpec	Controls access to GenerateDataKeyPair and GenerateDataKeyPairWithoutPlaintext operations based on the value of the DataKeyPairSpec parameter in the request.	String
kms:EncryptionAlgorithm	Controls access to encryption operations based on the value of the encryption algorithm in the request.	String
kms:EncryptionContext	Controls access based on the presence of specified keys in the encryption context. The encryption context is an optional element in a cryptographic operation.	String
kms:ExpirationModel	Controls access to the ImportKeyMaterial operation based on the value of the ExpirationModel parameter in the request.	String
kms:GrantConstraintType	Controls access to the CreateGrant operation based on the constraint in the request.	String
kms:GrantIsForAWSRequest	Controls access to the CreateGrant operation when the request comes from a specified AWS service.	Bool
kms:GrantOperations	Controls access to the CreateGrant operation based on the operations in the grant.	String
kms:GranteePrincipal	Controls access to the CreateGrant operation based on the grantee principal in the grant.	String
kms:KeyOrigin	Controls access to an API operation based on the Origin property of the CMK created by or used in the operation. Use it to qualify authorization of the CreateKey operation or any operation that is authorized for a CMK resource.	String
kms:MessageType	Controls access to the Sign and Verify operations based on the value of the MessageType parameter in the request.	String

Condition Keys	Description	Type
kms:ReEncryptOnSnsTopic	Controls access to the ReEncrypt operation when it uses the customer master key that was used for the Encrypt operation.	Bool
kms:RetiringPrincipal	Controls access to the CreateGrant operation based on the retiring principal in the grant.	String
kms:SigningAlgorithm	Controls access to the Sign and Verify operations based on the signing algorithm in the request.	String
kms:ValidTo	Controls access to the ImportKeyMaterial operation based on the value of the ValidTo parameter in the request. You can use this condition key to allow users to import key material only when it expires by the specified date.	Numeric
kms:ViaService	Controls access when a request made on the principal's behalf comes from a specified AWS service.	String
kms:WrappingAlgorithm	Controls access to the GetParametersForImport operation based on the value of the WrappingAlgorithm parameter in the request.	String
kms:WrappingKeySpec	Controls access to the GetParametersForImport operation based on the value of the WrappingKeySpec parameter in the request.	String

Actions, Resources, and Condition Keys for Amazon Kinesis

Amazon Kinesis (service prefix: `kinesis`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Kinesis \(p. 1314\)](#)
- [Resource Types Defined by Amazon Kinesis \(p. 1317\)](#)
- [Condition Keys for Amazon Kinesis \(p. 1317\)](#)

Actions Defined by Amazon Kinesis

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in

a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AddTagsToStream	Adds or updates tags for the specified Amazon Kinesis stream. Each stream can have up to 10 tags.	Tagging	stream* (p. 1317)		
CreateStream	Creates a Amazon Kinesis stream.	Write	stream* (p. 1317)		
DecreaseStreamRetentionPeriod	Decreases the stream's retention period, which is the length of time data records are accessible after they are added to the stream.	Write	stream* (p. 1317)		
DeleteStream	Deletes a stream and all its shards and data.	Write	stream* (p. 1317)		
DeregisterStreamWithKinesis	Deregisters a stream consumer with a Kinesis data stream.	Write	consumer* (p. 1317) stream* (p. 1317)		
DescribeLimits	Describes the shard limits and usage for the account.	Read			
DescribeStream	Describes the specified stream.	Read	stream* (p. 1317)		
DescribeStreamConsumer	Gets the description of a registered stream consumer.	Read	consumer* (p. 1317) stream* (p. 1317)		
DescribeStreamSummary	Provides a summarized description of the specified Kinesis data stream without the shard list.	Read	stream* (p. 1317)		
DisableEnhancedMonitoring	Disables enhanced monitoring.	Write			
EnableEnhancedMonitoring	API_EnableEnhancedMonitoring.htm	Write			
GetRecords	Gets data records from a shard.	Read	stream* (p. 1317)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetShardIterator	Gets a shard iterator. A shard iterator expires five minutes after it is returned to the requester.	Read	stream* (p. 1317)		
IncreaseStreamRetentionPeriod , which is the length of time data records are accessible after they are added to the stream.		Write	stream* (p. 1317)		
ListShards	Lists the shards in a stream and provides information about each shard.	List			
ListStreamConsumers	Lists the stream consumers registered to receive data from a Kinesis stream using enhanced fan-out, and provides information about each consumer.	List			
ListStreams	Lists your streams.	List			
ListTagsForStream	Lists the tags for the specified Amazon Kinesis stream.	Read	stream* (p. 1317)		
MergeShards	Merges two adjacent shards in a stream and combines them into a single shard to reduce the stream's capacity to ingest and transport data.	Write	stream* (p. 1317)		
PutRecord	Writes a single data record from a producer into an Amazon Kinesis stream.	Write	stream* (p. 1317)		
PutRecords	Writes multiple data records from a producer into an Amazon Kinesis stream in a single call (also referred to as a PutRecords request).	Write	stream* (p. 1317)		
RegisterStreamConsumer with the Kinesis data stream.		Write	consumer* (p. 1317)		
			stream* (p. 1317)		
RemoveTagsFromStream	Description for SplitShard	Tagging	stream* (p. 1317)		
SplitShard	Description for SplitShard	Write	stream* (p. 1317)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
SubscribeToShard	Listening to a specific shard with enhanced fan-out.	Read	consumer* (p. 1317)		
			stream* (p. 1317)		
UpdateShardCount	Updates the shard count of the specified stream to the specified number of shards.	Write			

Resource Types Defined by Amazon Kinesis

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 1314\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
stream	arn:\${Partition}:kinesis:\${Region}:\${Account}:stream/\${StreamName}	
consumer	arn:\${Partition}:kinesis:\${Region}:\${Account}:#\${StreamType}/ \${StreamName}/consumer/\${ConsumerName}: \${ConsumerCreationTimestamp}	

Condition Keys for Amazon Kinesis

Kinesis has no service-specific context keys that can be used in the Condition element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Amazon Kinesis Analytics

Amazon Kinesis Analytics (service prefix: `kinesisanalytics`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Kinesis Analytics \(p. 1318\)](#)

- [Resource Types Defined by Amazon Kinesis Analytics \(p. 1319\)](#)
- [Condition Keys for Amazon Kinesis Analytics \(p. 1319\)](#)

Actions Defined by Amazon Kinesis Analytics

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AddApplicationInput	Adds input to the application.	Write	application* (p. 1319)		
AddApplicationOutput	Adds output to the application.	Write	application* (p. 1319)		
AddApplicationReference	Adds reference data source to the application.	Write	application* (p. 1319)		
CreateApplication	Creates an application.	Write		aws:RequestTag/\${TagKey} (p. 1320) aws:TagKeys (p. 1320)	
DeleteApplication	Deletes the application.	Write	application* (p. 1319)		
DeleteApplicationOutput	Deletes the specified output of the application.	Write	application* (p. 1319)		
DeleteApplicationReference	Deletes the specified reference data source of the application.	Write	application* (p. 1319)		
DescribeApplication	Describes the specified application.	Read	application* (p. 1319)		
DiscoverInputSchema	Discovers the input schema for the application.	Read			
GetApplicationSettings	Grant permission to Kinesis Data Analytics console to	Read	application* (p. 1319)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
[permission only]	display stream results for Kinesis Data Analytics SQL runtime applications.				
ListApplications	List applications for the account	List			
ListTagsForResource	Fetch the tags associated with the application.	Read	application* (p. 1319)		
StartApplication	Starts the application.	Write	application* (p. 1319)		
StopApplication	Stops the application.	Write	application* (p. 1319)		
TagResource	Add tags to the application.	Tagging	application* (p. 1319)		
			aws:RequestTag/ {\$TagKey} (p. 1320) aws:TagKeys (p. 1320)		
UntagResource	Remove the specified tags from the application.	Tagging	application* (p. 1319)		
			aws:TagKeys (p. 1320)		
UpdateApplication	Updates the application.	Write	application* (p. 1319)		

Resource Types Defined by Amazon Kinesis Analytics

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1318\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
application	<code>arn:\${Partition}:kinesisanalytics:\${Region}: \${Account}:application/\${ApplicationName}</code>	aws:ResourceTag/ \${TagKey} (p. 1320)

Condition Keys for Amazon Kinesis Analytics

Amazon Kinesis Analytics defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the

policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Filters actions based on the allowed set of values for each of the tags	String
aws:ResourceTag/\${TagKey}	Filters actions based on tag-value associated with the resource	String
aws:TagKeys	Filters actions based on the presence of mandatory tag keys in the request	String

Actions, Resources, and Condition Keys for Amazon Kinesis Analytics V2

Amazon Kinesis Analytics V2 (service prefix: `kinesisanalytics`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Kinesis Analytics V2 \(p. 1320\)](#)
- [Resource Types Defined by Amazon Kinesis Analytics V2 \(p. 1322\)](#)
- [Condition Keys for Amazon Kinesis Analytics V2 \(p. 1323\)](#)

Actions Defined by Amazon Kinesis Analytics V2

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AddApplicationCloudWatchLoggingOption	Adds cloudwatch logging option to the application.	Write	application* (p. 1322)		
AddApplicationInput	Adds input to the application.	Write	application* (p. 1322)		
AddApplicationInputConfiguration	Adds input processing configuration to the application.	Write	application* (p. 1322)		
AddApplicationOutput	Adds output to the application.	Write	application* (p. 1322)		
AddApplicationReference	Adds reference data source to the application.	Write	application* (p. 1322)		
AddApplicationVpcConfiguration	Adds VPC configuration to the application.	Write	application* (p. 1322)		
CreateApplication	Creates an application.	Write		aws:RequestTag/\${TagKey} (p. 1323) aws:TagKeys (p. 1323)	
CreateApplicationSnapshot	Creates a snapshot for an application.	Write	application* (p. 1322)		
DeleteApplication	Deletes the application.	Write	application* (p. 1322)		
DeleteApplicationCloudWatchLoggingOption	Deletes the specified cloudwatch logging option of the application.	Write	application* (p. 1322)		
DeleteApplicationInputProcessingConfiguration	Deletes the specified input processing configuration of the application.	Write	application* (p. 1322)		
DeleteApplicationOutput	Deletes the specified output of the application.	Write	application* (p. 1322)		
DeleteApplicationReference	Deletes the specified reference data source of the application.	Write	application* (p. 1322)		
DeleteApplicationSnapshot	Deletes a snapshot for an application.	Write	application* (p. 1322)		
DeleteApplicationVpcConfiguration	Deletes the specified VPC configuration of the application.	Write	application* (p. 1322)		
DescribeApplication	Describes the specified application.	Read	application* (p. 1322)		
DescribeApplicationSnapshot	Describes an application snapshot.	Read	application* (p. 1322)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DiscoverInputSchema	Discovers the input schema for the application.	Read			
ListApplicationSnapshots	Lists the snapshots for an application.	Read	application* (p. 1322)		
ListApplications	List applications for the account	List			
ListTagsForResource	Fetch the tags associated with the application.	Read	application* (p. 1322)		
StartApplication	Starts the application.	Write	application* (p. 1322)		
StopApplication	Stops the application.	Write	application* (p. 1322)		
TagResource	Add tags to the application.	Tagging	application* (p. 1322)		
				aws:RequestTag/\${TagKey} (p. 1323)	
				aws:TagKeys (p. 1323)	
UntagResource	Remove the specified tags from the application.	Tagging	application* (p. 1322)		
				aws:TagKeys (p. 1323)	
UpdateApplication	Updates the application.	Write	application* (p. 1322)		

Resource Types Defined by Amazon Kinesis Analytics V2

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1320\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
application	<code>arn:\${Partition}:kinesisanalytics:\${Region}:\${Account}:application/\${ApplicationName}</code>	aws:ResourceTag/\${TagKey} (p. 1323)

Condition Keys for Amazon Kinesis Analytics V2

Amazon Kinesis Analytics V2 defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
<code>aws:RequestTag/\${TagKey}</code>	Filters actions based on the allowed set of values for each of the tags	String
<code>aws:ResourceTag/\${TagKey}</code>	Filters actions based on tag-value associated with the resource	String
<code>aws:TagKeys</code>	Filters actions based on the presence of mandatory tag keys in the request	String

Actions, Resources, and Condition Keys for Amazon Kinesis Firehose

Amazon Kinesis Firehose (service prefix: `firehose`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Kinesis Firehose \(p. 1323\)](#)
- [Resource Types Defined by Amazon Kinesis Firehose \(p. 1325\)](#)
- [Condition Keys for Amazon Kinesis Firehose \(p. 1325\)](#)

Actions Defined by Amazon Kinesis Firehose

You can specify the following actions in the Action element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the Resource element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type.

Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateDeliveryStream	Creates a delivery stream.	Write	deliverystream*		
			(p. 1325)	aws:RequestTag/ \${TagKey} (p. 1325)	
				aws:TagKeys (p. 1325)	
DeleteDeliveryStream	Deletes a delivery stream and its data.	Write	deliverystream*		
	(p. 1325)		(p. 1325)		
DescribeDeliveryStream	Describes the specified delivery stream and gets the status.	List	deliverystream*		
	(p. 1325)		(p. 1325)		
ListDeliveryStreams	Lists your delivery streams.	List			
ListTagsForDeliveryStream	Lists the tags for the specified delivery stream.	List	deliverystream*		
	(p. 1325)		(p. 1325)		
PutRecord	Writes a single data record into an Amazon Kinesis Firehose delivery stream.	Write	deliverystream*		
	(p. 1325)		(p. 1325)		
PutRecordBatch	Writes multiple data records into a delivery stream in a single call, which can achieve higher throughput per producer than when writing single records.	Write	deliverystream*		
	(p. 1325)		(p. 1325)		
StartDeliveryStream	Enables server-side encryption (SSE) for the delivery stream.	Write	deliverystream*		
	(p. 1325)		(p. 1325)		
StopDeliveryStream	Disables the specified destination of the specified delivery stream.	Write	deliverystream*		
	(p. 1325)		(p. 1325)		
TagDeliveryStream	Adds or updates tags for the specified delivery stream.	Write	deliverystream*		
	(p. 1325)			aws:RequestTag/ \${TagKey} (p. 1325)	
				aws:TagKeys (p. 1325)	
UntagDeliveryStream	Removes tags from the specified delivery stream.	Write	deliverystream*		
	(p. 1325)		(p. 1325)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:TagKeys (p. 1325)	
UpdateDestination	Updates the specified destination of the specified delivery stream.	Write	deliverystream* (p. 1325)		

Resource Types Defined by Amazon Kinesis Firehose

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1323\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
deliverystream	<code>arn:\${Partition}:firehose:\${Region}:\${Account}:deliverystream/\${DeliveryStreamName}</code>	aws:ResourceTag/\${TagKey} (p. 1325)

Condition Keys for Amazon Kinesis Firehose

Amazon Kinesis Firehose defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
<code>aws:RequestTag/\${TagKey}</code>	Filters create requests based on the allowed set of values for each of the tags	String
<code>aws:ResourceTag/\${TagKey}</code>	Filters actions based on tag-value associated with the resource.	String
<code>aws:TagKeys</code>	Filters create requests based on the presence of mandatory tags in the request	String

Actions, Resources, and Condition Keys for Amazon Kinesis Video Streams

Amazon Kinesis Video Streams (service prefix: `kinesisvideo`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Kinesis Video Streams \(p. 1326\)](#)
- [Resource Types Defined by Amazon Kinesis Video Streams \(p. 1329\)](#)
- [Condition Keys for Amazon Kinesis Video Streams \(p. 1329\)](#)

Actions Defined by Amazon Kinesis Video Streams

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ConnectAsMaster	Grants permission to connect as a master to the signaling channel specified by the endpoint	Write	channel* (p. 1329)		
ConnectAsViewer	Grants permission to connect as a viewer to the signaling channel specified by the endpoint	Write	channel* (p. 1329)		
CreateSignalingChannel	Grants permission to create a signaling channel	Write	channel* (p. 1329)		
				aws:RequestTag/\${TagKey} (p. 1329) aws:TagKeys (p. 1329)	
CreateStream	Grants permission to create a Kinesis video stream	Write	stream* (p. 1329)		
				aws:RequestTag/\${TagKey} (p. 1329)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:TagKeys (p. 1329)	
DeleteSignalingChannel	Grants permission to delete an existing signaling channel	Write	channel* (p. 1329)		
DeleteStream	Grants permission to delete an existing Kinesis video stream	Write	stream* (p. 1329)		
DescribeSignalingChannel	Grants permission to describe the specified signaling channel	List	channel* (p. 1329)		
DescribeStream	Grants permission to describe the specified Kinesis video stream	List	stream* (p. 1329)		
GetDASHStreamingURLsForMPREG-DASH	Grants permission to create URLs for MPREG-DASH video streaming	Read	stream* (p. 1329)		
GetDataEndpoint	Grants permission to get an endpoint for a specified stream for either reading or writing media data to Kinesis Video Streams	Read	stream* (p. 1329)		
GetHLSStreamingURLsForHLS	Grants permission to create URLs for HLS video streaming	Read	stream* (p. 1329)		
GetIceServerConfig	Grants permission to get the ICE server configuration	Read	channel* (p. 1329)		
GetMedia	Grants permission to return media content of a Kinesis video stream	Read	stream* (p. 1329)		
GetMediaForFragmentationPoint	Grants permission to read and return media data only from persisted storage	Read	stream* (p. 1329)		
GetSignalingChannelEndPoints	Grants permission to get endpoints for a specified combination of protocol and role for a signaling channel	Read	channel* (p. 1329)		
ListFragments	Grants permission to list the fragments from archival storage based on the pagination token or selector type with range specified	List	stream* (p. 1329)		
ListSignalingChannels	Grants permission to list your signaling channels	List			
ListStreams	Grants permission to list your Kinesis video streams	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListTagsForResource	Grants permission to fetch the tags associated with your resource	Read	channel (p. 1329)		
	stream (p. 1329)				
ListTagsForStream	Grants permission to fetch the tags associated with Kinesis video stream	Read	stream* (p. 1329)		
PutMedia	Grants permission to send media data to a Kinesis video stream	Write	stream* (p. 1329)		
SendAlexaOfferToAlexa	Grants permission to send the Alexa SDP offer to the master	Write	channel* (p. 1329)		
TagResource	Grants permission to attach set of tags to your resource	Tagging	channel (p. 1329)		
	stream (p. 1329)				
			aws:RequestTag/\${TagKey} (p. 1329) aws:TagKeys (p. 1329)		
TagStream	Grants permission to attach set of tags to your Kinesis video streams	Tagging	stream* (p. 1329)		
			aws:RequestTag/\${TagKey} (p. 1329) aws:TagKeys (p. 1329)		
UntagResource	Grants permission to remove one or more tags from your resource	Tagging	channel (p. 1329)		
	stream (p. 1329)				
			aws:TagKeys (p. 1329)		
UntagStream	Grants permission to remove one or more tags from your Kinesis video streams	Tagging	stream* (p. 1329)		
			aws:TagKeys (p. 1329)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UpdateDataRetention	Grants permission to update the data retention period of your Kinesis video stream	Write	stream* (p. 1329)		
UpdateSignalingChannel	Grants permission to update an existing signaling channel	Write	channel* (p. 1329)		
UpdateStream	Grants permission to update an existing Kinesis video stream	Write	stream* (p. 1329)		

Resource Types Defined by Amazon Kinesis Video Streams

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 1326\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
stream	arn:\${Partition}:kinesisvideo:\${Region}: \${Account}:stream/\${StreamName}/ \${CreationTime}	aws:ResourceTag/\${TagKey} (p. 1329)
channel	arn:\${Partition}:kinesisvideo:\${Region}: \${Account}:channel/\${ChannelName}/ \${CreationTime}	aws:ResourceTag/\${TagKey} (p. 1329)

Condition Keys for Amazon Kinesis Video Streams

Amazon Kinesis Video Streams defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Filters requests based on the allowed set of values for each of the tags	String
aws:ResourceTag/\${TagKey}	Filters actions based on tag-value associated with the stream.	String
aws:TagKeys	Filters requests based on the presence of mandatory tag keys in the request	String

Actions, Resources, and Condition Keys for AWS Lake Formation

AWS Lake Formation (service prefix: `lakeformation`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Lake Formation \(p. 1330\)](#)
- [Resource Types Defined by AWS Lake Formation \(p. 1331\)](#)
- [Condition Keys for AWS Lake Formation \(p. 1331\)](#)

Actions Defined by AWS Lake Formation

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
BatchGrantPermissions	Grants data lake permissions to one or more principals in a batch.	Permissions management			
BatchRevokePermissions	Revokes data lake permissions from one or more principals in a batch.	Permissions management			
DeregisterResource	Deregisters a registered location.	Write			
DescribeResource	Describes a registered location.	Read			
GetDataAccess	Grants virtual data lake access permissions.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetDataLakeSettings	Retrieves data lake settings such as the list of data lake administrators and database and table default permissions.	Read			
GetEffectivePermissions	Retrieves permissions attached to resources in the given path.	Read			
GrantPermissions	Grants data lake permissions to a principal.	Permissions management			
ListPermissions	Lists permissions filtered by principal or resource.	List			
ListResources	Lists registered locations.	List			
PutDataLakeSettings	Overwrites data lake settings such as the list of data lake administrators and database and table default permissions.	Permissions management			
RegisterResource	Registers a new location to be managed by Lake Formation.	Write			
RevokePermissions	Revokes data lake permissions from a principal.	Permissions management			
UpdateResource	Updates a registered location.	Write			

Resource Types Defined by AWS Lake Formation

AWS Lake Formation does not support specifying a resource ARN in the `Resource` element of an IAM policy statement. To allow access to AWS Lake Formation, specify “`Resource`”: “`*`” in your policy.

Condition Keys for AWS Lake Formation

Lake Formation has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for AWS Lambda

AWS Lambda (service prefix: `lambda`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Lambda \(p. 1332\)](#)
- [Resource Types Defined by AWS Lambda \(p. 1335\)](#)
- [Condition Keys for AWS Lambda \(p. 1336\)](#)

Actions Defined by AWS Lambda

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AddLayerVersion	Adds a permission policy to a <code>version</code> of a function layer.	Permissions management	<code>layerVersion*</code> (p. 1335)		
AddPermission	Adds a permission to the resource policy associated with the specified AWS Lambda function.	Permissions management	<code>function*</code> (p. 1335)		<code>lambda:Principal</code> (p. 1336)
CreateAlias	Creates an alias that points to the specified Lambda function version.	Write	<code>function*</code> (p. 1335)		
CreateEventSourceMapping	Identifies a stream as an event source for a Lambda function.	Write		<code>lambda:FunctionArn</code> (p. 1336)	
CreateFunction	Creates a new Lambda function.	Write	<code>function*</code> (p. 1335)		<code>lambda:Layer</code> (p. 1336)
DeleteAlias	Deletes the specified Lambda function alias.	Write	<code>function*</code> (p. 1335)		
DeleteEventSourceMapping	Removes an event source mapping.	Write	<code>eventSourceMapping*</code> (p. 1336)		<code>lambda:FunctionArn</code> (p. 1336)

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteFunction	Deletes the specified Lambda function code and configuration.	Write	function* (p. 1335)		
DeleteFunctionConcurrency	Remove concurrency limit set on Lambda function.	Write	function* (p. 1335)		
DeleteLayerVersion	Deletes a version of a function layer.	Write	layerVersion* (p. 1335)		
DisableReplicationPermission [permission only]	Removes resource policy permission that allows Lambda replication service to retrieve function code and configuration.	Permissions management	function* (p. 1335)		
EnableReplicationPermission [permission only]	Adds a permission to resource policy that gives Lambda replication service permission to get function code and configuration.	Permissions management	function* (p. 1335)		
GetAccountSettings	Returns account limits and usage statistics, such as concurrency and code storage.	Read			
GetAlias	Returns the specified alias information such as the alias ARN, description, and function version it is pointing to.	Read	function* (p. 1335)		
GetEventSourceMapping	Returns configuration information for the specified event source mapping.	Read	eventSourceMapping* (p. 1336)		
				lambda:FunctionArn (p. 1336)	
GetFunction	Returns the configuration information of the Lambda function and a presigned URL link to the .zip file you uploaded with CreateFunction so you can download the .zip file.	Read	function* (p. 1335)		
GetFunctionConfiguration	Returns the configuration information of the Lambda function.	Read	function* (p. 1335)		
GetLayerVersion	Returns information about a version of a function layer, with a link to download the layer archive that is valid for 10 minutes.	Read	layerVersion* (p. 1335)		
GetLayerVersionPolicy	Returns the permissions policy for a layer version.	Read	layerVersion* (p. 1335)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetPolicy	Returns the resource policy associated with the specified Lambda function.	Read	function* (p. 1335)		
InvokeAsync	Submits an invocation request to AWS Lambda. Is deprecated	Write	function* (p. 1335)		
InvokeFunction [permission only]	Invokes a specific Lambda function.	Write	function* (p. 1335)		
ListAliases	Returns list of aliases created for a Lambda function.	List	function* (p. 1335)		
ListEventSourceMappings	Returns a list of event source mappings you created using the <code>CreateEventSourceMapping</code> .	List			
ListFunctions	Returns a list of your Lambda functions.	List			
ListLayerVersions	Returns a list of your Lambda layer versions.	List			
ListLayers	Lists function layers and shows information about the latest version of each.	List			
ListTags	Lists tags for a Lambda function.	Read	function* (p. 1335)		
ListVersionsByFunction	List all versions of a function.	List	function* (p. 1335)		
PublishLayerVersion	Creates a function layer from a ZIP archive. Each time you call <code>PublishLayerVersion</code> with the same version name, a new version is created.	Write	layer* (p. 1335)		
PublishVersion	Publishes a version of your function from the current snapshot of \$LATEST.	Write	function* (p. 1335)		
PutFunctionConcurrency	Adds concurrency limit to a Lambda function.	Write	function* (p. 1335)		
RemoveLayerVersionPermissions	Removes a statement from the permissions policy for a layer version.	Permissions management	layerVersion* (p. 1335)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
RemovePermission	You can remove individual permissions from an resource policy associated with a Lambda function by providing a statement ID that you provided when you added the permission.	Permissions management	function* (p. 1335)		
				lambda:Principal (p. 1336)	
TagResource	Adds tags to a Lambda function.	Write	function* (p. 1335)		
UntagResource	Removes tags from a Lambda function.	Write	function* (p. 1335)		
UpdateAlias	Using this API you can update the function version to which the alias points and the alias description.	Write	function* (p. 1335)		
UpdateEventSourceMapping	You can update an event source mapping	Write	eventSourceMapping* (p. 1336)		
				lambda:FunctionArn (p. 1336)	
UpdateFunctionCode	Updates the code for the specified Lambda function.	Write	function* (p. 1335)		
UpdateFunctionConfiguration	Updates the configuration parameters for the specified Lambda function by using the values provided in the request.	Write	function* (p. 1335)		
				lambda:Layer (p. 1336)	

Resource Types Defined by AWS Lambda

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 1332\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
function	arn:\${Partition}:lambda:\${Region}: \${Account}:function:\${FunctionName}	
layer	arn:\${Partition}:lambda:\${Region}: \${Account}:layer:\${LayerName}	
layerVersion	arn:\${Partition}:lambda:\${Region}: \${Account}:layer:\${LayerName}: \${LayerVersion}	

Resource Types	ARN	Condition Keys
eventSourceMappings	arn:\${Partition}:lambda:\${Region}:\${Account}:event-source-mapping:\${UUID}	

Condition Keys for AWS Lambda

AWS Lambda defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
lambda:FunctionArn	The ARN of a lambda function.	ARN
lambda:Layer	The ARN of a lambda layer.	String
lambda:Principal	The AWS principal.	String

Actions, Resources, and Condition Keys for Launch Wizard

Launch Wizard (service prefix: `launchwizard`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).

Topics

- [Actions Defined by Launch Wizard \(p. 1336\)](#)
- [Resource Types Defined by Launch Wizard \(p. 1337\)](#)
- [Condition Keys for Launch Wizard \(p. 1337\)](#)

Actions Defined by Launch Wizard

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type.

Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteApp [permission only]	Delete an application	Write			
DescribeProvisionedApplications [permission only]	Describe provisioning applications	Read			
DescribeProvisioningEvents [permission only]	Describe provisioning events	Read			
GetInfrastructureSuggestion [permission only]	Get infrastructure suggestion	Read			
GetIpAddress [permission only]	Get customer's ip address	Read			
GetResourceCostEstimate [permission only]	Get resource cost estimate	Read			
ListProvisionedApps [permission only]	List provisioning applications	List			
StartProvisioning [permission only]	Start a provisioning	Write			

Resource Types Defined by Launch Wizard

Launch Wizard does not support specifying a resource ARN in the `Resource` element of an IAM policy statement. To allow access to Launch Wizard, specify “`Resource`”: “`*`” in your policy.

Condition Keys for Launch Wizard

Launch Wizard has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Amazon Lex

Amazon Lex (service prefix: `lex`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Lex \(p. 1338\)](#)
- [Resource Types Defined by Amazon Lex \(p. 1341\)](#)
- [Condition Keys for Amazon Lex \(p. 1341\)](#)

Actions Defined by Amazon Lex

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateBotVersion	Creates a new version based on the \$LATEST version of the specified bot.	Write	bot* (p. 1341)		
CreateIntentVersion	Creates a new version based on the \$LATEST version of the specified intent.	Write	intent* (p. 1341)		
CreateSlotTypeVersion	Creates a new version based on the \$LATEST version of the specified slot type.	Write	slottype* (p. 1341)		
DeleteBot	Deletes all versions of a bot.	Write	bot* (p. 1341)		
DeleteBotAlias	Deletes an alias for a specific bot.	Write	bot* (p. 1341)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteBotChannelAssociation	Deletes the association between a Amazon Lex bot alias and a messaging platform.	Write	channel* (p. 1341)		
DeleteBotVersion	Deletes a specific version of a bot.	Write	bot* (p. 1341)		
DeleteIntent	Deletes all versions of an intent.	Write	intent* (p. 1341)		
DeleteIntentVersion	Deletes a specific version of an intent.	Write	intent* (p. 1341)		
DeleteSlotType	Deletes all versions of a slot type.	Write	slottype* (p. 1341)		
DeleteSlotTypeVersion	Deletes a specific version of a slot type.	Write	slottype* (p. 1341)		
DeleteUtterances	Deletes the information Amazon Lex maintains for utterances on a specific bot and userId.	Write	bot* (p. 1341)		
GetBot	Returns information for a specific bot. In addition to the bot name, the bot version or alias is required.	Read	bot* (p. 1341)		
GetBotAlias	Returns information about a Amazon Lex bot alias.	Read	bot* (p. 1341)		
GetBotAliases	Returns a list of aliases for a given Amazon Lex bot.	List	bot* (p. 1341)		
GetBotChannelAssociation	Returns information about the association between a Amazon Lex bot and a messaging platform.	Read	channel* (p. 1341)		
GetBotChannelAssociations	Returns a list of all of the channels associated with a single bot.	List	channel* (p. 1341)		
GetBotVersions	Returns information for all versions of a specific bot.	List	bot* (p. 1341)		
GetBots	Returns information for the \$LATEST version of all bots, subject to filters provided by the client.	List			
GetBuiltInIntent	Returns information about a built-in intent.	Read			
GetBuiltInIntents	Gets a list of built-in intents that meet the specified criteria.	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetBuiltinSlotTypes	Gets a list of built-in slot types that meet the specified criteria.	Read			
GetIntent	Returns information for a specific intent. In addition to the intent name, you must also specify the intent version.	Read	intent* (p. 1341)		
GetIntentVersion	Returns information for all versions of a specific intent.	List	intent* (p. 1341)		
GetIntents	Returns information for the \$LATEST version of all intents, subject to filters provided by the client.	List			
GetSlotType	Returns information about a specific version of a slot type. In addition to specifying the slot type name, you must also specify the slot type version.	Read	slottype* (p. 1341)		
GetSlotTypeVersion	Returns information for all versions of a specific slot type.	List	slottype* (p. 1341)		
GetSlotTypes	Returns information for the \$LATEST version of all slot types, subject to filters provided by the client.	List			
GetUtterancesView	Returns a view of aggregate utterance data for versions of a bot for a recent time period.	List	bot* (p. 1341)		
PostContent	Sends user input (text or speech) to Amazon Lex.	Write	bot* (p. 1341)		
PostText	Sends user input (text-only) to Amazon Lex.	Write	bot* (p. 1341)		
PutBot	Creates or updates the \$LATEST version of a Amazon Lex conversational bot.	Write	bot* (p. 1341)		
PutBotAlias	Creates or updates an alias for the specific bot.	Write	bot* (p. 1341)		
PutIntent	Creates or updates the \$LATEST version of an intent.	Write	intent* (p. 1341)		
PutSlotType	Creates or updates the \$LATEST version of a slot type.	Write	slottype* (p. 1341)		

Resource Types Defined by Amazon Lex

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 1338\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
bot	arn:\${Partition}:lex:\${Region}: \${Account}:bot:\${BotName}: \${BotVersionOrAlias}	
channel	arn:\${Partition}:lex:\${Region}: \${Account}:bot-channel:\${BotName}: \${BotAlias}:\${ChannelName}	
intent	arn:\${Partition}:lex:\${Region}: \${Account}:intent:\${IntentName}: \${IntentVersion}	
slottype	arn:\${Partition}:lex:\${Region}: \${Account}:slottype:\${SlotName}: \${SlotVersion}	

Condition Keys for Amazon Lex

Amazon Lex defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
lex:associatedIntents	Enables you to control access based on the intents included in the request.	String
lex:associatedSlotTypes	Enables you to control access based on the slot types included in the request.	String
lex:channelType	Enables you to control access based on the channel type included in the request.	String

Actions, Resources, and Condition Keys for AWS License Manager

AWS License Manager (service prefix: `license-manager`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS License Manager \(p. 1342\)](#)
- [Resource Types Defined by AWS License Manager \(p. 1343\)](#)
- [Condition Keys for AWS License Manager \(p. 1344\)](#)

Actions Defined by AWS License Manager

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateLicenseConfiguration	Creates a new license configuration	Tagging		aws:RequestTag/\${TagKey} (p. 1344) aws:TagKeys (p. 1344)	
DeleteLicenseConfiguration	Permanently deletes a license configuration	Write	license-configuration* (p. 1344)		
GetLicenseConfiguration	Gets a license configuration	List	license-configuration* (p. 1344)		
.GetServiceSettings	Gets service settings	List			
ListAssociationsForLicenseConfiguration	Lists associations for a selected license configuration	List	license-configuration* (p. 1344)		
ListLicenseConfigurations	Lists license configurations	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListLicenseSpecificationsAssociatedWithResource	Lists license specifications associated with a selected resource	List			
ListResourceInventory	Lists resource inventory	List			
ListTagsForResource	Lists tags for a selected resource	List	license-configuration* (p. 1344)		
ListUsageForLicenseConfiguration	Lists usage records for selected license configuration	List	license-configuration* (p. 1344)		
TagResource	Tags a selected resource	Tagging	license-configuration* (p. 1344)		
				aws:RequestTag/ {\$TagKey} (p. 1344)	
				aws:TagKeys (p. 1344)	
UntagResource	Untags a selected resource	Tagging	license-configuration* (p. 1344)		
UpdateLicenseConfiguration	Updates an existing license configuration	Write	license-configuration* (p. 1344)		
UpdateLicenseSpecificationForResource	Updates license specifications for a selected resource	Write	license-configuration* (p. 1344)		
UpdateServiceSettings	Updates service settings	Permissions management			

Resource Types Defined by AWS License Manager

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 1342\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
license-configuration	arn:\${Partition}:license-manager:\${Region}:\${Account}:license-configuration/\${LicenseConfigurationId}	license-manager:ResourceTag/\${TagKey} (p. 1344)

Condition Keys for AWS License Manager

AWS License Manager defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Filters create requests based on allowed set of values for each of the mandatory tags	String
aws:TagKeys	Enforce tag keys that are used in the request	String
license-manager:ResourceTag/source.\${TagKey}	Filters actions based on tag-value associated with the source.	String

Actions, Resources, and Condition Keys for Amazon Lightsail

Amazon Lightsail (service prefix: `lightsail`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Lightsail \(p. 1344\)](#)
- [Resource Types Defined by Amazon Lightsail \(p. 1355\)](#)
- [Condition Keys for Amazon Lightsail \(p. 1356\)](#)

Actions Defined by Amazon Lightsail

You can specify the following actions in the Action element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AllocateStaticIp	Creates a static IP address that can be attached to an instance.	Write	StaticIp* (p. 1355)		
AttachDisk	Attaches a disk to an instance.	Write	Disk* (p. 1355)		
			Instance* (p. 1355)		
AttachInstancesToLoadBalancer	Attaches one or more instances to a load balancer.	Write	Instance* (p. 1355)		
			LoadBalancer* (p. 1356)		
AttachLoadBalancerTLSCertificate	Attaches a TLS certificate to a load balancer.	Write	LoadBalancer* (p. 1356)		
AttachStaticIp	Attaches a static IP address to an instance.	Write	Instance* (p. 1355)		
			StaticIp* (p. 1355)		
CloseInstancePublicPorts	Closes a public port of an instance.	Write	Instance* (p. 1355)		
CopySnapshot	Copies a snapshot from one AWS Region to another in Amazon Lightsail.	Write			
CreateCloudFormationStacks	Creates a new Amazon EC2 instance stack from an exported Amazon Lightsail snapshot.	List	ExportSnapshotRecord* (p. 1356)		
CreateDisk	Creates a disk.	Write	Disk* (p. 1355)		
			aws:RequestTag/\${TagKey} (p. 1356)	aws:TagKeys (p. 1356)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateDiskFromSnapshot	Creates a disk from snapshot.	Write	Disk* (p. 1355)		
				aws:RequestTag/ {\$TagKey} (p. 1356)	aws:TagKeys (p. 1356)
CreateDiskSnapshot	Creates a disk snapshot.	Write	Disk* (p. 1355)		
				aws:RequestTag/ {\$TagKey} (p. 1356)	aws:TagKeys (p. 1356)
CreateDomain	Creates a domain resource for the specified domain name.	Write	Domain* (p. 1355)		
				aws:RequestTag/ {\$TagKey} (p. 1356)	aws:TagKeys (p. 1356)
CreateDomainEntries	Creates one or more DNS record entries for a domain resource: Address (A), canonical name (CNAME), mail exchanger (MX), name server (NS), start of authority (SOA), service locator (SRV), or text (TXT).	Write	Domain* (p. 1355)		
CreateInstanceSnapshot	Creates an instance snapshot.	Write	Instance* (p. 1355)		
			InstanceSnapshot* (p. 1355)		
				aws:RequestTag/ {\$TagKey} (p. 1356)	aws:TagKeys (p. 1356)
CreateInstances	Creates one or more instances.	Write	KeyValuePair* (p. 1355)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:RequestTag/ \${TagKey} (p. 1356)	aws:TagKeys (p. 1356)
CreateInstancesFromSnapshot	Creates one or more instances based on an instance snapshot.	Write	Instance* (p. 1355)	InstanceSnapshot* (p. 1355)	aws:RequestTag/ \${TagKey} (p. 1356)
					aws:TagKeys (p. 1356)
CreateKeyPair	Creates a key pair used to authenticate and connect to an instance.	Write	KeyPair* (p. 1355)	aws:RequestTag/ \${TagKey} (p. 1356)	aws:TagKeys (p. 1356)
CreateLoadBalancer	Creates a load balancer.	Write	LoadBalancer* (p. 1356)	aws:RequestTag/ \${TagKey} (p. 1356)	aws:TagKeys (p. 1356)
CreateLoadBalancerCertificate	Creates a load balancer TLS certificate.	Write	LoadBalancer* (p. 1356)	aws:RequestTag/ \${TagKey} (p. 1356)	aws:TagKeys (p. 1356)
CreateRelationalDatabase	Creates a new relational database.	Write	RelationalDatabase* (p. 1356)	aws:RequestTag/ \${TagKey} (p. 1356)	aws:TagKeys (p. 1356)
CreateRelationalDatabaseFromSnapshot	Creates a new relational database from a snapshot.	Write	RelationalDatabase* (p. 1356)	aws:RequestTag/ \${TagKey} (p. 1356)	aws:TagKeys (p. 1356)

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:RequestTag/ \${TagKey} (p. 1356) aws:TagKeys (p. 1356)	
CreateRelationalDatabaseSnapshot	Creates a relational database snapshot.	Write	RelationalDatabaseSnapshot* (p. 1356)	aws:RequestTag/ \${TagKey} (p. 1356) aws:TagKeys (p. 1356)	
DeleteDisk	Deletes a disk.	Write	Disk* (p. 1355)		
DeleteDiskSnapshot	Deletes a disk snapshot.	Write	Disk* (p. 1355)		
DeleteDomain	Deletes a domain resource and all of its DNS records.	Write	Domain* (p. 1355)		
DeleteDomainEntry	Deletes a DNS record entry for a domain resource.	Write	Domain* (p. 1355)		
DeleteInstance	Deletes an instance.	Write	Instance* (p. 1355)		
DeleteInstanceSnapshot	Deletes an instance snapshot.	Write	InstanceSnapshot* (p. 1355)		
DeleteKeyValuePair	Deletes a key pair used to authenticate and connect to an instance.	Write	KeyValuePair* (p. 1355)		
DeleteKnownHostEntry	Deletes the known host key or certificate used by the Amazon Lightsail browser-based SSH or RDP clients to authenticate an instance.	Write	Instance* (p. 1355)		
DeleteLoadBalancer	Deletes a load balancer.	Write	LoadBalancer* (p. 1356)		
DeleteLoadBalancerCertificate	Deletes a load balancer TLS certificate.	Write	LoadBalancer* (p. 1356)		
DeleteRelationalDatabase	Deletes a relational database.	Write	RelationalDatabase* (p. 1356)		
DeleteRelationalDatabaseSnapshot	Deletes relational database snapshot.	Write	RelationalDatabaseSnapshot* (p. 1356)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DetachDisk	Detaches a disk from an instance.	Write	Disk* (p. 1355)		
DetachInstancesFromLoadBalancer	Detaches one or more instances from a load balancer.	Write	Instance* (p. 1355)		
	LoadBalancer* (p. 1356)				
DetachStaticIp	Detaches a static IP from an instance to which it is attached.	Write	Instance* (p. 1355)		
	StaticIp* (p. 1355)				
DownloadDefaultKeyPair	Downloads the default key pair used to authenticate and connect to instances in a specific AWS Region.	Write	KeyPair* (p. 1355)		
ExportSnapshot	Exports an Amazon Lightsail snapshot to Amazon EC2.	Write			
GetActiveNames	Returns the names of all active (not deleted) resources.	Read			
GetBlueprints	Returns a list of instance images, or blueprints. You can use a blueprint to create a new instance already running a specific operating system, as well as a pre-installed application or development stack. The software that runs on your instance depends on the blueprint you define when creating the instance.	List			
GetBundles	Returns a list of instance bundles. You can use a bundle to create a new instance with a set of performance specifications, such as CPU count, disk size, RAM size, and network transfer allowance. The cost of your instance depends on the bundle you define when creating the instance.	List			
GetCloudFormation	Returns information about all CloudFormation stacks used to create Amazon EC2 resources from exported Amazon Lightsail snapshots.	List	CloudFormationStackRecord* (p. 1356)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetDisk	Returns information about a disk.	Read	Disk* (p. 1355)		
GetDiskSnapshot	Returns information about a disk snapshot.	Read	Disk* (p. 1355)		
GetDiskSnapshots	Returns information about all disk snapshots.	List	Disk* (p. 1355)		
GetDisks	Returns information about all disks.	List			
GetDomain	Returns DNS records for a domain resource.	Read	Domain* (p. 1355)		
GetDomains	Returns DNS records for all domain resources.	Read	Domain* (p. 1355)		
GetExportSnapshotRecords	Returns information about GetExportSnapshotRecords to export Amazon Lightsail snapshots to Amazon EC2.	List	ExportSnapshotRecord* (p. 1356)		
GetInstance	Returns information about an instance.	Read	Instance* (p. 1355)		
GetInstanceAccessKeys	Returns temporary keys you can use to authenticate and connect to an instance.	Write	Instance* (p. 1355)		
GetInstanceMetrics	Returns the data points for the specified metric of an instance.	Read	Instance* (p. 1355)		
GetInstancePortStates	Returns the port states of an instance.	Read	Instance* (p. 1355)		
GetInstanceSnapshot	Returns information about an instance snapshot.	Read	InstanceSnapshot* (p. 1355)		
GetInstanceSnapshots	Returns information about all instance snapshots.	List	InstanceSnapshot* (p. 1355)		
GetInstanceState	Returns the state of an instance.	Read	Instance* (p. 1355)		
GetInstances	Returns information about all instances.	Read	Instance* (p. 1355)		
GetKeyPair	Returns information about a key pair.	List	KeyValuePair* (p. 1355)		
GetKeyPairs	Returns information about all key pairs.	Read	KeyValuePair* (p. 1355)		
GetLoadBalancer	Returns information about a load balancer.	Read	LoadBalancer* (p. 1356)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetLoadBalancerMetrics	Returns the data points for the specified metric of a load balancer.	Read	LoadBalancer* (p. 1356)		
GetLoadBalancerTLSCertificates	Returns information about a Load Balancer TLS certificate.	Read	LoadBalancer* (p. 1356)		
GetLoadBalancers	Returns information about load balancers.	Read	LoadBalancer* (p. 1356)		
GetOperation	Returns information about an operation. Operations include events such as when you create an instance, allocate a static IP, attach a static IP, and so on.	Read			
GetOperations	Returns information about all operations. Operations include events such as when you create an instance, allocate a static IP, attach a static IP, and so on.	Read			
GetOperationsForResource	Returns operations for a resource	Read	Domain (p. 1355)		
Instance (p. 1355)					
InstanceSnapshot (p. 1355)					
KeyPair (p. 1355)					
StaticIp (p. 1355)					
GetRegions	Returns a list of all valid AWS Regions for Amazon Lightsail.	List			
GetRelationalDatabaseBlueprints	Returns information about a relational database.	List	RelationalDatabase* (p. 1356)		
GetRelationalDatabaseImages	Returns a list of relational database images, or blueprints. You can use a blueprint to create a new database running a specific database engine. The database engine that runs on your database depends on the blueprint you define when creating the relational database.	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetRelationalDatabaseBundles	Returns a list of relational database bundles. You can use a bundle to create a new database with a set of performance specifications, such as CPU count, disk size, RAM size, network transfer allowance, and standard of high availability. The cost of your database depends on the bundle you define when creating the relational database.	List			
GetRelationalDatabaseEvents	Returns events for a relational database.	Read			
GetRelationalDatabaseLogStreamEvents	Returns events for the specified log stream of a relational database.	Read			
GetRelationalDatabaseMetrics	Returns the log streams available for a relational database.	Read			
GetRelationalDatabaseMasterPassword	Returns the master user password for a relational database.	Write			
GetRelationalDatabaseMetrics	Returns the data points for the specified metric of a relational database.	Read			
GetRelationalDatabaseParameters	Returns the parameters of a relational database.	List			
GetRelationalDatabaseSnapshot	Returns information about a relational database snapshot.	List	RelationalDatabase* (p. 1356)		
GetRelationalDatabaseSnapshots	Returns information about all relational database snapshots.	List	RelationalDatabase* (p. 1356)		
GetRelationalDatabases	Return information about all relational databases.	Read	RelationalDatabase* (p. 1356)		
GetStaticIp	Returns information about a static IP.	Read	StaticIp* (p. 1355)		
GetStaticIps	Returns information about all static IPs.	Read	StaticIp* (p. 1355)		
ImportKeyPair	Imports a public key from a key pair.	Write	KeyValuePair* (p. 1355)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
IsVpcPeered	Returns a boolean value indicating whether the Amazon Lightsail virtual private cloud (VPC) is peered.	Read			
OpenInstancePublicPorts	Adds, or opens a public port of an instance .	Write	Instance* (p. 1355)		
PeerVpc	Tries to peer the Amazon Lightsail virtual private cloud (VPC) with the default VPC.	Write			
PutInstancePublicPorts	Sets the specified open ports for an instance , and closes all ports for every protocol not included in the request.	Write	Instance* (p. 1355)		
RebootInstance	Reboots an instance that is in a running state.	Write	Instance* (p. 1355)		
RebootRelationalDatabase	Reboots a relational database that is in a running state.	Write	RelationalDatabase* (p. 1356)		
ReleaseStaticIp	Deletes a static IP.	Write	StaticIp* (p. 1355)		
StartInstance	Starts an instance that is in a stopped state.	Write	Instance* (p. 1355)		
StartRelationalDatabase	Starts a relational database that is in a stopped state.	Write	RelationalDatabase* (p. 1356)		
StopInstance	Stops an instance that is in a running state.	Write	Instance* (p. 1355)		
StopRelationalDatabase	Stops a relational database that is in a running state.	Write	RelationalDatabase* (p. 1356)		
TagResource	Tags a resource.	Write	Disk (p. 1355)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			LoadBalancer (p. 1356)		
			RelationalDatabase (p. 1356)		
			RelationalDatabaseSnapshot (p. 1356)		
			StaticIp (p. 1355)		
			aws:RequestTag/ \${TagKey} (p. 1356)		
			aws:TagKeys (p. 1356)		
UnpeerVpc	Attempts to unpeer the Amazon Lightsail virtual private cloud (VPC) from the default VPC.	Write			
UntagResource	Untags a resource.	Write	Disk (p. 1355)		
			DiskSnapshot (p. 1355)		
			Domain (p. 1355)		
			Instance (p. 1355)		
			InstanceSnapshot (p. 1355)		
			KeyValuePair (p. 1355)		
			LoadBalancer (p. 1356)		
			RelationalDatabase (p. 1356)		
			RelationalDatabaseSnapshot (p. 1356)		
			StaticIp (p. 1355)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:RequestTag/ {\$TagKey} (p. 1356) aws:TagKeys (p. 1356)	
UpdateDomainEntry	Updates a domain recordset after it is created.	Write	Domain* (p. 1355)		
UpdateLoadBalancerAttribute	Updates a load balancer attribute, such as the health check path and session stickiness.	Write	LoadBalancer* (p. 1356)		
UpdateRelationalDatabase	Updates a relational database.	Write	RelationalDatabase* (p. 1356)		
UpdateRelationalDatabaseParameters	Updates the parameters of a relational database.	Write			

Resource Types Defined by Amazon Lightsail

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1344\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
Domain	arn:\${Partition}:lightsail:\${Region}: \${Account}:Domain/\${Id}	aws:ResourceTag/ {\$TagKey} (p. 1356)
Instance	arn:\${Partition}:lightsail:\${Region}: \${Account}:Instance/\${Id}	aws:ResourceTag/ {\$TagKey} (p. 1356)
InstanceSnapshot	arn:\${Partition}:lightsail:\${Region}: \${Account}:InstanceSnapshot/\${Id}	aws:ResourceTag/ {\$TagKey} (p. 1356)
KeyPair	arn:\${Partition}:lightsail:\${Region}: \${Account}:KeyPair/\${Id}	aws:ResourceTag/ {\$TagKey} (p. 1356)
StaticIp	arn:\${Partition}:lightsail:\${Region}: \${Account}:StaticIp/\${Id}	aws:ResourceTag/ {\$TagKey} (p. 1356)
Disk	arn:\${Partition}:lightsail:\${Region}: \${Account}:Disk/\${Id}	aws:ResourceTag/ {\$TagKey} (p. 1356)
DiskSnapshot	arn:\${Partition}:lightsail:\${Region}: \${Account}:DiskSnapshot/\${Id}	aws:ResourceTag/ {\$TagKey} (p. 1356)

Resource Types	ARN	Condition Keys
LoadBalancer	arn:\${Partition}:lightsail:\${Region}: \${Account}:LoadBalancer/\${Id}	aws:ResourceTag/\${TagKey} (p. 1356)
PeeredVpc	arn:\${Partition}:lightsail:\${Region}: \${Account}:PeeredVpc/\${Id}	
LoadBalancerTlsCertificate	arn:\${Partition}:lightsail:\${Region}: \${Account}:LoadBalancerTlsCertificate/\${Id}	
ExportSnapshotRecord	arn:\${Partition}:lightsail:\${Region}: \${Account}:ExportSnapshotRecord/\${Id}	
CloudFormationStackRecord	arn:\${Partition}:lightsail:\${Region}: \${Account}:CloudFormationStackRecord/\${Id}	
RelationalDatabase	arn:\${Partition}:lightsail:\${Region}: \${Account}:RelationalDatabase/\${Id}	aws:ResourceTag/\${TagKey} (p. 1356)
RelationalDatabaseSnapshot	arn:\${Partition}:lightsail:\${Region}: \${Account}:RelationalDatabaseSnapshot/\${Id}	aws:ResourceTag/\${TagKey} (p. 1356)

Condition Keys for Amazon Lightsail

Amazon Lightsail defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Filters actions based on the presence of tag key-value pairs in the request	String
aws:ResourceTag/\${TagKey}	Filters actions based on tag key-value pairs attached to the resource	String
aws:TagKeys	Filters actions based on the presence of tag keys in the request	String

Actions, Resources, and Condition Keys for Amazon Machine Learning

Amazon Machine Learning (service prefix: `machinelearning`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Machine Learning \(p. 1357\)](#)
- [Resource Types Defined by Amazon Machine Learning \(p. 1360\)](#)
- [Condition Keys for Amazon Machine Learning \(p. 1360\)](#)

Actions Defined by Amazon Machine Learning

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AddTags	Adds one or more tags to an object, up to a limit of 10. Each tag consists of a key and an optional value	Tagging	batchprediction (p. 1360)		
			datasource (p. 1360)		
			evaluation (p. 1360)		
			mlmodel (p. 1360)		
CreateBatchPrediction	Generates predictions for a group of observations	Write	batchprediction* (p. 1360)		
			datasource* (p. 1360)		
			mlmodel* (p. 1360)		
CreateDataSourceFromRDS	Creates a DataSource object from an Amazon RDS	Write	datasource* (p. 1360)		
CreateDataSourceFromRedshift	Creates a DataSource from a database hosted on an Amazon Redshift cluster	Write	datasource* (p. 1360)		
CreateDataSourceFromS3	Creates a DataSource object from S3	Write	datasource* (p. 1360)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateEvaluation	Creates a new Evaluation of an MLModel	Write	datasource* (p. 1360)		
	evaluation* (p. 1360)				
	mlmodel* (p. 1360)				
CreateMLModel	Creates a new MLModel	Write	datasource* (p. 1360)		
	mlmodel* (p. 1360)				
CreateRealtimeEndpoint	Creates a real-time endpoint for the ML Model	Write	mlmodel* (p. 1360)		
DeleteBatchPrediction	Assigns the DELETED status to BatchPrediction, rendering it unusable	Write	batchprediction* (p. 1360)		
DeleteDataSource	Assigns the DELETED status to a DataSource, rendering it unusable	Write	datasource* (p. 1360)		
DeleteEvaluation	Assigns the DELETED status to an Evaluation, rendering it unusable	Write	evaluation* (p. 1360)		
DeleteMLModel	Assigns the DELETED status to an MLModel, rendering it unusable	Write	mlmodel* (p. 1360)		
DeleteRealtimeEndpoint	Deletes a real time endpoint of the ML Model	Write	mlmodel* (p. 1360)		
DeleteTags	Deletes the specified tags associated with an ML object. After this operation is complete, you can't recover deleted tags	Tagging	batchprediction (p. 1360)		
datasource (p. 1360)					
evaluation (p. 1360)					
mlmodel (p. 1360)					
DescribeBatchPredictions	Returns a list of BatchPrediction operations that match the search criteria in the request	List			
DescribeDataSources	Returns a list of DataSource that match the search criteria in the request	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeEvaluations	Returns a list of DescribeEvaluations that match the search criteria in the request	List			
DescribeMLModel	Returns a list of MLModel that match the search criteria in the request	List			
DescribeTags	Describes one or more of the tags for your Amazon ML object	List	batchprediction (p. 1360)		
			datasource (p. 1360)		
			evaluation (p. 1360)		
			mlmodel (p. 1360)		
GetBatchPrediction	Returns a BatchPrediction that includes detailed metadata, status, and data file information	Read	batchprediction* (p. 1360)		
GetDataSource	Returns a DataSource that includes metadata and data file information, as well as the current status of the DataSource	Read	datasource* (p. 1360)		
GetEvaluation	Returns an Evaluation that includes metadata as well as the current status of the Evaluation	Read	datasource* (p. 1360)		
GetMLModel	Returns an MLModel that includes detailed metadata, and data source information as well as the current status of the MLModel	Read	mlmodel* (p. 1360)		
Predict	Generates a prediction for the observation using the specified ML Model	Write	mlmodel* (p. 1360)		
UpdateBatchPrediction	Updates the BatchPredictionName of a BatchPrediction	Write	batchprediction* (p. 1360)		
UpdateDataSource	Updates the DataSourceName of a DataSource	Write	datasource* (p. 1360)		
UpdateEvaluation	Updates the EvaluationName of an Evaluation	Write	evaluation* (p. 1360)		
UpdateMLModel	Updates the MLModelName and the ScoreThreshold of an MLModel	Write	mlmodel* (p. 1360)		

Resource Types Defined by Amazon Machine Learning

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 1357\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
batchprediction	arn:\${Partition}:machinelearning:\${Region}:\${Account}:batchprediction/\${BatchPredictionId}	
datasource	arn:\${Partition}:machinelearning:\${Region}:\${Account}:datasource/\${DatasourceId}	
evaluation	arn:\${Partition}:machinelearning:\${Region}:\${Account}:evaluation/\${EvaluationId}	
mlmodel	arn:\${Partition}:machinelearning:\${Region}:\${Account}:mlmodel/\${MlModelId}	

Condition Keys for Amazon Machine Learning

Machine Learning has no service-specific context keys that can be used in the Condition element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Amazon Macie

Amazon Macie (service prefix: `macie`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Macie \(p. 1360\)](#)
- [Resource Types Defined by Amazon Macie \(p. 1361\)](#)
- [Condition Keys for Amazon Macie \(p. 1361\)](#)

Actions Defined by Amazon Macie

You can specify the following actions in the Action element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AssociateMember	Enables the user to associate a specified AWS account with Amazon Macie as a member account.	Write			
AssociateS3Resources	Enables the user to associate specified S3 resources with Amazon Macie for monitoring and data classification.	Write		aws:SourceArn (p. 1362)	
DisassociateMember	Enables the user to remove the specified member account from Amazon Macie.	Write			
DisassociateS3Resources	Enables the user to remove specified S3 resources from being monitored by Amazon Macie.	Write		aws:SourceArn (p. 1362)	
ListMemberAccounts	Enables the user to list all Amazon Macie member accounts for the current Macie master account.	List			
ListS3Resources	Enables the user to list all the S3 resources associated with Amazon Macie.	List			
UpdateS3Resources	Enables the user to update the classification types for the specified S3 resources.	Write		aws:SourceArn (p. 1362)	

Resource Types Defined by Amazon Macie

Amazon Macie does not support specifying a resource ARN in the `Resource` element of an IAM policy statement. To allow access to Amazon Macie, specify “`Resource`”: “`*`” in your policy.

Condition Keys for Amazon Macie

Amazon Macie defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:SourceArn	Allow access to the specified actions only when the request operates on the specified aws resource	Arn

Actions, Resources, and Condition Keys for Manage Amazon API Gateway

Manage Amazon API Gateway (service prefix: `apigateway`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Manage Amazon API Gateway \(p. 1362\)](#)
- [Resource Types Defined by Manage Amazon API Gateway \(p. 1364\)](#)
- [Condition Keys for Manage Amazon API Gateway \(p. 1364\)](#)

Actions Defined by Manage Amazon API Gateway

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
<code>DELETE</code>	Used to delete resources	Write	<code>apigateway-general*</code> (p. 1364)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:RequestTag/ \${TagKey} (p. 1364) aws:TagKeys (p. 1364)	
GET	Used to get information about resources	Read	apigateway-general* (p. 1364)		
PATCH	Used to update resources	Write	apigateway-general* (p. 1364)	aws:RequestTag/ \${TagKey} (p. 1364) aws:TagKeys (p. 1364)	
POST	Used to create child resources	Write	apigateway-general* (p. 1364)	aws:RequestTag/ \${TagKey} (p. 1364) aws:TagKeys (p. 1364)	
PUT	Used to update resources (and, although not recommended, can be used to create child resources)	Write	apigateway-general* (p. 1364)	aws:RequestTag/ \${TagKey} (p. 1364) aws:TagKeys (p. 1364)	
SetWebACL	Gives WebAcl permissions to WAF	Write	apigateway-general* (p. 1364)		
UpdateRestApiPolicy	Used to update the Resource Policy for a given API	Write	apigateway-general* (p. 1364)		

Resource Types Defined by Manage Amazon API Gateway

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1362\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
apigateway-general	<code>arn:\${Partition}:apigateway:\${Region}::\${ApiGatewayResourcePath}</code>	<code>aws:ResourceTag/\${TagKey}</code> (p. 1364)

Condition Keys for Manage Amazon API Gateway

Manage Amazon API Gateway defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
<code>aws:RequestTag/\${TagKey}</code>		String
<code>aws:ResourceTag/\${TagKey}</code>		String
<code>aws:TagKeys</code>		String

Actions, Resources, and Condition Keys for AWS Managed Apache Cassandra Service

AWS Managed Apache Cassandra Service (service prefix: `cassandra`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Managed Apache Cassandra Service \(p. 1365\)](#)
- [Resource Types Defined by AWS Managed Apache Cassandra Service \(p. 1365\)](#)
- [Condition Keys for AWS Managed Apache Cassandra Service \(p. 1366\)](#)

Actions Defined by AWS Managed Apache Cassandra Service

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
Alter	Grants permission to alter a keyspace or table	Write	keyspace (p. 1366)		
			table (p. 1366)		
Create	Grants permission to create a keyspace or table	Write	keyspace (p. 1366)		
			table (p. 1366)		
Drop	Grants permission to drop a keyspace or table	Write	keyspace (p. 1366)		
			table (p. 1366)		
Modify	Grants permission to INSERT, UPDATE or DELETE data in a table	Write	table* (p. 1366)		
Select	Grants permission to SELECT data from a table	Read	table* (p. 1366)		

Resource Types Defined by AWS Managed Apache Cassandra Service

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1365\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
keyspace	arn:\${Partition}:cassandra:\${Region}: \${Account}:keyspace/\${KeyspaceName}	
table	arn:\${Partition}:cassandra:\${Region}: \${Account}:keyspace/\${KeyspaceName}/table/ \${tableName}	

Condition Keys for AWS Managed Apache Cassandra Service

MCS has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Amazon Managed Blockchain

Amazon Managed Blockchain (service prefix: `managedblockchain`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Managed Blockchain \(p. 1366\)](#)
- [Resource Types Defined by Amazon Managed Blockchain \(p. 1368\)](#)
- [Condition Keys for Amazon Managed Blockchain \(p. 1369\)](#)

Actions Defined by Amazon Managed Blockchain

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateMember	Grants permission to create a member of an Amazon Managed Blockchain network.	Write	network* (p. 1369)		
CreateNetwork	Grants permission to create an Amazon Managed Blockchain network.	Write			
CreateNode	Grants permission to create a node within a member of an Amazon Managed Blockchain network.	Write	member* (p. 1369)		
CreateProposal	Grants permission to create a proposal that other blockchain network members can vote on to add or remove a member in an Amazon Managed Blockchain network.	Write	network* (p. 1369)		
DeleteMember	Grants permission to delete a member and all associated resources from an Amazon Managed Blockchain network.	Write	member* (p. 1369)		
DeleteNode	Grants permission to delete a node from a member of an Amazon Managed Blockchain network.	Write	node* (p. 1369)		
GetMember	Grants permission to return detailed information about a member of an Amazon Managed Blockchain network.	Read	member* (p. 1369)		
GetNetwork	Grants permission to return detailed information about an Amazon Managed Blockchain network.	Read	network* (p. 1369)		
GetNode	Grants permission to return detailed information about a node within a member of an Amazon Managed Blockchain network.	Read	node* (p. 1369)		
GetProposal	Grants permission to return detailed information about a proposal of an Amazon Managed Blockchain network.	Read	proposal* (p. 1369)		
ListInvitations	Grants permission to list the invitations extended to the	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
	active AWS account from any Managed Blockchain network.				
ListMembers	Grants permission to list the members of an Amazon Managed Blockchain network and the properties of their memberships.	List	network* (p. 1369)		
ListNetworks	Grants permission to return information about the Amazon Managed Blockchain networks in which the current AWS account has members.	List			
ListNodes	Grants permission to list the nodes within a member of an Amazon Managed Blockchain network.	List	member* (p. 1369)		
ListProposalVotes	Grants permission to list all votes for a proposal, including the value of the vote and the unique identifier of the member that cast the vote for the given Amazon Managed Blockchain network.	List	proposal* (p. 1369)		
ListProposals	Grants permission to list proposals for the given Amazon Managed Blockchain network.	List	network* (p. 1369)		
RejectInvitation	Grants permission to reject the invitation to join the blockchain network.	Write	invitation* (p. 1369)		
VoteOnProposal	Grants permission to cast a vote for a proposal on behalf of the blockchain network member specified.	Write	proposal* (p. 1369)		

Resource Types Defined by Amazon Managed Blockchain

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1366\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
network	arn:\${Partition}:managedblockchain:\${Region}::networks/\${NetworkId}	
member	arn:\${Partition}:managedblockchain:\${Region}::\$Account:members/\${MemberId}	
node	arn:\${Partition}:managedblockchain:\${Region}::\$Account:nodes/\${NodeId}	
proposal	arn:\${Partition}:managedblockchain:\${Region}::proposals/\${ProposalId}	
invitation	arn:\${Partition}:managedblockchain:\${Region}::\$Account:invitations/\${InvitationId}	

Condition Keys for Amazon Managed Blockchain

Managed Blockchain has no service-specific context keys that can be used in the Condition element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Amazon Managed Streaming for Kafka

Amazon Managed Streaming for Kafka (service prefix: `kafka`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).

Topics

- [Actions Defined by Amazon Managed Streaming for Kafka \(p. 1369\)](#)
- [Resource Types Defined by Amazon Managed Streaming for Kafka \(p. 1371\)](#)
- [Condition Keys for Amazon Managed Streaming for Kafka \(p. 1371\)](#)

Actions Defined by Amazon Managed Streaming for Kafka

You can specify the following actions in the Action element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the Resource element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type.

Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateCluster	Grants permission to create a cluster.	Write		aws:RequestTag \${TagKey} (p. 1372)	DescribeSecurityGroups ec2:DescribeSubnets
				aws:TagKeys (p. 1372)	ec2:DescribeVpcs iam:AttachRolePolicy iam>CreateServiceLinkedRole iam:PutRolePolicy kms>CreateGrant kms:DescribeKey
CreateConfiguration	Grants permission to create a configuration.	Write			
DeleteCluster	Grants permission to delete a cluster.	Write			
DescribeCluster	Grants permission to describe a cluster.	Read			
DescribeClusterOperation	Returns a description of the operation specified by the ARN.	Read			
DescribeConfiguration	Grants permission to describe a configuration.	Read			
DescribeConfigurationRevision	Grants permission to describe a configuration revision.	Read			
GetBootstrapBrokerConnection	Grants permission to get connection details for the broker nodes in a cluster.	Read			
ListClusterOperations	Returns a list of all the operations that have been performed on the specified MSK cluster.	Read			
ListClusters	Grants permission to return a list of all clusters in the current account.	List			
ListConfigurations	Grants permission to return a list of all configurations in the current account.	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListNodes	Grants permission to return a list of nodes in a cluster.	List			
ListTagsForResource	Grants permission to list tags of a MSK resource.	Read	cluster (p. 1371)		
TagResource	Grants permission to tag a MSK resource.	Tagging	cluster (p. 1371)		
				aws:RequestTag/\${TagKey} (p. 1372) aws:TagKeys (p. 1372)	
UntagResource	Grants permission to remove tags from a MSK resource.	Tagging	cluster (p. 1371)		
				aws:TagKeys (p. 1372)	
UpdateBrokerStorage	Updates the storage size of the broker nodes of the cluster	Write			
UpdateClusterConfiguration	Update Kafka configuration for a cluster.	Write			

Resource Types Defined by Amazon Managed Streaming for Kafka

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 1369\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
cluster	<code>arn:\${Partition}:kafka:\${Region}:\${Account}:cluster/\${ClusterName}/\${UUID}</code>	aws:ResourceTag/\${TagKey} (p. 1372)

Condition Keys for Amazon Managed Streaming for Kafka

Amazon Managed Streaming for Kafka defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the [IAM Policy Reference](#).

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Filter requests based on the allowed set of values for each of the tags	String
aws:ResourceTag/\${TagKey}	Filter actions based on tag-value associated with a MSK resource.	String
aws:TagKeys	Filter requests based on the presence of mandatory tag keys in the request	String

Actions, Resources, and Condition Keys for AWS Marketplace

AWS Marketplace (service prefix: `aws-marketplace`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Marketplace \(p. 1372\)](#)
- [Resource Types Defined by AWS Marketplace \(p. 1374\)](#)
- [Condition Keys for AWS Marketplace \(p. 1374\)](#)

Actions Defined by AWS Marketplace

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AcceptAgreement	Allows users to approve a pending subscription request (for providers who	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
	provide products that require subscription verification).				
CancelAgreement	Allows users to cancel pending subscription requests for products that require subscription verification.	Write			
GetAgreementApprovers	Allows users to view the details of their pending subscription requests (for providers who provide products that require subscription verification).	Read			
GetAgreementRequesters	Allows users to view the details of their subscription requests for data products that require subscription verification.	Read			
ListAgreementApplicants	Allows users to list their pending subscription requests (for providers who provide products that require subscription verification).	List			
ListAgreementRecipients	Allows users to list their subscription requests for products that require subscription verification.	List			
RejectAgreementApplicants	Allows users to decline pending subscription requests (for providers who provide products that require subscription verification).	Write			
Subscribe	Allows users to subscribe to AWS Marketplace products. Includes the ability to send a subscription request for products that require subscription verification. Includes the ability to enable auto-renewal for an existing subscription.	Write			
Unsubscribe	Allows users to remove subscriptions to AWS Marketplace products. Includes the ability to disable auto-renewal for an existing subscription.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UpdateAgreement	Allows users to make changes to a prospective subscription request, including the ability to delete the prospective subscriber's information (for providers who provide products that require subscription verification).	Write			
ViewSubscription	Allows users to see their account's subscriptions.	List			

Resource Types Defined by AWS Marketplace

AWS Marketplace does not support specifying a resource ARN in the `Resource` element of an IAM policy statement. To allow access to AWS Marketplace, specify “`Resource`”: “`*`” in your policy.

Condition Keys for AWS Marketplace

Marketplace has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for AWS Marketplace Catalog

AWS Marketplace Catalog (service prefix: `aws-marketplace`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Marketplace Catalog \(p. 1374\)](#)
- [Resource Types Defined by AWS Marketplace Catalog \(p. 1375\)](#)
- [Condition Keys for AWS Marketplace Catalog \(p. 1375\)](#)

Actions Defined by AWS Marketplace Catalog

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the Resource element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CancelChangeSet	Cancels a running change set.	Write			
DescribeChangeSets	Returns the details of an existing change set.	Read			
DescribeEntity	Returns the details of an existing entity.	Read			
ListChangeSets	Lists existing change sets.	Read			
ListEntities	Lists existing entities.	Read			
StartChangeSet	Requests a new change set.	Write		catalog:ChangeType (p. 1376)	

Resource Types Defined by AWS Marketplace Catalog

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 1374\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
Entity	arn:\${Partition}:aws-marketplace:\${Region}: \${Account}://\${Catalog}/\${EntityType}/ \${ResourceId}	
ChangeSet	arn:\${Partition}:aws-marketplace: \${Region}:\${Account}://\${Catalog}/ChangeSet/ \${ResourceId}	

Condition Keys for AWS Marketplace Catalog

AWS Marketplace Catalog defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
catalog:ChangeType	Enables you to verify change type in the StartChangeSet request.	String

Actions, Resources, and Condition Keys for AWS Marketplace Entitlement Service

AWS Marketplace Entitlement Service (service prefix: `aws-marketplace`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

Topics

- [Actions Defined by AWS Marketplace Entitlement Service \(p. 1376\)](#)
- [Resource Types Defined by AWS Marketplace Entitlement Service \(p. 1376\)](#)
- [Condition Keys for AWS Marketplace Entitlement Service \(p. 1377\)](#)

Actions Defined by AWS Marketplace Entitlement Service

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
<code>GetEntitlements</code>	Retrieves entitlement values for a given product. The results can be filtered based on customer identifier or product dimensions	Read			

Resource Types Defined by AWS Marketplace Entitlement Service

AWS Marketplace Entitlement Service does not support specifying a resource ARN in the `Resource` element of an IAM policy statement. To allow access to AWS Marketplace Entitlement Service, specify `"Resource": "*"` in your policy.

Condition Keys for AWS Marketplace Entitlement Service

Marketplace Entitlement has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for AWS Marketplace Image Building Service

AWS Marketplace Image Building Service (service prefix: `aws-marketplace`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Marketplace Image Building Service \(p. 1377\)](#)
- [Resource Types Defined by AWS Marketplace Image Building Service \(p. 1378\)](#)
- [Condition Keys for AWS Marketplace Image Building Service \(p. 1378\)](#)

Actions Defined by AWS Marketplace Image Building Service

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeBuilds [permission only]	Describes Image Builds identified by a build Id	Read			
ListBuilds [permission only]	Lists Image Builds.	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
StartBuild [permission only]	Starts an Image Build	Write			

Resource Types Defined by AWS Marketplace Image Building Service

AWS Marketplace Image Building Service does not support specifying a resource ARN in the `Resource` element of an IAM policy statement. To allow access to AWS Marketplace Image Building Service, specify `"Resource": "*"` in your policy.

Condition Keys for AWS Marketplace Image Building Service

Marketplace Image Build has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for AWS Marketplace Management Portal

AWS Marketplace Management Portal (service prefix: `aws-marketplace-management`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Marketplace Management Portal \(p. 1378\)](#)
- [Resource Types Defined by AWS Marketplace Management Portal \(p. 1379\)](#)
- [Condition Keys for AWS Marketplace Management Portal \(p. 1379\)](#)

Actions Defined by AWS Marketplace Management Portal

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type.

Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
uploadFiles	Allows a user to access the File Upload page inside the AWS Marketplace Management Portal.	Write			
viewMarketing	Allows a user to access the Marketing page inside the AWS Marketplace Management Portal.	List			
viewReports	Allows a user to access the Reports page inside the AWS Marketplace Management Portal.	List			
viewSettings	Allows a user to access the Settings page inside the AWS Marketplace Management Portal.	List			
viewSupport	Allows a user to access the Customer Support Eligibility page inside the AWS Marketplace Management Portal.	List			

Resource Types Defined by AWS Marketplace Management Portal

AWS Marketplace Management Portal does not support specifying a resource ARN in the Resource element of an IAM policy statement. To allow access to AWS Marketplace Management Portal, specify "Resource": "*" in your policy.

Condition Keys for AWS Marketplace Management Portal

Marketplace Portal has no service-specific context keys that can be used in the Condition element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for AWS Marketplace Metering Service

AWS Marketplace Metering Service (service prefix: aws-marketplace) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).

- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Marketplace Metering Service \(p. 1380\)](#)
- [Resource Types Defined by AWS Marketplace Metering Service \(p. 1380\)](#)
- [Condition Keys for AWS Marketplace Metering Service \(p. 1381\)](#)

Actions Defined by AWS Marketplace Metering Service

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
BatchMeterUsage	Called from a SaaS application listed on the AWS Marketplace to post metering records for a set of customers.	Write			
MeterUsage	Emits metering records.	Write			
RegisterUsage	Allows you to verify that the customer running your paid software is subscribed to your product on AWS Marketplace, enabling you to guard against unauthorized use. Meters software use per ECS task, per hour, with usage prorated to the second.	Write			
ResolveCustomer	Resolves a registration token to obtain a <code>CustomerId</code> and product code.	Write			

Resource Types Defined by AWS Marketplace Metering Service

AWS Marketplace Metering Service does not support specifying a resource ARN in the `Resource` element of an IAM policy statement. To allow access to AWS Marketplace Metering Service, specify `"Resource": "*"` in your policy.

Condition Keys for AWS Marketplace Metering Service

Marketplace Metering has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for AWS Marketplace Procurement Systems Integration

AWS Marketplace Procurement Systems Integration (service prefix: `aws-marketplace`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Marketplace Procurement Systems Integration \(p. 1381\)](#)
- [Resource Types Defined by AWS Marketplace Procurement Systems Integration \(p. 1382\)](#)
- [Condition Keys for AWS Marketplace Procurement Systems Integration \(p. 1382\)](#)

Actions Defined by AWS Marketplace Procurement Systems Integration

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
<code>DescribeProcurementSystemIntegration</code> [permission only]	Describes the Procurement System Integration configuration (e.g. Coupa) for the individual account, or for the entire AWS Organization if one exists. This action can only be performed by the	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
	master account if using an AWS Organization.				
PutProcurementSystemIntegration	Creates or updates the Procurement System integration configuration (e.g. Coupa) for the individual account, or for the entire AWS Organization if one exists. This action can only be performed by the master account if using an AWS Organization.	Write			

Resource Types Defined by AWS Marketplace Procurement Systems Integration

AWS Marketplace Procurement Systems Integration does not support specifying a resource ARN in the Resource element of an IAM policy statement. To allow access to AWS Marketplace Procurement Systems Integration, specify "Resource": "*" in your policy.

Condition Keys for AWS Marketplace Procurement Systems Integration

Marketplace Procurement Integration has no service-specific context keys that can be used in the Condition element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Amazon Mechanical Turk

Amazon Mechanical Turk (service prefix: `mechanicalturk`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Mechanical Turk \(p. 1382\)](#)
- [Resource Types Defined by Amazon Mechanical Turk \(p. 1387\)](#)
- [Condition Keys for Amazon Mechanical Turk \(p. 1387\)](#)

Actions Defined by Amazon Mechanical Turk

You can specify the following actions in the Action element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AcceptQualification	The <code>AcceptQualification</code> operation grants a Worker's request for a Qualification	Write			
ApproveAssignment	The <code>ApproveAssignment</code> operation approves the results of a completed assignment	Write			
AssociateQualificationWithWorker	The <code>AssociateQualificationWithWorker</code> operation gives a Worker a Qualification	Write			
CreateAdditionalAssignmentForHIT	The <code>CreateAdditionalAssignmentForHIT</code> operation increases the maximum number of assignments of an existing HIT	Write			
CreateHIT	The <code>CreateHIT</code> operation creates a new HIT (Human Intelligence Task)	Write			
CreateHITType	The <code>CreateHITType</code> operation creates a new HIT type	Write			
CreateHITWithHITType	The <code>CreateHITWithHITType</code> operation creates a new Human Intelligence Task (HIT) using an existing HITTypeID generated by the <code>CreateHITType</code> operation	Write			
CreateQualificationType	The <code>CreateQualificationType</code> operation creates a new Qualification type, which is represented by a QualificationType data structure	Write			
CreateWorkerBlock	The <code>CreateWorkerBlock</code> operation allows you to prevent a Worker from working on your HITs	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteHIT	The DeleteHIT operation disposes of a HIT that is no longer needed	Write			
DeleteQualificationType	The DeleteQualificationType operation disposes a Qualification type and disposes any HIT types that are associated with the Qualification type	Write			
DeleteWorkerBlock	The DeleteWorkerBlock operation allows you to reinstate a blocked Worker to work on your HITs	Write			
DisassociateQualificationFromWorker	The DisassociateQualificationFromWorker operation revokes a previously granted Qualification from a user	Write			
GetAccountBalance	The GetAccountBalance operation retrieves the amount of money in your Amazon Mechanical Turk account	Read			
GetAssignment	The GetAssignment retrieves an assignment with an AssignmentStatus value of Submitted, Approved, or Rejected, using the assignment's ID	Read			
GetFileUploadURL	The GetFileUploadURL operation generates and returns a temporary URL	Read			
GetHIT	The GetHIT operation retrieves the details of the specified HIT	Read			
GetQualificationScore	The GetQualificationScore operation returns the value of a Worker's Qualification for a given Qualification type	Read			
GetQualificationType	The GetQualificationType operation retrieves information about a Qualification type using its ID	Read			
ListAssignmentsForHIT	The ListAssignmentsForHIT operation retrieves completed assignments for a HIT	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListBonusPayments	The ListBonusPayments operation retrieves the amounts of bonuses you have paid to Workers for a given HIT or assignment	List			
ListHITs	The ListHITs operation returns all of a Requester's HITs	List			
ListHITsForQualificationType	The ListHITsForQualificationType operation returns the HITs that use the given QualificationType for a QualificationRequirement	List			
ListQualificationRequests	The ListQualificationRequests operation retrieves requests for Qualifications of a particular Qualification type	List			
ListQualificationTypes	The ListQualificationTypes operation searches for Qualification types using the specified search query, and returns a list of Qualification types	List			
ListReviewPolicyResultsForHIT	The ListReviewPolicyResultsForHIT operation retrieves the computed results and the actions taken in the course of executing your Review Policies during a CreateHIT operation	List			
ListReviewableHITs	The ListReviewableHITs operation returns all of a Requester's HITs that have not been approved or rejected	List			
ListWorkerBlocks	The ListWorkersBlocks operation retrieves a list of Workers who are blocked from working on your HITs	List			
ListWorkersWithQualificationType	The ListWorkersWithQualificationType operation returns all of the Workers with a given Qualification type	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
NotifyWorkers	The NotifyWorkers operation sends an email to one or more Workers that you specify with the Worker ID	Write			
RejectAssignment	The RejectAssignment operation rejects the results of a completed assignment	Write			
RejectQualificationRequest	The RejectQualificationRequest operation rejects a user's request for a Qualification	Write			
SendBonus	The SendBonus operation issues a payment of money from your account to a Worker	Write			
SendTestEventNotification	The SendTestEventNotification operation causes Amazon Mechanical Turk to send a notification message as if a HIT event occurred, according to the provided notification specification	Write			
UpdateExpirationForHIT	The UpdateExpirationForHIT operation allows you extend the expiration time of a HIT beyond its current expiration or expire a HIT immediately	Write			
UpdateHITReviewStatus	The UpdateHITReviewStatus operation toggles the status of a HIT	Write			
UpdateHITTypeOfHIT	The UpdateHITTypeOfHIT operation allows you to change the HITType properties of a HIT	Write			
UpdateNotificationSettings	The UpdateNotificationSettings operation creates, updates, disables or re-enables notifications for a HIT type	Write			
UpdateQualificationType	The UpdateQualificationType operation modifies the attributes of an existing Qualification type, which is represented by a QualificationType data structure	Write			

Resource Types Defined by Amazon Mechanical Turk

Amazon Mechanical Turk does not support specifying a resource ARN in the `Resource` element of an IAM policy statement. To allow access to Amazon Mechanical Turk, specify “`Resource`”: “`*`” in your policy.

Condition Keys for Amazon Mechanical Turk

MechanicalTurk has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Amazon Message Delivery Service

Amazon Message Delivery Service (service prefix: `ec2messages`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

Topics

- [Actions Defined by Amazon Message Delivery Service \(p. 1387\)](#)
- [Resource Types Defined by Amazon Message Delivery Service \(p. 1388\)](#)
- [Condition Keys for Amazon Message Delivery Service \(p. 1388\)](#)

Actions Defined by Amazon Message Delivery Service

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources (“`*`”) in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AcknowledgeMessage	Acknowledges a message, signaling it will not be delivered again	Write			
DeleteMessage	Deletes a message	Write			
FailMessage	Fails a message, signifying the message could not be processed successfully, ensuring it cannot be replied to or delivered again	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetEndpoint	Routes traffic to the correct endpoint based on the given destination for the messages	Read			
GetMessages	Delivers messages to clients/instances using long polling	Read			
SendReply	Sends replies from clients/instances to upstream service	Write			

Resource Types Defined by Amazon Message Delivery Service

Amazon Message Delivery Service does not support specifying a resource ARN in the `Resource` element of an IAM policy statement. To allow access to Amazon Message Delivery Service, specify “`Resource`”: “`*`” in your policy.

Condition Keys for Amazon Message Delivery Service

EC2 Messages has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for AWS Migration Hub

AWS Migration Hub (service prefix: `mgh`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Migration Hub \(p. 1388\)](#)
- [Resource Types Defined by AWS Migration Hub \(p. 1390\)](#)
- [Condition Keys for AWS Migration Hub \(p. 1390\)](#)

Actions Defined by AWS Migration Hub

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources (“`*`”) in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you

specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AssociateCreatedArtifact	Associate a given AWS artifact to a MigrationTask	Write	migrationTask* (p. 1390)		
AssociateDiscoveredResource	Associate a given ADS resource to a MigrationTask	Write	migrationTask* (p. 1390)		
CreateHomeRegionControl	Create a Migration Hub Home Region Control	Write			
CreateProgressUpdateStream	Create a ProgressUpdateStream	Write	progressUpdateStream* (p. 1390)		
DeleteProgressUpdateStream	Delete a ProgressUpdateStream	Write	progressUpdateStream* (p. 1390)		
DescribeApplicationService	Get an Application Discovery Service Application's state	Read			
DescribeHomeRegionControls	List Home Region Controls	List			
DescribeMigrationTask	Describe a MigrationTask	Read	migrationTask* (p. 1390)		
DisassociateCreatedArtifact	Disassociate a given AWS artifact from a MigrationTask	Write	migrationTask* (p. 1390)		
DisassociateDiscoveredResource	Disassociate a given ADS resource from a MigrationTask	Write	migrationTask* (p. 1390)		
GetHomeRegion	Get the Migration Hub Home Region	Read			
ImportMigrationTask	Import a MigrationTask	Write	migrationTask* (p. 1390)		
ListCreatedArtifacts	List associated created artifacts for a MigrationTask	List	migrationTask* (p. 1390)		
ListDiscoveredResources	List associated ADS resources from a MigrationTask	List	migrationTask* (p. 1390)		
ListMigrationTasks	List MigrationTasks	List			
ListProgressUpdateStreams	List ProgressUpdateStreams	List			
NotifyApplicationService	Update an Application Discovery Service Application's state	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
NotifyMigrationTaskState	Notify latest MigrationTask state	Write	migrationTask* (p. 1390)		
PutResourceAttributes	Put ResourceAttributes	Write	migrationTask* (p. 1390)		

Resource Types Defined by AWS Migration Hub

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1388\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
progressUpdateStream	arn:\${Partition}:mgh:\${Region}: \${Account}:progressUpdateStream/\${Stream}	
migrationTask	arn:\${Partition}:mgh:\${Region}: \${Account}:progressUpdateStream/\${Stream}/ migrationTask/\${Task}	

Condition Keys for AWS Migration Hub

Migration Hub has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Amazon Mobile Analytics

Amazon Mobile Analytics (service prefix: `mobileanalytics`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Mobile Analytics \(p. 1391\)](#)
- [Resource Types Defined by Amazon Mobile Analytics \(p. 1391\)](#)
- [Condition Keys for Amazon Mobile Analytics \(p. 1391\)](#)

Actions Defined by Amazon Mobile Analytics

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
<code>GetFinancialReports</code>	Grant access to financial metrics for an app	Read			
<code>GetReports</code>	Grant access to standard metrics for an app	Read			
<code>PutEvents</code>	The PutEvents operation records one or more events	Write			

Resource Types Defined by Amazon Mobile Analytics

Amazon Mobile Analytics does not support specifying a resource ARN in the `Resource` element of an IAM policy statement. To allow access to Amazon Mobile Analytics, specify “Resource”: “*” in your policy.

Condition Keys for Amazon Mobile Analytics

Mobile Analytics has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for AWS Mobile Hub

AWS Mobile Hub (service prefix: `mobilehub`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Mobile Hub \(p. 1392\)](#)
- [Resource Types Defined by AWS Mobile Hub \(p. 1393\)](#)
- [Condition Keys for AWS Mobile Hub \(p. 1394\)](#)

Actions Defined by AWS Mobile Hub

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateProject	Create a project	Write			
CreateServiceRole	Enable AWS Mobile Hub in the account by creating the required service role	Write			
DeleteProject	Delete the specified project	Write	project* (p. 1393)		
DeleteProjectSnapshot	Delete a saved snapshot of project configuration	Write			
DeployToStage	Deploy changes to the specified stage	Write			
DescribeBundle	Describe the download bundle	Read			
ExportBundle	Export the download bundle	Read			
ExportProject	Export the project configuration	Read	project* (p. 1393)		
GenerateProjectParameters	Generate project parameters required for code generation	Write	project* (p. 1393)		
GetProject	Get project configuration and resources	Read	project* (p. 1393)		
GetProjectSnapshot	Fetch the previously exported project configuration snapshot	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ImportProject	Create a new project from the previously exported project configuration	Write			
InstallBundle	Install a bundle in the project deployments S3 bucket	Write			
ListAvailableConnectors	List the available SaaS (Software as a Service) connectors	List			
ListAvailableFeatures	List available features	List			
ListAvailableRegions	List available regions for projects	List			
ListBundles	List the available download bundles	List			
ListProjectSnapshots	List saved snapshots of project configuration	List			
ListProjects	List projects	List			
SynchronizeProject	Synchronize state of resources into project	Write	project* (p. 1393)		
UpdateProject	Update project	Write	project* (p. 1393)		
ValidateProject	Validate a mobile hub project.	Read			
VerifyServiceRole	Verify AWS Mobile Hub is enabled in the account	Read			

Resource Types Defined by AWS Mobile Hub

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 1392\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
project	arn:\${Partition}:mobilehub:\${Region}: \${Account}:project/\${ProjectId}	

Condition Keys for AWS Mobile Hub

Mobile Hub has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Amazon MQ

Amazon MQ (service prefix: `mq`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- View a list of the [API operations available for this service](#).

Topics

- [Actions Defined by Amazon MQ \(p. 1394\)](#)
- [Resource Types Defined by Amazon MQ \(p. 1396\)](#)
- [Condition Keys for Amazon MQ \(p. 1397\)](#)

Actions Defined by Amazon MQ

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateBroker	Grants permission to create a broker.	Write		aws:RequestTag/\${TagKey} (p. 1397)	
CreateConfiguration	Grants permission to create a new configuration for the specified configuration name. Amazon MQ uses the default configuration (the engine type and engine version).	Write		aws:RequestTag/\${TagKey} (p. 1397)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateTags	Grants permission to create tags.	Write	brokers (p. 1396)		
			configurations (p. 1396)		
				aws:RequestTag/ \${TagKey} (p. 1397)	aws:TagKeys (p. 1397)
CreateUser	Grants permission to create an ActiveMQ user.	Write	brokers* (p. 1396)		
DeleteBroker	Grants permission to delete a broker.	Write	brokers* (p. 1396)		
DeleteTags	Grants permission to delete tags.	Write	brokers (p. 1396)		
			configurations (p. 1396)		
				aws:TagKeys (p. 1397)	
DeleteUser	Grants permission to delete an ActiveMQ user.	Write	brokers* (p. 1396)		
DescribeBroker	Grants permission to return information about the specified broker.	Read	brokers* (p. 1396)		
DescribeBrokerEngineInformation	Grants permission to return information about broker engines.	Read			
DescribeBrokerInstanceOptions	Grants permission to return information about the broker instance options	Read			
DescribeConfigurationInformation	Grants permission to return information about the specified configuration.	Read	configurations* (p. 1396)		
DescribeConfigurationRevision	Grants permission to return the specified configuration revision for the specified configuration.	Read	configurations* (p. 1396)		
DescribeUser	Grants permission to return information about an ActiveMQ user.	Read	brokers* (p. 1396)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListBrokers	Grants permission to return a list of all brokers.	List			
ListConfigurationRevisions	Grants permission to return a list of all existing revisions for the specified configuration.	List	configurations* (p. 1396)		
ListConfigurations	Grants permission to return a list of all configurations.	List			
ListTags	Grants permission to return a list of tags.	List	brokers (p. 1396)		
			configurations (p. 1396)		
ListUsers	Grants permission to return a list of all ActiveMQ users.	List	brokers* (p. 1396)		
RebootBroker	Grants permission to reboot a broker.	Write	brokers* (p. 1396)		
UpdateBroker	Grants permission to add a pending configuration change to a broker.	Write	brokers* (p. 1396)		
UpdateConfiguration	Grants permission to update the specified configuration.	Write	configurations* (p. 1396)		
UpdateUser	Grants permission to update the information for an ActiveMQ user.	Write	brokers* (p. 1396)		

Resource Types Defined by Amazon MQ

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1394\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
brokers	<code>arn:\${Partition}:mq:\${Region}: \${Account}:broker:\${broker-id}</code>	aws:ResourceTag/\${TagKey} (p. 1397)
configurations	<code>arn:\${Partition}:mq:\${Region}: \${Account}:configuration:\${configuration-id}</code>	aws:ResourceTag/\${TagKey} (p. 1397)

Condition Keys for Amazon MQ

Amazon MQ defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
<code>aws:RequestTag/\${TagKey}</code>		String
<code>aws:ResourceTag/\${TagKey}</code>		String
<code>aws:TagKeys</code>		String

Actions, Resources, and Condition Keys for Amazon Neptune

Amazon Neptune (service prefix: `neptune-db`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Neptune \(p. 1397\)](#)
- [Resource Types Defined by Amazon Neptune \(p. 1398\)](#)
- [Condition Keys for Amazon Neptune \(p. 1398\)](#)

Actions Defined by Amazon Neptune

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
connect	Connect to database	Write	database* (p. 1398)		

Resource Types Defined by Amazon Neptune

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1397\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
database	<code>arn:\${Partition}:neptune-db:\${Region}:\${Account}: \${RelativeId}/database</code>	

Condition Keys for Amazon Neptune

Neptune has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Network Manager

Network Manager (service prefix: `networkmanager`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Network Manager \(p. 1398\)](#)
- [Resource Types Defined by Network Manager \(p. 1403\)](#)
- [Condition Keys for Network Manager \(p. 1404\)](#)

Actions Defined by Network Manager

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the Resource element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AssociateCustomerGatewayToDevice	Grants permission to associate a customer gateway to a device	Write	device* (p. 1404)		
	global-network* (p. 1403)				
	link (p. 1404)				
			networkmanager:cgwArn (p. 1404)		
AssociateLink	Grants permission to associate a link to a device	Write	device* (p. 1404)		
	global-network* (p. 1403)				
	link* (p. 1404)				
CreateDevice	Grants permission to create a new device	Write	global-network* (p. 1403)		
			aws:RequestTag/ {\$TagKey} (p. 1404)		
CreateGlobalNetwork	Grants permission to create a new global network	Write	aws:RequestTag/ {\$TagKey} (p. 1404)		
			aws:TagKeys (p. 1404)		
CreateLink	Grants permission to create a new link	Write	global-network* (p. 1403)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			site (p. 1403)		
	CreateSite Grants permission to create a new site	Write	global-network* (p. 1403)	aws:RequestTag/ {\$TagKey} (p. 1404)	aws:TagKeys (p. 1404)
				aws:RequestTag/ {\$TagKey} (p. 1404)	aws:TagKeys (p. 1404)
	DeleteDevice Grants permission to delete a device	Write	device* (p. 1404)		
			global-network* (p. 1403)		
DeleteGlobalNetwork	Grants permission to delete a global network	Write	global-network* (p. 1403)		
	DeleteLink Grants permission to delete a link	Write	global-network* (p. 1403)	link* (p. 1404)	
	DeleteSite Grants permission to delete a site	Write	global-network* (p. 1403)	site* (p. 1403)	
	DeregisterTransitGateway Grants permission to deregister a transit gateway from a global network	Write	global-network* (p. 1403)	networkmanager:tgwArn (p. 1404)	
	DescribeGlobalNetworks Grants permission to describe global networks	List	global-network (p. 1403)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions	
DisassociateCustomerGateway	Grants permission to disassociate a customer gateway from a device	Write	global-network* (p. 1403)			
				networkmanager:cgwArn (p. 1404)		
DisassociateLink	Grants permission to disassociate a link from a device	Write	device* (p. 1404)			
			global-network* (p. 1403)			
			link* (p. 1404)			
GetCustomerGatewayAssociations	Grants permission to describe customer gateway associations	List	global-network* (p. 1403)			
GetDevices	Grants permission to describe devices	List	global-network* (p. 1403)			
				device (p. 1404)		
GetLinkAssociations	Grants permission to describe link associations	List	global-network* (p. 1403)			
				device (p. 1404)		
				link (p. 1404)		
GetLinks	Grants permission to describe links	List	global-network* (p. 1403)			
				link (p. 1404)		
GetSites	Grants permission to describe global networks	List	global-network* (p. 1403)			
				site (p. 1403)		
GetTransitGatewayRegistrations	Grants permission to describe transit gateway registrations	List	global-network* (p. 1403)			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListTagsForResource	Grants permission to lists tag for a Network Manager resource	Read	device (p. 1404)		
			global-network (p. 1403)		
			link (p. 1404)		
			site (p. 1403)		
				aws:ResourceTag/\${TagKey} (p. 1404)	
RegisterTransitGateway	Grants permission to register a transit gateway to a global network	Write	global-network* (p. 1403)		
				networkmanager:tgwArn (p. 1404)	
TagResource	Grants permission to tag a Network Manager resource	Tagging	device (p. 1404)		
			global-network (p. 1403)		
			link (p. 1404)		
			site (p. 1403)		
				aws:TagKeys (p. 1404) aws:RequestTag/\${TagKey} (p. 1404) aws:ResourceTag/\${TagKey} (p. 1404)	
UntagResource	Grants permission to untag a Network Manager resource	Tagging	device (p. 1404)		
				global-network (p. 1403)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			link (p. 1404)		
			site (p. 1403)		
				aws:TagKeys (p. 1404)	
UpdateDevice	Grants permission to update a device	Write	device* (p. 1404)		
			global-network* (p. 1403)		
UpdateGlobalNetwork	Grants permission to update a global network	Write	global-network* (p. 1403)		
UpdateLink	Grants permission to update a link	Write	global-network* (p. 1403)		
			link* (p. 1404)		
UpdateSite	Grants permission to update a site	Write	global-network* (p. 1403)		
			site* (p. 1403)		

Resource Types Defined by Network Manager

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 1398\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
global-network	arn:\${Partition}:networkmanager::\${Account}:global-network/\${ResourceId}	aws:ResourceTag/\${TagKey} (p. 1404)
site	arn:\${Partition}:networkmanager::\${Account}:site/\${GlobalNetworkId}/\${ResourceId}	aws:ResourceTag/\${TagKey} (p. 1404)

Resource Types	ARN	Condition Keys
link	arn:\${Partition}:networkmanager::\${Account}:link/\${GlobalNetworkId}/ \${ResourceId}	aws:ResourceTag/\${TagKey} (p. 1404)
device	arn:\${Partition}:networkmanager::\${Account}:device/\${GlobalNetworkId}/ \${ResourceId}	aws:ResourceTag/\${TagKey} (p. 1404)

Condition Keys for Network Manager

Network Manager defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Filters actions based on the presence of tag key-value pairs in the request	String
aws:ResourceTag/\${TagKey}	Filters actions based on tag key-value pairs attached to the resource	String
aws:TagKeys	Filters actions based on the presence of tag keys in the request	String
networkmanager:cgwdisassociated	Controls which customer gateways can be associated or disassociated	String
networkmanager:tgwderegistered	Controls which transit gateways can be registered or deregistered	String

Actions, Resources, and Condition Keys for AWS OpsWorks

AWS OpsWorks (service prefix: `opsworks`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS OpsWorks \(p. 1405\)](#)
- [Resource Types Defined by AWS OpsWorks \(p. 1409\)](#)
- [Condition Keys for AWS OpsWorks \(p. 1409\)](#)

Actions Defined by AWS OpsWorks

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AssignInstance	Assign a registered instance to a layer	Write	stack (p. 1409)		
AssignVolume	Assigns one of the stack's registered Amazon EBS volumes to a specified instance	Write	stack (p. 1409)		
AssociateElasticIp	Associates one of the stack's registered Elastic IP addresses with a specified instance	Write	stack (p. 1409)		
AttachElasticLoadBalancing	Attaches an Elastic Load Balancing load balancer to a specified layer	Write	stack (p. 1409)		
CloneStack	Creates a clone of a specified stack	Write	stack (p. 1409)		
CreateApp	Creates an app for a specified stack	Write	stack (p. 1409)		
CreateDeployment	Runs deployment or stack commands	Write	stack (p. 1409)		
CreateInstance	Creates an instance in a specified stack	Write	stack (p. 1409)		
CreateLayer	Creates a layer	Write	stack (p. 1409)		
CreateStack	Creates a new stack	Write			
CreateUserProfile	Creates a new user profile	Write			
DeleteApp	Deletes a specified app	Write	stack (p. 1409)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteInstance	Deletes a specified instance, which terminates the associated Amazon EC2 instance	Write	stack (p. 1409)		
DeleteLayer	Deletes a specified layer	Write	stack (p. 1409)		
DeleteStack	Deletes a specified stack	Write	stack (p. 1409)		
DeleteUserProfile	Deletes a user profile	Write			
DeregisterEcsCluster	Deletes a user profile	Write	stack (p. 1409)		
DeregisterElasticIp	Deregisters a specified Elastic IP address	Write	stack (p. 1409)		
DeregisterInstance	Deregister a registered Amazon EC2 or on-premises instance	Write	stack (p. 1409)		
DeregisterRdsDbInstance	Deregisters an Amazon RDS instance	Write	stack (p. 1409)		
DeregisterVolume	Deregisters an Amazon EBS volume	Write	stack (p. 1409)		
DescribeAgentVersions	Describes the available AWS OpsWorks agent versions	List	stack (p. 1409)		
DescribeApps	Requests a description of a specified set of apps	List	stack (p. 1409)		
DescribeCommands	Describes the results of specified commands	List	stack (p. 1409)		
DescribeDeployments	Requests a description of a specified set of deployments	List	stack (p. 1409)		
DescribeEcsClusters	Describes Amazon ECS clusters that are registered with a stack	List	stack (p. 1409)		
DescribeElasticips	Describes Elastic IP addresses	List	stack (p. 1409)		
DescribeElasticLoadBalancingInstances	Describes a stack's Elastic Load Balancing instances	List	stack (p. 1409)		
DescribeInstances	Requests a description of a set of instances	List	stack (p. 1409)		
DescribeLayers	Requests a description of one or more layers in a specified stack	List	stack (p. 1409)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeLoadBasedScalingConfigurations	Describes load-based auto scaling configurations for specified layers	List	stack (p. 1409)		
DescribeMyUserPermissions	Describes a user's SSH information	List			
DescribePermissions	Describes the permissions for a specified stack	List	stack (p. 1409)		
DescribeRaidArrays	Describe an instance's RAID arrays	List	stack (p. 1409)		
DescribeRdsDbInstances	Describes Amazon RDS instances	List	stack (p. 1409)		
DescribeServiceErrors	Describes AWS OpsWorks service errors	List	stack (p. 1409)		
DescribeStackProvisioningParameters	Requests a description of a stack's provisioning parameters	List	stack (p. 1409)		
DescribeStackSummary	Describes the number of layers and apps in a specified stack, and the number of instances in each state, such as running_setup or online	List	stack (p. 1409)		
DescribeStacks	Requests a description of one or more stacks	List	stack (p. 1409)		
DescribeTimeBasedScalingConfigurations	Describes time-based auto scaling configurations for specified instances	List	stack (p. 1409)		
DescribeUserProfiles	Describe specified users	List			
DescribeVolumes	Describes an instance's Amazon EBS volumes	List	stack (p. 1409)		
DetachElasticLoadBalancing	Detaches a specified Elastic Load Balancing instance from its layer	Write	stack (p. 1409)		
DisassociateElasticIp	Disassociates an Elastic IP address from its instance	Write	stack (p. 1409)		
GetHostnameSuggestions	Gets a generated host name for the specified layer, based on the current host name theme	Read	stack (p. 1409)		
GrantAccess	Grants RDP access to a Windows instance for a specified time period	Write	stack (p. 1409)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListTags	Returns a list of tags that are applied to the specified stack or layer	List	stack (p. 1409)		
RebootInstance	Reboots a specified instance	Write	stack (p. 1409)		
RegisterEcsCluster	Registers a specified Amazon ECS cluster with a stack	Write	stack (p. 1409)		
RegisterElasticIp	Registers an Elastic IP address with a specified stack	Write	stack (p. 1409)		
RegisterInstance	Registers instances with a specified stack that were created outside of AWS OpsWorks	Write	stack (p. 1409)		
RegisterRdsDbInstance	Registers an Amazon RDS instance with a stack	Write	stack (p. 1409)		
RegisterVolume	Registers an Amazon EBS volume with a specified stack	Write	stack (p. 1409)		
SetLoadBasedAutoScaling	Specify the load-based auto scaling configuration for a specified layer	Write	stack (p. 1409)		
SetPermission	Specifies a user's permissions	Permissions management	stack (p. 1409)		
SetTimeBasedAutoScaling	Specify the time-based auto scaling configuration for a specified instance	Write	stack (p. 1409)		
StartInstance	Starts a specified instance	Write	stack (p. 1409)		
StartStack	Starts a stack's instances	Write	stack (p. 1409)		
StopInstance	Stops a specified instance	Write	stack (p. 1409)		
StopStack	Stops a specified stack	Write	stack (p. 1409)		
TagResource	Apply tags to a specified stack or layer	Write	stack (p. 1409)		
UnassignInstance	Unassigns a registered instance from all of its layers	Write	stack (p. 1409)		
UnassignVolume	Unassigns an assigned Amazon EBS volume	Write	stack (p. 1409)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UntagResource	Removes tags from a specified stack or layer	Write	stack (p. 1409)		
UpdateApp	Updates a specified app	Write	stack (p. 1409)		
UpdateElasticIp	Updates a registered Elastic IP address's name	Write	stack (p. 1409)		
UpdateInstance	Updates a specified instance	Write	stack (p. 1409)		
UpdateLayer	Updates a specified layer	Write	stack (p. 1409)		
UpdateMyUserProfile	Updates a user's SSH public key	Write			
UpdateRdsDbInstnace	Updates an Amazon RDS instance	Write	stack (p. 1409)		
UpdateStack	Updates a specified stack	Write	stack (p. 1409)		
UpdateUserProfile	Updates a specified user profile	Permissions management			
UpdateVolume	Updates an Amazon EBS volume's name or mount point	Write	stack (p. 1409)		

Resource Types Defined by AWS OpsWorks

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1405\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
stack	<code>arn:\${Partition}:opsworks:\${Region}:\${Account}:stack/\${StackId}/</code>	

Condition Keys for AWS OpsWorks

OpsWorks has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for AWS OpsWorks Configuration Management

AWS OpsWorks Configuration Management (service prefix: `opsworks-cm`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS OpsWorks Configuration Management \(p. 1410\)](#)
- [Resource Types Defined by AWS OpsWorks Configuration Management \(p. 1411\)](#)
- [Condition Keys for AWS OpsWorks Configuration Management \(p. 1411\)](#)

Actions Defined by AWS OpsWorks Configuration Management

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AssociateNode	Associate a node to a configuration management server.	Write			
CreateBackup	Create a backup for the specified server.	Write			
CreateServer	Create a new server.	Write			
DeleteBackup	Delete the specified backup and possibly its S3 bucket.	Write			
DeleteServer	Deletes the specified server with his corresponding CF stack and possibly the S3 bucket.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeAccount	Describe the service limits for the user's account.	List			
DescribeBackups	Describe a single backup, all backups of a specified server or all backups of the user's account.	List			
DescribeEvents	Describe all events of the specified server.	List			
DescribeNodeAssociations	Describe the association status for the specified node token and the specified server.	List			
DescribeServers	Describes the specified server or all servers of the user's account.	List			
DisassociateNode	Disassociates a specified node from a server.	Write			
RestoreServer	Applies a backup to specified server. Possibly swaps out the ec2-instance if specified.	Write			
StartMaintenance	Start the server maintenance immediately.	Write			
UpdateServer	Update general server settings.	Write			
UpdateServerEngines	Update server settings specific to the configuration management type.	Write			

Resource Types Defined by AWS OpsWorks Configuration Management

AWS OpsWorks Configuration Management does not support specifying a resource ARN in the Resource element of an IAM policy statement. To allow access to AWS OpsWorks Configuration Management, specify "Resource": "*" in your policy.

Condition Keys for AWS OpsWorks Configuration Management

OpsworksCM has no service-specific context keys that can be used in the Condition element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for AWS Organizations

AWS Organizations (service prefix: organizations) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).

- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Organizations \(p. 1412\)](#)
- [Resource Types Defined by AWS Organizations \(p. 1416\)](#)
- [Condition Keys for AWS Organizations \(p. 1417\)](#)

Actions Defined by AWS Organizations

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AcceptHandshake	Grants permission to send a response to the originator of a handshake agreeing to the action proposed by the handshake request.	Write	handshake* (p. 1417)		
AttachPolicy	Grants permission to attach a policy to a root, an organizational unit, or an individual account.	Write	policy* (p. 1417)		
			account (p. 1417)		
			organizationalunit (p. 1417)		
			root (p. 1417)		
CancelHandshake	Grants permission to cancel a handshake.	Write	handshake* (p. 1417)		
CreateAccount	Grants permission to create an AWS account that is automatically a member of the organization with the credentials that made the request.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateGovCloud	Grants permission to create an AWS GovCloud (US) account.	Write			
CreateOrganization	Grants permission to create an organization. The account with the credentials that calls the CreateOrganization operation automatically becomes the master account of the new organization.	Write			
CreateOrganizationalUnit	Grants permission to create an organizational unit (OU) within a root or parent OU.	Write	organizationalunit (p. 1417)		
			root (p. 1417)		
CreatePolicy	Grants permission to create a policy that you can attach to a root, an organizational unit (OU), or an individual AWS account.	Write			
DeclineHandshake	Grants permission to decline a handshake request. This sets the handshake state to DECLINED and effectively deactivates the request.	Write	handshake* (p. 1417)		
DeleteOrganization	Grants permission to delete the organization.	Write			
DeleteOrganizationalUnit	Grants permission to delete an organizational unit from a root or another OU.	Write	organizationalunit* (p. 1417)		
DeletePolicy	Grants permission to delete a policy from your organization.	Write	policy* (p. 1417)		
DescribeAccount	Grants permission to retrieve Organizations-related details about the specified account.	Read	account* (p. 1417)		
DescribeCreateAccountStatus	Grants permission to retrieve the current status of an asynchronous request to create an account.	Read			
DescribeEffectivePolicy	Grants permission to retrieve the effective policy for an account.	Read	account* (p. 1417)		
DescribeHandshake	Grants permission to retrieve details about a previously requested handshake.	Read	handshake* (p. 1417)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeOrganization	Grants permission to retrieves details about the organization that the calling credentials belong to.	Read			
DescribeOrganizationalUnit	Grants permission to retrieve details about an organizational unit (OU).	Read	organizationalunit* (p. 1417)		
DescribePolicy	Grants permission to retrieves details about a policy.	Read	policy* (p. 1417)		
DetachPolicy	Grants permission to detach a policy from a target root, organizational unit, or account.	Write	policy* (p. 1417)		
			account (p. 1417)		
			organizationalunit (p. 1417)		
			root (p. 1417)		
DisableAWSServiceIntegration	Grants permission to disable integration of an AWS service (the service that is specified by ServicePrincipal) with AWS Organizations.	Write		organizations:ServicePrincipal (p. 1417)	
DisablePolicyType	Grants permission to disable an organization policy type in a root.	Write	root* (p. 1417)		
EnableAWSServiceIntegration	Grants permission to enable integration of an AWS service (the service that is specified by ServicePrincipal) with AWS Organizations.	Write		organizations:ServicePrincipal (p. 1417)	
EnableAllFeatures	Grants permission to start the process to enable all features in an organization, upgrading it from supporting only Consolidated Billing features.	Write			
EnablePolicyType	Grants permission to enable a policy type in a root.	Write	root* (p. 1417)		
InviteAccountToOrganization	Grants permission to send invitation to another AWS account, asking it to join your organization as a member account.	Write	account (p. 1417)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
LeaveOrganization	Grants permission to remove a member account from its parent organization.	Write			
ListAWSServiceActions	Grants permission to retrieve the list of the AWS services for which you enabled integration with your organization.	List			
ListAccounts	Grants permission to list all of the the accounts in the organization.	List			
ListAccountsForParent	Grants permission to list the accounts in an organization that are contained by a root or organizational unit (OU).	List	organizationalunit (p. 1417)		
			root (p. 1417)		
ListChildren	Grants permission to list all of the OUs or accounts that are contained in a parent OU or root.	List	organizationalunit (p. 1417)		
			root (p. 1417)		
ListCreateAccountTasks	Grants permission to list the synchronous account creation requests that are currently being tracked for the organization.	List			
ListHandshakesForAccount	Grants permission to list all handshakes that are associated with an account.	List			
ListHandshakesForOrganization	Grants permission to list the handshakes that are associated with the organization.	List			
ListOrganizationUnits	Grants permission to lists all of the organizational units (OUs) in a parent organizational unit or root.	List	organizationalunit (p. 1417)		
			root (p. 1417)		
ListParents	Grants permission to list the root or organizational units (OUs) that serve as the immediate parent of a child OU or account.	List	account (p. 1417)		
			organizationalunit (p. 1417)		
ListPolicies	Grants permission to list all of the policies in an organization.	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListPoliciesForTargets	Grants permission to list all of the policies that are directly attached to a root, organizational unit (OU), or account.	List	account (p. 1417)		
	organizationalunit (p. 1417)				
	root (p. 1417)				
ListRoots	Grants permission to list all of the roots that are defined in the organization.	List			
ListTagsForResource	Grants permission to list all tags for the specified resource.	List			
ListTargetsForPolicy	Grants permission to list all the roots, OUs, and accounts to which a policy is attached.	List	policy* (p. 1417)		
MoveAccount	Grants permission to move an account from its current root or OU to another parent root or OU.	Write	account* (p. 1417)		
organizationalunit (p. 1417)					
root (p. 1417)					
RemoveAccountFromOrganizationalUnit	Grants permission to remove the specified account from the organization.	Write	account* (p. 1417)		
TagResource	Grants permission to add one or more tags to the specified resource.	Tagging			
UntagResource	Grants permission to remove one or more tags from the specified resource.	Tagging			
UpdateOrganizationalUnit	Grants permission to rename an organizational unit (OU).	Write	organizationalunit* (p. 1417)		
UpdatePolicy	Grants permission to update an existing policy with a new name, description, or content.	Write	policy* (p. 1417)		

Resource Types Defined by AWS Organizations

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 1412\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
account	arn:\${Partition}:organizations::\${MasterAccountId}:account/o-\${OrganizationId}/\${AccountId}	
handshake	arn:\${Partition}:organizations::\${MasterAccountId}:handshake/o-\${OrganizationId}/\${HandshakeType}/h-\${HandshakeId}	
organization	arn:\${Partition}:organizations::\${MasterAccountId}:organization/o-\${OrganizationId}	
organizationalunit	arn:\${Partition}:organizations::\${MasterAccountId}:ou/o-\${OrganizationId}/ou-\${OrganizationalUnitId}	
policy	arn:\${Partition}:organizations::\${MasterAccountId}:policy/o-\${OrganizationId}/\${PolicyType}/p-\${PolicyId}	
awspolicy	arn:\${Partition}:organizations::aws:policy/\${PolicyType}/p-\${PolicyId}	
root	arn:\${Partition}:organizations::\${MasterAccountId}:root/o-\${OrganizationId}/r-\${RootId}	

Condition Keys for AWS Organizations

AWS Organizations defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
organizations:ServicePrincipal	Enables you to filter the request to only the specified service principal names.	String

Actions, Resources, and Condition Keys for AWS Outposts

AWS Outposts (service prefix: `outposts`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).

- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Outposts \(p. 1418\)](#)
- [Resource Types Defined by AWS Outposts \(p. 1418\)](#)
- [Condition Keys for AWS Outposts \(p. 1419\)](#)

Actions Defined by AWS Outposts

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateOutpost	Creates an Outpost	Write			
GetOutpost	Gets information about the specified Outpost	Read			
GetOutpostInstances	Lists the instance types for the specified Outpost	Read			
ListOutposts	List the Outposts for your AWS account	List			
ListSites	Lists the sites for the specified AWS account	List			

Resource Types Defined by AWS Outposts

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1418\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
Outpost	arn:\${Partition}:outposts:\${Region}: \${Account}:outpost/\${OutpostId}	
Site	arn:\${Partition}:outposts:\${Region}: \${Account}:site/\${SiteId}	
Order	arn:\${Partition}:outposts:\${Region}: \${Account}:order/\${OrderId}	

Condition Keys for AWS Outposts

Outposts has no service-specific context keys that can be used in the Condition element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for AWS Performance Insights

AWS Performance Insights (service prefix: `pi`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

Topics

- [Actions Defined by AWS Performance Insights \(p. 1419\)](#)
- [Resource Types Defined by AWS Performance Insights \(p. 1420\)](#)
- [Condition Keys for AWS Performance Insights \(p. 1420\)](#)

Actions Defined by AWS Performance Insights

You can specify the following actions in the Action element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the Resource element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeDimensions	For a specific time period, retrieve the top N dimension keys for a metric.	Read	metric-resource* (p. 1420)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetResourceMetrics	Retrieve PI metrics for a set of data sources, over a time period.	Read	metric-resource* (p. 1420)		

Resource Types Defined by AWS Performance Insights

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1419\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
metric-resource	arn:\${Partition}:pi:\${Region}: \${Account}:metrics/\${ServiceType}/ \${Identifier}	

Condition Keys for AWS Performance Insights

Performance Insights has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Amazon Personalize

Amazon Personalize (service prefix: `personalize`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).

Topics

- [Actions Defined by Amazon Personalize \(p. 1420\)](#)
- [Resource Types Defined by Amazon Personalize \(p. 1423\)](#)
- [Condition Keys for Amazon Personalize \(p. 1424\)](#)

Actions Defined by Amazon Personalize

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateCampaign	Creates a campaign	Write	campaign* (p. 1423)		
CreateDataset	Creates a dataset	Write	dataset* (p. 1423)		
CreateDatasetGroup	Creates a dataset group	Write	datasetGroup* (p. 1423)		
CreateDatasetImportJob	Creates a dataset import job	Write	datasetImportJob* (p. 1423)		
CreateEventTracker	Creates an event tracker	Write	eventTracker* (p. 1423)		
CreateSchema	Creates a schema	Write	schema* (p. 1423)		
CreateSolution	Creates a solution	Write	solution* (p. 1423)		
CreateSolutionVersion	Creates a solution version	Write	solution* (p. 1423)		
DeleteCampaign	Deletes a campaign	Write	campaign* (p. 1423)		
DeleteDataset	Deletes a dataset	Write	dataset* (p. 1423)		
DeleteDatasetGroup	Deletes a dataset group	Write	datasetGroup* (p. 1423)		
DeleteEventTracker	Deletes an event tracker	Write	eventTracker* (p. 1423)		
DeleteSchema	Deletes a schema	Write	schema* (p. 1423)		
DeleteSolution	Deletes a solution including all versions of the solution	Write	solution* (p. 1423)		
DescribeAlgorithm	Describes an algorithm	Read	algorithm* (p. 1423)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeCampaign	Describes a campaign	Read	campaign* (p. 1423)		
DescribeDataset	Describes a dataset	Read	dataset* (p. 1423)		
DescribeDatasetGroup	Describes a dataset group	Read	datasetGroup* (p. 1423)		
DescribeDatasetImportJob	Describes a dataset import job	Read	datasetImportJob* (p. 1423)		
DescribeEventTracker	Describes an event tracker	Read	eventTracker* (p. 1423)		
DescribeFeatureTransformation	Describes a feature transformation	Read	featureTransformation* (p. 1423)		
DescribeRecipe	Describes a recipe	Read	recipe* (p. 1423)		
DescribeSchema	Describes a schema	Read	schema* (p. 1423)		
DescribeSolution	Describes a solution	Read	solution* (p. 1423)		
DescribeSolutionVersion	Describes a version of a solution	Read	solution* (p. 1423)		
GetPersonalizedRecommendations	Gets a re-ranked list of recommendations	Write	campaign* (p. 1423)		
GetRecommendations	Gets a list of recommendations from a campaign	Read	campaign* (p. 1423)		
GetSolutionMetrics	Gets metrics for a solution version	Read	solution* (p. 1423)		
ListCampaigns	Lists campaigns	List			
ListDatasetGroups	Lists dataset groups	List			
ListDatasetImportJobs	Lists dataset import jobs	List			
ListDatasets	Lists datasets	List			
ListEventTrackers	Lists event trackers	List			
ListRecipes	Lists recipes	List			
ListSchemas	Lists schemas	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListSolutionVersions	Lists versions of a solution	List			
ListSolutions	Lists solutions	List			
PutEvents	Records real time event data	Write	eventTracker* (p. 1423)		
UpdateCampaign	Updates a campaign	Write	campaign* (p. 1423)		

Resource Types Defined by Amazon Personalize

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1420\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
schema	<code>arn:\${Partition}:personalize:\${Region}: \${Account}:schema/\${ResourceId}</code>	
featureTransformation	<code>arn:\${Partition}:personalize:\${Region}: \${Account}:feature-transformation/ \${ResourceId}</code>	
dataset	<code>arn:\${Partition}:personalize:\${Region}: \${Account}:dataset/\${ResourceId}</code>	
datasetGroup	<code>arn:\${Partition}:personalize:\${Region}: \${Account}:dataset-group/\${ResourceId}</code>	
datasetImportJob	<code>arn:\${Partition}:personalize:\${Region}: \${Account}:dataset-import-job/\${ResourceId}</code>	
solution	<code>arn:\${Partition}:personalize:\${Region}: \${Account}:solution/\${ResourceId}</code>	
campaign	<code>arn:\${Partition}:personalize:\${Region}: \${Account}:campaign/\${ResourceId}</code>	
eventTracker	<code>arn:\${Partition}:personalize:\${Region}: \${Account}:event-tracker/\${ResourceId}</code>	
recipe	<code>arn:\${Partition}:personalize:\${Region}: \${Account}:recipe/\${ResourceId}</code>	
algorithm	<code>arn:\${Partition}:personalize:\${Region}: \${Account}:algorithm/\${ResourceId}</code>	

Condition Keys for Amazon Personalize

Personalize has no service-specific context keys that can be used in the Condition element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Amazon Pinpoint

Amazon Pinpoint (service prefix: `mobiletargeting`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Pinpoint \(p. 1424\)](#)
- [Resource Types Defined by Amazon Pinpoint \(p. 1434\)](#)
- [Condition Keys for Amazon Pinpoint \(p. 1435\)](#)

Actions Defined by Amazon Pinpoint

You can specify the following actions in the Action element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the Resource element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateApp	Create an app.	Write		aws:RequestTag/\${TagKey} (p. 1435) aws:TagKeys (p. 1435) aws:ResourceTag/\${TagKey} (p. 1435)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateCampaign	Create a campaign for an app.	Write	apps* (p. 1435)		
				aws:RequestTag/ {\$TagKey} (p. 1435) aws:TagKeys (p. 1435) aws:ResourceTag/ {\$TagKey} (p. 1435)	
CreateEmailTemplate	Create an email template.	Write		aws:RequestTag/ {\$TagKey} (p. 1435) aws:TagKeys (p. 1435) aws:ResourceTag/ {\$TagKey} (p. 1435)	
CreateExportJob	Create an export job that exports endpoint definitions to Amazon S3.	Write	apps* (p. 1435)		
CreateImportJob	Import endpoint definitions from to create a segment.	Write	apps* (p. 1435)		
CreateJourney	Create a Journey for an app.	Write	apps* (p. 1435)		
				aws:RequestTag/ {\$TagKey} (p. 1435) aws:TagKeys (p. 1435) aws:ResourceTag/ {\$TagKey} (p. 1435)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreatePushTemplate	Create a push notification template.	Write		aws:RequestTag/\${TagKey} (p. 1435) aws:TagKeys (p. 1435) aws:ResourceTag/\${TagKey} (p. 1435)	
CreateSegment	Create a segment that is based on endpoint data reported to Pinpoint by your app. To allow a user to create a segment by importing endpoint data from outside of Pinpoint, allow the mobiletargeting:CreateImportJob action.	Write	apps* (p. 1435)	aws:RequestTag/\${TagKey} (p. 1435) aws:TagKeys (p. 1435) aws:ResourceTag/\${TagKey} (p. 1435)	
CreateSmsTemplate	Create an sms message template.	Write		aws:RequestTag/\${TagKey} (p. 1435) aws:TagKeys (p. 1435) aws:ResourceTag/\${TagKey} (p. 1435)	
CreateVoiceTemplate	Create a voice message template.	Write		aws:RequestTag/\${TagKey} (p. 1435) aws:TagKeys (p. 1435) aws:ResourceTag/\${TagKey} (p. 1435)	
DeleteAdmChannel	Delete the ADM channel for an app.	Write	apps* (p. 1435)		
DeleteApnsChannel	Delete the APNs channel for an app.	Write	apps* (p. 1435)		
DeleteApnsSandboxChannel	Delete the APNs sandbox channel for an app.	Write	apps* (p. 1435)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteApnsVoipChannelForApp	Delete the APNs VoIP channel for an app.	Write	apps* (p. 1435)		
DeleteApnsVoipSandboxChannelForApp	Delete the APNs VoIP sandbox channel for an app.	Write	apps* (p. 1435)		
DeleteApp	Delete a specific campaign.	Write	apps* (p. 1435)		
DeleteBaiduChannelForApp	Delete the Baidu channel for an app.	Write	apps* (p. 1435)		
DeleteCampaign	Delete a specific campaign.	Write	apps* (p. 1435)		
			campaigns* (p. 1435)		
DeleteEmailChannelForApp	Delete the email channel for an app.	Write	apps* (p. 1435)		
DeleteEmailTemplate	Delete an email template or an email template version.	Write	templates* (p. 1435)		
DeleteEndpoint	Delete an endpoint.	Write	apps* (p. 1435)		
DeleteEventStreamForApp	Delete the event stream for an app.	Write	apps* (p. 1435)		
DeleteGcmChannelForApp	Delete the GCM channel for an app.	Write	apps* (p. 1435)		
DeleteJourney	Delete a specific journey.	Write	apps* (p. 1435)		
			journeys* (p. 1435)		
DeletePushTemplate	Delete a push notification template or a push notification template version.	Write	templates* (p. 1435)		
DeleteSegment	Delete a specific segment.	Write	apps* (p. 1435)		
			segments* (p. 1435)		
DeleteSmsChannelForApp	Delete the SMS channel for an app.	Write	apps* (p. 1435)		
DeleteSmsTemplate	Delete an sms message template or an sms message template version.	Write	templates* (p. 1435)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteUserEndpoint	Delete all of the endpoints that are associated with a user ID.	Write	apps* (p. 1435)		
DeleteVoiceChannel	Delete the Voice channel for an app.	Write	apps* (p. 1435)		
DeleteVoiceTemplate	Delete a voice message template or a voice message template version.	Write	templates* (p. 1435)		
GetAdmChannel	Retrieve information about the Amazon Device Messaging (ADM) channel for an app.	Read	apps* (p. 1435)		
GetApnsChannel	Retrieve information about the APNs channel for an app.	Read	apps* (p. 1435)		
GetApnsSandbox	Retrieve information about the APNs sandbox channel for an app.	Read	apps* (p. 1435)		
GetApnsVoipChannel	Retrieve information about the APNs VoIP channel for an app.	Read	apps* (p. 1435)		
GetApnsVoipSandbox	Retrieve information about the APNs VoIP sandbox channel for an app.	Read	apps* (p. 1435)		
GetApp	Retrieve information about a specific app in your Amazon Pinpoint account.	Read	apps* (p. 1435)		
GetApplicationSettings	Retrieve the default settings for an app.	List	apps* (p. 1435)		
GetApps	Retrieve a list of apps in your Amazon Pinpoint account.	List	apps* (p. 1435)		
GetBaiduChannel	Retrieve information about the Baidu channel for an app.	Read	apps* (p. 1435)		
GetCampaign	Retrieve information about a specific campaign.	Read	apps* (p. 1435)		
			campaigns* (p. 1435)		
GetCampaignActivities	Retrieve information about the activities performed by a campaign.	List	apps* (p. 1435)		
			campaigns* (p. 1435)		
GetCampaignVersion	Retrieve information about a specific campaign version.	Read	apps* (p. 1435)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			campaigns* (p. 1435)		
GetCampaignVersion	Retrieve information about the current and prior versions of a campaign.	List	apps* (p. 1435)		
			campaigns* (p. 1435)		
GetCampaigns	Retrieve information about all campaigns for an app.	List	apps* (p. 1435)		
GetChannels	Get all channels information for your app.	List	apps* (p. 1435)		
GetEmailChannel	Obtain information about the email channel in an app.	Read	apps* (p. 1435)		
GetEmailTemplate	Retrieve information about a specific or the active version of an email template.	Read	templates* (p. 1435)		
GetEndpoint	Retrieve information about a specific endpoint.	Read	apps* (p. 1435)		
GetEventStream	Retrieve information about the event stream for an app.	Read	apps* (p. 1435)		
GetExportJob	Obtain information about a specific export job.	Read	apps* (p. 1435)		
GetExportJobs	Retrieve a list of all of the export jobs for an app.	List	apps* (p. 1435)		
GetGcmChannel	Retrieve information about the GCM channel for an app.	Read	apps* (p. 1435)		
GetImportJob	Retrieve information about a specific import job.	Read	apps* (p. 1435)		
GetImportJobs	Retrieve information about all import jobs for an app.	List	apps* (p. 1435)		
GetJourney	Retrieve information about a specific journey.	Read	apps* (p. 1435)		
			journeys* (p. 1435)		
GetPushTemplate	Retrieve information about a specific or the active version of an push notification template.	Read	templates* (p. 1435)		
GetSegment	Retrieve information about a specific segment.	Read	apps* (p. 1435)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			segments* (p. 1435)		
GetSegmentExport	Retrieve information about jobs that export endpoint definitions from segments to Amazon S3.	List	apps*	(p. 1435)	
			segments*	(p. 1435)	
GetSegmentImport	Retrieve information about jobs that create segments by importing endpoint definitions from .	List	apps*	(p. 1435)	
			segments*	(p. 1435)	
GetSegmentVersion	Retrieve information about a specific segment version.	Read	apps*	(p. 1435)	
			segments*	(p. 1435)	
GetSegmentVersions	Retrieve information about the current and prior versions of a segment.	List	apps*	(p. 1435)	
			segments*	(p. 1435)	
GetSegments	Retrieve information about the segments for an app.	List	apps*	(p. 1435)	
GetSmsChannel	Obtain information about the SMS channel in an app.	Read	apps*	(p. 1435)	
GetSmsTemplate	Retrieve information about a specific or the active version of an sms message template.	Read	templates*	(p. 1435)	
GetUserEndpoints	Retrieve information about the endpoints that are associated with a user ID.	Read	apps*	(p. 1435)	
GetVoiceChannel	Obtain information about the Voice channel in an app.	Read	apps*	(p. 1435)	
GetVoiceTemplate	Retrieve information about a specific or the active version of a voice message template.	Read	templates*	(p. 1435)	
ListJourneys	Retrieve information about all journeys for an app.	List	apps*	(p. 1435)	
ListTagsForResource	List tags for a resource.	List	apps	(p. 1435)	
			campaigns	(p. 1435)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			segments (p. 1435)		
ListTemplateVersions	Retrieve all versions about a specific template.	List	templates* (p. 1435)		
ListTemplates	Retrieve metadata about the queried templates.	List	templates* (p. 1435)		
PhoneNumberValidation	Obtain metadata for a phone number, such as the number type (mobile, landline, or VoIP), location, and provider.	Read	apps* (p. 1435)		
PutEventStream	Create or update an event stream for an app.	Write	apps* (p. 1435)		
PutEvents	Create or update events for an app.	Write	apps* (p. 1435)		
RemoveAttributes	Used to remove the attributes for an app.	Write	apps* (p. 1435)		
SendMessages	Send an SMS message or push notification to specific endpoints.	Write	apps* (p. 1435)		
SendUsersMessages	Send an SMS message or push notification to all endpoints that are associated with a specific user ID.	Write	apps* (p. 1435)		
TagResource	Adds tags to a resource.	Tagging	apps (p. 1435)		
			campaigns (p. 1435)		
			segments (p. 1435)		
			aws:RequestTag/ \${TagKey} (p. 1435)		
UntagResource	Removes tags from a resource.	Tagging	aws:TagKeys (p. 1435)		
			apps (p. 1435)		
			campaigns (p. 1435)		
			segments (p. 1435)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:RequestTag/ \${TagKey} (p. 1435) aws:TagKeys (p. 1435)	
UpdateAdmChannel	Update the Amazon Device Messaging (ADM) channel for an app.	Write	apps* (p. 1435)		
UpdateApnsChannel	Update the Apple Push Notification service (APNs) channel for an app.	Write	apps* (p. 1435)		
UpdateApnsSandbox	Update the Apple Push Notification service (APNs) sandbox channel for an app.	Write	apps* (p. 1435)		
UpdateApnsVoip	Update the Apple Push Notification service (APNs) VoIP channel for an app.	Write	apps* (p. 1435)		
UpdateApnsVoipSandbox	Update the Apple Push Notification service (APNs) VoIP sandbox channel for an app.	Write	apps* (p. 1435)		
UpdateApplicationSettings	Update the default settings for an app.	Write	apps* (p. 1435)		
UpdateBaiduChannel	Update the Baidu channel for an app.	Write	apps* (p. 1435)		
UpdateCampaign	Update a specific campaign.	Write	apps* (p. 1435)		
			campaigns* (p. 1435)		
				aws:RequestTag/ \${TagKey} (p. 1435) aws:TagKeys (p. 1435)	
UpdateEmailChannel	Update the email channel for an app.	Write	apps* (p. 1435)		
UpdateEmailTemplate	Update a specific email template under the same version or generate a new version.	Write	templates* (p. 1435)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions	
				aws:RequestTag/ \${TagKey} (p. 1435) aws:TagKeys (p. 1435)		
UpdateEndpoint	Create an endpoint or update the information for an endpoint.	Write	apps* (p. 1435)			
UpdateEndpointsBatch	Create or update endpoints as a Batch operation.	Write	apps* (p. 1435)			
UpdateGcmChannel	Update the Firebase Cloud Messaging (FCM) or Google Cloud Messaging (GCM) API key that allows to send push notifications to your Android app.	Write	apps* (p. 1435)			
UpdateJourney	Update a specific journey.	Write	apps* (p. 1435)			
				journeys* (p. 1435)		
					aws:RequestTag/ \${TagKey} (p. 1435)	
UpdateJourneyState	Update a specific journey state.	Write	apps* (p. 1435)			
			journeys* (p. 1435)			
				aws:RequestTag/ \${TagKey} (p. 1435)		
UpdatePushTemplate	Update a specific push notification template under the same version or generate a new version.	Write	templates* (p. 1435)			
				aws:RequestTag/ \${TagKey} (p. 1435)		
				aws:TagKeys (p. 1435)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UpdateSegment	Update a specific segment.	Write	apps* (p. 1435)		
			segments* (p. 1435)		
				aws:RequestTag/ \${TagKey} (p. 1435)	
UpdateSmsChannel	Update the SMS channel for an app.	Write	aws:TagKeys (p. 1435)		
			apps* (p. 1435)		
UpdateSmsTemplate	Update a specific sms message template under the same version or generate a new version.	Write	templates* (p. 1435)		
				aws:RequestTag/ \${TagKey} (p. 1435)	
UpdateTemplateParameter	Update the active version parameter of a specific template.	Write	aws:TagKeys (p. 1435)		
			templates* (p. 1435)		
UpdateVoiceChannel	Update the Voice channel for an app.	Write	apps* (p. 1435)		
UpdateVoiceTemplate	Update a specific voice message template under the same version or generate a new version.	Write	templates* (p. 1435)		
				aws:RequestTag/ \${TagKey} (p. 1435)	
				aws:TagKeys (p. 1435)	

Resource Types Defined by Amazon Pinpoint

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 1424\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
apps	arn:\${Partition}:mobiletargeting:\${Region}: \${Account}:apps/\${AppId}	aws:ResourceTag/\${TagKey} (p. 1435)
campaigns	arn:\${Partition}:mobiletargeting:\${Region}: \${Account}:apps/\${AppId}/campaigns/ \${CampaignId}	aws:ResourceTag/\${TagKey} (p. 1435)
journeys	arn:\${Partition}:mobiletargeting:\${Region}: \${Account}:apps/\${AppId}/journeys/ \${JourneyId}	aws:ResourceTag/\${TagKey} (p. 1435)
segments	arn:\${Partition}:mobiletargeting:\${Region}: \${Account}:apps/\${AppId}/segments/ \${SegmentId}	aws:ResourceTag/\${TagKey} (p. 1435)
templates	arn:\${Partition}:mobiletargeting:\${Region}: \${Account}:templates/\${TemplateName}/ \${ChannelType}	aws:ResourceTag/\${TagKey} (p. 1435)

Condition Keys for Amazon Pinpoint

Amazon Pinpoint defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Filters access by a key that is present in the request the user makes to the pinpoint service.	String
aws:ResourceTag/\${TagKey}	Filters access by a tag key and value pair.	String
aws:TagKeys	Filters access by the list of all the tag key names present in the request the user makes to the pinpoint service.	String

Actions, Resources, and Condition Keys for Amazon Pinpoint Email Service

Amazon Pinpoint Email Service (service prefix: ses) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- View a list of the [API operations available for this service](#).

Topics

- [Actions Defined by Amazon Pinpoint Email Service \(p. 1436\)](#)
- [Resource Types Defined by Amazon Pinpoint Email Service \(p. 1441\)](#)
- [Condition Keys for Amazon Pinpoint Email Service \(p. 1441\)](#)

Actions Defined by Amazon Pinpoint Email Service

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateConfigurationSet	Create a configuration set. Configuration sets are groups of rules that you can apply to the emails you send using Amazon Pinpoint	Write	configuration-set* (p. 1441)		
				aws:TagKeys (p. 1441) aws:RequestTag/ {\$TagKey} (p. 1441)	
CreateConfigurationSetEventDestination	Create an event destination	Write	configuration-set* (p. 1441)		
CreateDedicatedIpAddresses	Create a new pool of dedicated IP addresses	Write	dedicated-ip-pool* (p. 1441)		
				aws:TagKeys (p. 1441) aws:RequestTag/ {\$TagKey} (p. 1441)	
CreateDeliverabilityPlaceholderTest	Create a new predictive inbox placement test.	Write	identity* (p. 1441)		
				aws:TagKeys (p. 1441)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:RequestTag/\${TagKey} (p. 1441)	
CreateEmailIdentity	Verifies an email identity for use with Amazon Pinpoint	Write	identity* (p. 1441)		
				aws:TagKeys (p. 1441) aws:RequestTag/\${TagKey} (p. 1441)	
DeleteConfigurationSet	Delete an existing configuration set	Write	configuration-set* (p. 1441)		
DeleteConfigurationSetEventDestination	Delete an event destination	Write	configuration-set* (p. 1441)		
DeleteDedicatedIpPool	Delete a dedicated IP pool	Write	dedicated-ip-pool* (p. 1441)		
DeleteEmailIdentity	Deletes an email identity that you previously verified for use with Amazon Pinpoint	Write	identity* (p. 1441)		
GetAccount	Obtain information about the email-sending status and capabilities	Read			
GetBlacklistReport	Retrieve a list of the blacklists that your dedicated IP addresses appear on	Read			
GetConfigurationSet	Get information about an existing configuration set	Read	configuration-set* (p. 1441)		
GetConfigurationSetDestinations	Retrieve a list of event destinations that are associated with a configuration set	Read	configuration-set* (p. 1441)		
GetDedicatedIp	Get information about a dedicated IP address	Read			
GetDedicatedIps	List the dedicated IP addresses that are associated with your Amazon Pinpoint account	Read	dedicated-ip-pool* (p. 1441)		
GetDeliverabilityDashboard	Show the status of the Deliverability dashboard	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetDeliverabilityTestReport	Retrieve the results of a predictive inbox placement test	Read	deliverability-test-report* (p. 1441)		
GetDomainStatistics	Retrieve inbox placement and engagement rates for the domains that you use to send email	Read	identity* (p. 1441)		
GetEmailIdentity	Provides information about a specific identity associated with your Amazon Pinpoint account	Read	identity* (p. 1441)		
ListConfigurationSets	List all of the configuration sets associated with your Amazon Pinpoint account in the current region	List			
ListDedicatedIpPools	List all of the dedicated IP pools that exist in your Amazon Pinpoint account in the current AWS Region	List			
ListDeliverabilityTestReports	Show a list of the predictive inbox placement tests that you've performed, regardless of their statuses	List			
ListEmailIdentities	Returns a list of all of the email identities that are associated with your Amazon Pinpoint account	List			
ListTagsForResource	Retrieve a list of the tags (keys and values) that are associated with a specific resource.	Read	configuration-set (p. 1441)		
PutAccountDedicatedIpPoolUsageMetrics	Enable or disable the automatic暖up feature for dedicated IP addresses	Write			
PutAccountSendingWriters	Enable or disable the ability of your accounts to send email	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
PutConfigurationSetWithDedicatedIpPool	Associate a configuration set with a dedicated IP pool	Write	configuration-set* (p. 1441)		
PutConfigurationSetReputationMetricsForEmails	Enable or disable collection of reputation metrics for emails that you send using a particular configuration set in a specific AWS Region	Write	configuration-set* (p. 1441)		
PutConfigurationSetMessagesThatUseAConfigurationSet	Enable or disable email sending for messages that use a particular configuration set in a specific AWS Region	Write	configuration-set* (p. 1441)		
PutConfigurationSetForLandingPageAndClickTracking	Specify a custom domain to use for landing pages and click tracking elements in email that you send using Amazon Pinpoint	Write	configuration-set* (p. 1441)		
PutDedicatedIpInExistingDedicatedIpPool	Move a dedicated IP address to an existing dedicated IP pool	Write	dedicated-ip-pool* (p. 1441)		
PutDedicatedIpWarmUpAttributes	Put Dedicated IP warm up attributes	Write			
PutDeliverabilityDashboardAttributes	Enable or disable the Deliverability dashboard	Write			
PutEmailIdentityDkimAuthentication	Used to enable or disable DKIM authentication for an email identity	Write	identity* (p. 1441)		
PutEmailIdentityFeedbackForwarding	Used to enable or disable feedback forwarding for an identity	Write	identity* (p. 1441)		
PutEmailIdentityMailFromDomain	Used to enable or disable the Mail From domain configuration for an email identity	Write	identity* (p. 1441)		
SendEmail	Sends an email message	Write	identity* (p. 1441)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				ses:FeedbackAddress (p. 1441)	
TagResource	Add one or more tags (keys and values) to a specified resource.	Tagging	configuration-set (p. 1441)		
			dedicated-ip-pool (p. 1441)		
			deliverability-test-report (p. 1441)		
			identity (p. 1441)		
				aws:TagKeys (p. 1441)	aws:RequestTag/\${TagKey} (p. 1441)
UntagResource	Remove one or more tags (keys and values) from a specified resource.	Tagging	configuration-set (p. 1441)		
			dedicated-ip-pool (p. 1441)		
			deliverability-test-report (p. 1441)		
			identity (p. 1441)		
				aws:TagKeys (p. 1441)	
UpdateConfigurationSet	Update the configuration of an event destination for a configuration set	Write	configuration-set* (p. 1441)		

Resource Types Defined by Amazon Pinpoint Email Service

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1436\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
configuration-set	<code>arn:\${Partition}:ses:\${Region}: \${Account}:configuration-set/ \${ConfigurationSetName}</code>	aws:ResourceTag/\${TagKey} (p. 1441)
dedicated-ip-pool	<code>arn:\${Partition}:ses:\${Region}: \${Account}:dedicated-ip-pool/ \${CustomVerificationEmailTemplateName}</code>	aws:ResourceTag/\${TagKey} (p. 1441)
deliverability-test-report	<code>arn:\${Partition}:ses:\${Region}: \${Account}:deliverability-test-report/ \${CustomVerificationEmailTemplateName}</code>	aws:ResourceTag/\${TagKey} (p. 1441)
event-destination	<code>arn:\${Partition}:ses:\${Region}: \${Account}:configuration-set/ \${ConfigurationSetName}:event-destination/ \${EventDestinationName}</code>	
identity	<code>arn:\${Partition}:ses:\${Region}: \${Account}:identity/\${IdentityName}</code>	aws:ResourceTag/\${TagKey} (p. 1441)

Condition Keys for Amazon Pinpoint Email Service

Amazon Pinpoint Email Service defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Filters actions based on the presence of tag key-value pairs in the request	String
aws:ResourceTag/\${TagKey}	Filters actions based on tag key-value pairs attached to the resource	String
aws:TagKeys	Filters actions based on the presence of tag keys in the request	String
	The "Return-Path" address, which specifies where bounces ses:FeedbackAddress and complaints are sent by email feedback forwarding.	String
ses:FromAddress	The "From" address of a message.	String

Condition Keys	Description	Type
ses:FromDisplayNamemessage.	The "From" address that is used as the display name of a message.	String
ses:Recipients	The recipient addresses of a message, which include the "To", "CC", and "BCC" addresses.	String

Actions, Resources, and Condition Keys for Amazon Pinpoint SMS and Voice Service

Amazon Pinpoint SMS and Voice Service (service prefix: `sms-voice`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Pinpoint SMS and Voice Service \(p. 1442\)](#)
- [Resource Types Defined by Amazon Pinpoint SMS and Voice Service \(p. 1443\)](#)
- [Condition Keys for Amazon Pinpoint SMS and Voice Service \(p. 1443\)](#)

Actions Defined by Amazon Pinpoint SMS and Voice Service

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateConfigurationSet	Create a new configuration set. After you create the configuration set, you can add one or more event destinations to it.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateConfigurationSet	Create a new event destination in a configuration set.	Write			iam:PassRole
DeleteConfigurationSet	Deletes an existing configuration set.	Write			
DeleteConfigurationDestination	Deletes an event destination in a configuration destination.	Write			
GetConfigurationEventDestination	Obtain information about an event destination, including the types of events it reports, the Amazon Resource Name (ARN) of the destination, and the name of the event destination.	Read			
ListConfigurationSets	Return a list of configuration sets. This operation only returns the configuration sets that are associated with your account in the current AWS Region.	Read			
SendVoiceMessage	Create a new voice message and send it to a recipient's phone number.	Write			
UpdateConfigurationSet	Update an event destination in a configuration set. An event destination is a location that you publish information about your voice calls to. For example, you can log an event to an Amazon CloudWatch destination when a call fails.	Write			iam:PassRole

Resource Types Defined by Amazon Pinpoint SMS and Voice Service

Amazon Pinpoint SMS and Voice Service does not support specifying a resource ARN in the `Resource` element of an IAM policy statement. To allow access to Amazon Pinpoint SMS and Voice Service, specify `"Resource": "*"` in your policy.

Condition Keys for Amazon Pinpoint SMS and Voice Service

Pinpoint SMS Voice has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Amazon Polly

Amazon Polly (service prefix: `polly`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Polly \(p. 1444\)](#)
- [Resource Types Defined by Amazon Polly \(p. 1445\)](#)
- [Condition Keys for Amazon Polly \(p. 1445\)](#)

Actions Defined by Amazon Polly

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteLexicon	Deletes the specified pronunciation lexicon stored in an AWS Region	Write	lexicon* (p. 1445)		
DescribeVoices	Returns the list of voices that are available for use when requesting speech synthesis.	List			
GetLexicon	Returns the content of the specified pronunciation lexicon stored in an AWS Region.	Read	lexicon* (p. 1445)		
GetSpeechSynthesisTask	Enables the user to get information about specific speech synthesis task.	Read			
ListLexicons	Returns a list of pronunciation lexicons stored in an AWS Region.	List			
ListSpeechSynthesisTasks	Enables the user to list requested speech synthesis tasks.	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
PutLexicon	Stores a pronunciation lexicon in an AWS Region.	Write			
StartSpeechSynthesisTask	Enables the user to synthesize text inputs to provided S3 location.	Write	lexicon (p. 1445)		s3:PutObject
SynthesizeSpeech	Synthesizes UTF-8 input, plain text or SSML, to a stream of bytes.	Read	lexicon (p. 1445)		

Resource Types Defined by Amazon Polly

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1444\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
lexicon	<code>arn:\${Partition}:polly:\${Region}:\${Account}:lexicon/\${LexiconName}</code>	

Condition Keys for Amazon Polly

Polly has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the [IAM Policy Reference](#).

Actions, Resources, and Condition Keys for AWS Price List

AWS Price List (service prefix: `pricing`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Price List \(p. 1446\)](#)
- [Resource Types Defined by AWS Price List \(p. 1446\)](#)
- [Condition Keys for AWS Price List \(p. 1446\)](#)

Actions Defined by AWS Price List

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeServices	Returns the service details for all (paginated) services (if <code>serviceCode</code> is not set) or service detail for a particular service (if given <code>serviceCode</code>).	Read			
GetAttributeValue	Returns all (paginated) possible values for a given attribute.	Read			
GetProducts	Returns all matching products with given search criteria.	Read			

Resource Types Defined by AWS Price List

AWS Price List does not support specifying a resource ARN in the `Resource` element of an IAM policy statement. To allow access to AWS Price List, specify "Resource": "*" in your policy.

Condition Keys for AWS Price List

Price List has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for AWS Private Marketplace

AWS Private Marketplace (service prefix: `aws-marketplace`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Private Marketplace \(p. 1447\)](#)
- [Resource Types Defined by AWS Private Marketplace \(p. 1451\)](#)
- [Condition Keys for AWS Private Marketplace \(p. 1451\)](#)

Actions Defined by AWS Private Marketplace

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AssociateProductWithPrivateMarketplace [permission only]	Adds new approved products to the Private Marketplace. Also allows to approve a request for a product to be associated with the Private Marketplace. This action can be performed by any account in an AWS Organization, provided the user has permissions to do so, and the Organization's Service Control Policies allow it.	Write			
CreatePrivateMarketplaceForTheEntireOrganization [permission only]	Creates a Private Marketplace for the individual account, or for the entire AWS Organization if one exists. This action can only be performed by the master account if using an AWS Organization.	Write			
CreatePrivateMarketplaceProfile [permission only]	Creates a Private Marketplace Profile that customizes the white label experience on the AWS Marketplace website for the individual account, or for the entire AWS Organization if one exists. This action can only be performed by the	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
	master account if using an AWS Organization.				
CreatePrivateMarketplaceProducts [permission only]	Creates a new request for a product to be added to be associated with the Private Marketplace. This action can be performed by any account in an AWS Organization, provided the user has permissions to do so, and the Organization's Service Control Policies allow it.	Write			
DescribePrivateMarketplaceRequestedProducts [permission only]	Describes the status of requested products in the Private Marketplace for administrative purposes. This action can be performed by any account in an AWS Organization, provided the user has permissions to do so, and the Organization's Service Control Policies allow it.	List			
DescribePrivateMarketplaceProfile [permission only]	Describes details about the Private Marketplace Profile for administrative purposes. This action can be performed by any account in an AWS Organization, provided the user has permissions to do so, and the Organization's Service Control Policies allow it.	Read			
DescribePrivateMarketplaceAssociatedProducts [permission only]	Describes requests and associated products in the Private Marketplace. This action can be performed by any account in an AWS Organization, provided the user has permissions to do so, and the Organization's Service Control Policies allow it.	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribePrivateMarketplaceSettings [permission only]	Describes the Private Marketplace settings. This includes setting for enabling requests from end users and preferences for notifications. This action can be performed by any account in an AWS Organization, provided the user has permissions to do so, and the Organization's Service Control Policies allow it.	Read			
DescribePrivateMarketplace [permission only]	Describes the status of the Private Marketplace for administrative purposes. This action can be performed by any account in an AWS Organization, provided the user has permissions to do so, and the Organization's Service Control Policies allow it.	Read			
DisassociateProductFromThePrivateMarketplace [permission only]	Removes approved products from the Private Marketplace. Also allows to decline a request for a product to be associated with the Private Marketplace. This action can be performed by any account in an AWS Organization, provided the user has permissions to do so, and the Organization's Service Control Policies allow it.	Write			
ListPrivateMarketplaceProducts [permission only]	Queryable list for the products and status of products in the Private Marketplace for administrative purposes. This action can be performed by any account in an AWS Organization, provided the user has permissions to do so, and the Organization's Service Control Policies allow it.	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListPrivateMarketplaceAssociatedProducts [permission only]	Queryable list for requests [permission only] products in the Private Marketplace. This action can be performed by any account in an AWS Organization, provided the user has permissions to do so, and the Organization's Service Control Policies allow it.	List			
StartPrivateMarketplace [permission only]	Starts the Private Marketplace, enabling the customized AWS Marketplace experience, and enabling restrictions on the procurement of products based on what is available in the Private Marketplace. This action can be performed by any account in an AWS Organization, provided the user has permissions to do so, and the Organization's Service Control Policies allow it.	Write			
StopPrivateMarketplace [permission only]	Stops the Private Marketplace, disabling the customized AWS Marketplace experience and removing the Private Marketplace procurement restrictions on products. This action can be performed by any account in an AWS Organization, provided the user has permissions to do so, and the Organization's Service Control Policies allow it.	Write			
UpdatePrivateMarketplaceProfile [permission only]	Updates the Private Marketplace Profile that customizes the white label experience on the AWS Marketplace website for the individual account, or for the entire AWS Organization if one exists. This action can be performed by any account in an AWS Organization, provided the user has permissions to do so, and the Organization's Service Control Policies allow it.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UpdatePrivateMarketplaceSettings [permission only]	Updates the Private Marketplace settings. This includes setting for enabling requests from end users and preferences for notifications. This action can be performed by any account in an AWS Organization, provided the user has permissions to do so, and the Organization's Service Control Policies allow it.	Write			

Resource Types Defined by AWS Private Marketplace

AWS Private Marketplace does not support specifying a resource ARN in the `Resource` element of an IAM policy statement. To allow access to AWS Private Marketplace, specify “`Resource`”: “`*`” in your policy.

Condition Keys for AWS Private Marketplace

Private Marketplace has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Amazon QLDB

Amazon QLDB (service prefix: `qldb`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon QLDB \(p. 1451\)](#)
- [Resource Types Defined by Amazon QLDB \(p. 1453\)](#)
- [Condition Keys for Amazon QLDB \(p. 1453\)](#)

Actions Defined by Amazon QLDB

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources (“`*`”) in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in

a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateLedger	Grants permission to create a ledger	Write	ledger* (p. 1453)		
				aws:RequestTag/\${TagKey} (p. 1454) aws:TagKeys (p. 1454)	
DeleteLedger	Grants permission to delete a ledger	Write	ledger* (p. 1453)		
DescribeJournalS3Export	Grants permission to describe information about a journal export job	Read	ledger* (p. 1453)		
DescribeLedger	Grants permission to describe a ledger	Read	ledger* (p. 1453)		
ExecuteStatement	Grants permission to send commands to a ledger via the console	Write	ledger* (p. 1453)		
ExportJournalToS3	Grants permission to export journal contents to an Amazon S3 bucket	Write	ledger* (p. 1453)		
GetBlock	Grants permission to retrieve a block from a ledger for a given BlockAddress	Read	ledger* (p. 1453)		
GetDigest	Grants permission to retrieve a digest from a ledger for a given BlockAddress	Read	ledger* (p. 1453)		
GetRevision	Grants permission to retrieve a revision for a given document ID and a given BlockAddress	Read	ledger* (p. 1453)		
InsertSampleData	Grants permission to insert sample application data via the console	Write	ledger* (p. 1453)		
ListJournalS3Exports	Grants permission to list journal export jobs for all ledgers	List			
ListJournalS3Exports	Grants permission to list journal export jobs for a specified ledger	List	ledger* (p. 1453)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListLedgers	Grants permission to list existing ledgers	List			
ListTagsForResource	Grants permission to list tags for a resource	Read	ledger (p. 1453)		
SendCommand	Grants permission to send commands to a ledger	Write	ledger* (p. 1453)		
ShowCatalog	Grants permission to view a ledger's catalog via the console	Write	ledger* (p. 1453)		
TagResource	Grants permission to add one or more tags to a resource	Tagging	ledger (p. 1453)		
				aws:RequestTag/\${TagKey} (p. 1454)	
				aws:TagKeys (p. 1454)	
UntagResource	Grants permission to remove one or more tags to a resource	Tagging	ledger (p. 1453)		
				aws:TagKeys (p. 1454)	
UpdateLedger	Grants permission to update properties on a ledger	Write	ledger* (p. 1453)		

Resource Types Defined by Amazon QLDB

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 1451\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
ledger	<code>arn:\${Partition}:qldb:\${Region}:\${Account}:ledger/\${LedgerName}</code>	aws:ResourceTag/\${TagKey} (p. 1454)

Condition Keys for Amazon QLDB

Amazon QLDB defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Filters actions based on the presence of tag key-value pairs in the request	String
aws:ResourceTag/\${TagKey}	Filters actions based on tag key-value pairs attached to the resource	String
aws:TagKeys	Filters actions based on the presence of tag keys in the request	String

Actions, Resources, and Condition Keys for Amazon QuickSight

Amazon QuickSight (service prefix: `quicksight`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon QuickSight \(p. 1454\)](#)
- [Resource Types Defined by Amazon QuickSight \(p. 1457\)](#)
- [Condition Keys for Amazon QuickSight \(p. 1457\)](#)

Actions Defined by Amazon QuickSight

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateAdmin [permission only]	CreateAdmin enables the user to provision Amazon QuickSight	Write	user* (p. 1457)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
	administrators, authors, and readers.				
CreateGroup	Create a QuickSight group.	Write	group* (p. 1457)		
CreateGroupMember	Add a QuickSight user to a QuickSight group.	Write	group* (p. 1457)	quicksight:UserName (p. 1457)	
CreateReader [permission only]	CreateReader enables the user to provision Amazon QuickSight readers.	Write	user* (p. 1457)		
CreateUser [permission only]	CreateUser enables the user to provision Amazon QuickSight authors and readers.	Write	user* (p. 1457)		
DeleteGroup	Remove a user group from QuickSight.	Write	group* (p. 1457)		
DeleteGroupMember	Remove a user from a group so that he/she is no longer a member of the group.	Write	group* (p. 1457)	quicksight:UserName (p. 1457)	
DeleteUser	Delete the QuickSight user that is associated with the identity of the IAM user/role making the call. The IAM user is not deleted as a result of this call.	Write	user* (p. 1457)		
DeleteUserByPrincipal	Deletes a user identified by its principal ID.	Write	user* (p. 1457)		
DescribeGroup	Return a QuickSight group's description and ARN.	Read	group* (p. 1457)		
DescribeUser	Return information about a user, given the user name.	Read	user* (p. 1457)		
GetDashboardEmbedUrl	Return a QuickSight dashboard embedding URL.	Read	dashboard* (p. 1457)		
GetGroupMapping [permission only]	GetGroupMapping is used only in Amazon QuickSight Enterprise edition accounts. It enables the user to use Amazon QuickSight to identify and display the Microsoft Active Directory (Microsoft Active Directory) directory groups that are mapped to roles in Amazon QuickSight.	Read			
ListGroupMembers	Return a list of member users in a group.	List	group* (p. 1457)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListGroups	Get a list of all user groups in QuickSight.	List	group* (p. 1457)		
ListUserGroups	Return a list of groups that a given user is a member of.	List	user* (p. 1457)		
ListUsers	Return a list of all of the QuickSight users belonging to this account.	List	user* (p. 1457)		
RegisterUser	Create a QuickSight user, whose identity is associated with the IAM identity/role specified in the request.	Write	user* (p. 1457)	quicksight:iamArn (p. 1457) quicksight:SessionName (p. 1457)	
SearchDirectoryGroups [permission only]	SearchDirectoryGroups is used only in Amazon QuickSight Enterprise edition accounts. It enables the user to use Amazon QuickSight to display your Microsoft Active Directory directory groups so that you can choose which ones to map to roles in Amazon QuickSight.	Write			
SetGroupMapping [permission only]	SearchDirectoryGroups is used only in Amazon QuickSight Enterprise edition accounts. It enables the user to use Amazon QuickSight to display your Microsoft Active Directory directory groups so that you can choose which ones to map to roles in Amazon QuickSight.	Write			
Subscribe [permission only]	Subscribe enables the user to subscribe to Amazon QuickSight. Enabling this action also allows the user to upgrade the subscription to Enterprise edition.	Write			
Unsubscribe [permission only]	Unsubscribe enables the user to unsubscribe from Amazon QuickSight, which permanently deletes all users and their resources from Amazon QuickSight.	Write			
UpdateGroup	Change group description.	Write	group* (p. 1457)		
UpdateUser	Updates an Amazon QuickSight user.	Write	user* (p. 1457)		

Resource Types Defined by Amazon QuickSight

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 1454\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
user	arn:\${Partition}:quicksight:\${Region}: \${Account}:user/\${ResourceId}	
group	arn:\${Partition}:quicksight:\${Region}: \${Account}:group/\${ResourceId}	
dashboard	arn:\${Partition}:quicksight:\${Region}: \${Account}:dashboard/\${ResourceId}	

Condition Keys for Amazon QuickSight

Amazon QuickSight defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
quicksight:iamArn	IAM user ARN or role ARN.	String
quicksight:SessionName	The session name.	String
quicksight:UserName	The user name.	String

Actions, Resources, and Condition Keys for Amazon RDS

Amazon RDS (service prefix: `rds`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon RDS \(p. 1458\)](#)
- [Resource Types Defined by Amazon RDS \(p. 1474\)](#)

- [Condition Keys for Amazon RDS \(p. 1476\)](#)

Actions Defined by Amazon RDS

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AddRoleToDBCluster	Associates an Identity and Access Management (IAM) role from an Aurora DB cluster.	Write	cluster* (p. 1475)		iam:PassRole
AddRoleToDBInstance	Associates an AWS Identity and Access Management (IAM) role with a DB instance.	Write	db* (p. 1475)		iam:PassRole
AddSourceIdentifier	Adds a source identifier to an existing RDS event subscription.	Write	es* (p. 1475)		
AddTagsToResource	Adds metadata tags to an Amazon RDS resource.	Tagging	db (p. 1475)		
			es (p. 1475)		
			og (p. 1476)		
			pg (p. 1476)		
			ri (p. 1476)		
			secgrp (p. 1476)		
			snapshot (p. 1476)		
			subgrp (p. 1476)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:RequestTag/ \${TagKey} (p. 1477) aws:TagKeys (p. 1477) rds:req- tag/ \${TagKey} (p. 1478)	
ApplyPendingMaintenanceActionToResource	Applies a pending maintenance action to a resource.	Write	db* (p. 1475)		
AuthorizeDBSecurityGroupIngress	Enables ingress to a DB Security Group using one of two forms of authorization.	Permissions management	secgrp* (p. 1476)		
BacktrackDBCluster	Backtracks a DB cluster to a specific time, without creating a new DB cluster.	Write	cluster* (p. 1475)		
CancelExportTask	Cancels an export task in progress.	Write			
CopyDBClusterParameterGroup	Copies the specified DB cluster parameter group.	Write	cluster- pg* (p. 1475)		
CopyDBClusterSnapshot	Creates a snapshot of a DB cluster.	Write	cluster- snapshot* (p. 1475)		
CopyDBParameterGroup	Copies the specified DB parameter group.	Write	pg* (p. 1476)		
CopyDBSnapshot	Copies the specified DB snapshot.	Write	snapshot* (p. 1476)		
CopyOptionGroup	Copies the specified option group.	Write	og* (p. 1476)		
CreateDBCluster	Creates a new Amazon Aurora DB cluster.	Tagging	cluster* (p. 1475)		iam:PassRole
			cluster- pg* (p. 1475)		
			og* (p. 1476)		
			subgrp* (p. 1476)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:RequestTag/\${TagKey} (p. 1477) aws:TagKeys (p. 1477) rds:req-tag/\${TagKey} (p. 1478) rds:DatabaseEngine (p. 1477) rds:DatabaseName (p. 1477) rds:StorageEncrypted (p. 1477)	
CreateDBClusterEndpoint Associates it with an Amazon Aurora DB cluster.		Write	cluster* (p. 1475)		
			cluster-endpoint* (p. 1475)		
				rds:EndpointType (p. 1477) aws:RequestTag/\${TagKey} (p. 1477) aws:TagKeys (p. 1477)	
CreateDBClusterParameterGroup	Create a new DB cluster parameter group.	Tagging	cluster-pg* (p. 1475)		
				aws:RequestTag/\${TagKey} (p. 1477) aws:TagKeys (p. 1477) rds:req-tag/\${TagKey} (p. 1478)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateDBClusterSnapshot	Creates a snapshot of a DB cluster.	Tagging	cluster* (p. 1475) cluster-snapshot* (p. 1475)		
				aws:RequestTag/\${TagKey} (p. 1477) aws:TagKeys (p. 1477) rds:req-tag/\${TagKey} (p. 1478)	
CreateDBInstance	Creates a new DB instance.	Tagging	db* (p. 1475) og* (p. 1476) pg* (p. 1476) secgrp* (p. 1476) subgrp* (p. 1476)		iam:PassRole
				aws:RequestTag/\${TagKey} (p. 1477) aws:TagKeys (p. 1477) rds:req-tag/\${TagKey} (p. 1478)	
CreateDBInstanceReadReplica	Creates a DB instance that acts as a Read Replica of a source DB instance.	Tagging	db* (p. 1475) og* (p. 1476) subgrp* (p. 1476)		iam:PassRole

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:RequestTag/ \${TagKey} (p. 1477) aws:TagKeys (p. 1477) rds:req- tag/ \${TagKey} (p. 1478)	
CreateDBParameterGroup	Creates a new DB parameter group	Tagging	pg* (p. 1476)		
				aws:RequestTag/ \${TagKey} (p. 1477) aws:TagKeys (p. 1477) rds:req- tag/ \${TagKey} (p. 1478)	
CreateDBProxy	Grants permission to create a database proxy	Write			iam:PassRole
CreateDBSecurityGroup	Creates a new DB security group. DB security groups control access to a DB instance.	Tagging	secgrp* (p. 1476)		
				aws:RequestTag/ \${TagKey} (p. 1477) aws:TagKeys (p. 1477) rds:req- tag/ \${TagKey} (p. 1478)	
CreateDBSnapshot	Creates a DBSnapshot.	Tagging	db* (p. 1475)		
				snapshot* (p. 1476)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:RequestTag/\${TagKey} (p. 1477) aws:TagKeys (p. 1477) rds:req-tag/\${TagKey} (p. 1478)	
CreateDBSubnetGroup	Creates a new DB subnet group.	Tagging	subgrp* (p. 1476)		
				aws:RequestTag/\${TagKey} (p. 1477) aws:TagKeys (p. 1477) rds:req-tag/\${TagKey} (p. 1478)	
CreateEventSubscription	Creates an RDS event notification subscription.	Tagging	es* (p. 1475)		
				aws:RequestTag/\${TagKey} (p. 1477) aws:TagKeys (p. 1477) rds:req-tag/\${TagKey} (p. 1478)	
CreateGlobalCluster	Creates an Aurora global database spread across multiple regions.	Write	cluster* (p. 1475)		
			global-cluster* (p. 1476)		
CreateOptionGroup	Creates a new option group.	Tagging	og* (p. 1476)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:RequestTag/ \${TagKey} (p. 1477) aws:TagKeys (p. 1477) rds:req- tag/ \${TagKey} (p. 1478)	
DeleteDBCluster	The DeleteDBCluster action deletes a previously provisioned DB cluster.	Write	cluster* (p. 1475)		
			cluster- snapshot* (p. 1475)		
DeleteDBClusterEndpoint	Deletes a custom endpoint and removes it from an Amazon Aurora DB cluster.	Write	cluster- endpoint*	(p. 1475)	
DeleteDBClusterParameterGroup	Deletes a specified DB cluster parameter group.	Write	cluster- pg*	(p. 1475)	
DeleteDBClusterSnapshot	Deletes a DB cluster snapshot.	Write	cluster- snapshot*	(p. 1475)	
DeleteDBInstance	The DeleteDBInstance action deletes a previously provisioned DB instance.	Write	db*	(p. 1475)	
DeleteDBInstanceBasedOnTheSource	Deletes automated backups based on the source instance's DbiResourceId value or the restorable instance's resource ID.	Write			
DeleteDBParameterGroup	Deletes a specified DB parameter group.	Write	pg*	(p. 1476)	
DeleteDBProxy	Grants permission to delete a database proxy.	Write	proxy*	(p. 1476)	
DeleteDBSecurityGroup	Deletes a DB security group.	Write	secgrp*	(p. 1476)	
DeleteDBSnapshot	Deletes a DB snapshot.	Write	snapshot*	(p. 1476)	
DeleteDBSubnetGroup	Deletes a DB subnet group.	Write	subgrp*	(p. 1476)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteEventSubscription	Deletes an RDS event notification subscription.	Write	es* (p. 1475)		
DeleteGlobalCluster	Deletes a global database cluster.	Write	global-cluster* (p. 1476)		
DeleteOptionGroup	Deletes an existing option group.	Write	og* (p. 1476)		
DeregisterDBProxyTargets	Grants permission to remove targets from a database proxy target group	Write	cluster* (p. 1475)		
			db* (p. 1475)		
			proxy* (p. 1476)		
			target-group* (p. 1476)		
DescribeAccountAttributes	Lists all of the attributes for a customer account.	List			
DescribeCertificates	Lists the set of CA certificates provided by Amazon RDS for this AWS account.	List			
DescribeDBClusterBacktracks	Returns information about backtracks for a DB cluster.	List	cluster* (p. 1475)		
DescribeDBClusterEndpoints	Returns information about endpoints for an Amazon Aurora DB cluster.	List			
DescribeDBClusterParameterGroups	Returns a list of DB cluster parameter group descriptions.	List	cluster-pg* (p. 1475)		
DescribeDBClusterParameters	Returns the detailed parameter list for a particular DB cluster parameter group.	List	cluster-pg* (p. 1475)		
DescribeDBClusterSnapshotAttributes	Returns a list of DB cluster snapshot attributes and values for a manual DB cluster snapshot.	List	cluster-snapshot* (p. 1475)		
DescribeDBClusterSnapshots	Returns information about DB cluster snapshots.	Read			
DescribeDBClusters	Returns information about provisioned Aurora DB clusters.	List	cluster* (p. 1475)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeDBEngines	Returns a list of the available DB engines.	List	pg* (p. 1476)		
DescribeDBInstances	Returns a list of automated backups for both current and deleted instances.	List			
DescribeDBInstances	Returns information about provisioned RDS instances.	List			
DescribeDBLogFileList	Returns a list of DB log files for the DB instance.	List	db* (p. 1475)		
DescribeDBParameterGroups	Returns a list of DB Parameter Group descriptions.	List	pg* (p. 1476)		
DescribeDBParameters	Returns the detailed parameter list for a particular DB parameter group.	List	pg* (p. 1476)		
DescribeDBProxies	Grants permission to view proxies	List	proxy* (p. 1476)		
DescribeDBProxyTargets	Grants permission to view database proxy target group details	List	proxy* (p. 1476)		
DescribeDBProxyTargets	Grants permission to view database proxy target details	List	cluster* (p. 1475)		
			db* (p. 1475)		
			proxy* (p. 1476)		
			target-group* (p. 1476)		
DescribeDBSecurityGroups	Returns a list of DB Security Group descriptions.	List	secgrp* (p. 1476)		
DescribeDBSnapshots	Returns a list of DB snapshot attributes and values for a manual DB snapshot.	List	snapshot* (p. 1476)		
DescribeDBSnapshots	Returns information about DB snapshots.	List	db* (p. 1475)		
DescribeDBSubnets	Returns a list of DBSubnetGroup descriptions.	List	snapshot* (p. 1476)		
			subgrp* (p. 1476)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeEngineDefaultSystemParameters	Returns the default engine and system parameter information for the cluster database engine.	List			
DescribeEngineSystemParameters	Returns the default engine and system parameter information for the specified database engine.	List			
DescribeEventCategories	Displays a list of categories for event source types, or, if specified, for a specified source type.	List			
DescribeEventSubscriptions	Lists all the subscription descriptions for a customer account.	List	es* (p. 1475)		
DescribeEvents	Returns events related to DB instances, DB security groups, DB snapshots, and DB parameter groups for the past 14 days.	List	es* (p. 1475)		
DescribeExportTasks	Returns information about the export tasks.	List			
DescribeGlobalClusters	Returns information about Aurora global database clusters.	List			
DescribeOptionGroupOptions	Describes all available options.	List	og* (p. 1476)		
DescribeOptionGroups	Describes the available option groups.	List	og* (p. 1476)		
DescribeOrderableDBInstanceOptions	Returns a list of orderable DB instance options for the specified engine.	List			
DescribePendingMaintenanceActions	Returns a list of resources (for example, DB instances) that have at least one pending maintenance action.	List	db* (p. 1475)		
DescribeReservedDBInstances	Returns information about Reserved DB instances for this account, or about a specified reserved DB instance.	List	ri* (p. 1476)		
DescribeReservedDBInstanceOfferings	Lists available reserved DB instance offerings.	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeSourceRegions	Returns a list of the source AWS Regions where the current AWS Region can create a Read Replica or copy a DB snapshot from.	List			
DescribeValidDBModifications	Lists available modifications you can make to your DB instance	List	db* (p. 1475)		
DownloadCompleteLogFile	Downloads the contents of the specified database log file.	Read			
DownloadDBLogFilePart	Downloads all or a portion of the specified log file, up to 1 MB in size.	Read	db* (p. 1475)		
FailoverDBCluster	Forces a failover for a DB cluster.	Write	cluster* (p. 1475)		
ListTagsForResource	Lists all tags on an Amazon RDS resource.	Read	db (p. 1475)		
es (p. 1475)					
og (p. 1476)					
pg (p. 1476)					
ri (p. 1476)					
secgrp (p. 1476)					
snapshot (p. 1476)					
subgrp (p. 1476)					
ModifyCurrentDBCapacityForAuroraServerless	Modify current cluster capacity for an Amazon Aurora Severless DB cluster.	Write	cluster* (p. 1475)		
ModifyDBCluster	Modify a setting for an Amazon Aurora DB cluster.	Write	cluster* (p. 1475)		iam:PassRole
cluster-pg* (p. 1475)					
og* (p. 1476)					

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ModifyDBClusterEndpoint	Modifies the properties of an endpoint in an Amazon Aurora DB cluster.	Write	cluster-endpoint* (p. 1475)		
ModifyDBClusterParameterGroup	Modifies the parameters of a DB cluster parameter group.	Write	cluster-pg* (p. 1475)		
ModifyDBClusterSnapshotAttribute	Adds an attribute and values to or removes an attribute and values from, a manual DB cluster snapshot.	Write	cluster-snapshot* (p. 1475)		
ModifyDBInstance	Modify settings for a DB instance.	Write	db* (p. 1475)		iam:PassRole
			og* (p. 1476)		
			pg* (p. 1476)		
			secgrp* (p. 1476)		
ModifyDBParameterGroup	Modifies the parameters of a DB parameter group.	Write	pg* (p. 1476)		
ModifyDBProxy	Grants permission to modify database proxy	Write	proxy* (p. 1476)		iam:PassRole
ModifyDBProxyTargetGroup	Grants permission to modify target group for a database proxy	Write	target-group* (p. 1476)		
ModifyDBSnapshot	Updates a manual DB snapshot, which can be encrypted or not encrypted, with a new engine version.	Write	snapshot* (p. 1476)		
ModifyDBSnapshotAttribute	Adds an attribute and values to, or removes an attribute and values from, a manual DB snapshot.	Write	snapshot* (p. 1476)		
ModifyDBSubnetGroup	Modifies an existing DB subnet group.	Write	subgrp* (p. 1476)		
ModifyEventSubscription	Modifies an existing RDS event notification subscription.	Write	es* (p. 1475)		
ModifyGlobalCluster	Modify a setting for an Amazon Aurora global cluster.	Write	global-cluster* (p. 1476)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ModifyOptionGroup	Modifies an existing option group.	Write	og* (p. 1476)		iam:PassRole
PromoteReadReplica	Promotes a Read Replica DB instance to a standalone DB instance.	Write	db* (p. 1475)		
PromoteReadReplicaCluster	Promotes a Read Replica DB cluster to a standalone DB cluster.	Write	cluster* (p. 1475)		
PurchaseReservedDBInstancesOffering	Purchases a reserved DB instance offering.	Write	ri* (p. 1476)		
				aws:RequestTag/\${TagKey} (p. 1477)	aws:TagKeys (p. 1477)
RebootDBInstance	Rebooting a DB instance restarts the database engine service.	Write	db* (p. 1475)		
RegisterDBProxyTarget	Grants permission to add targets to a database proxy target group.	Write	target-group* (p. 1476)		
RemoveFromGlobalCluster	Detaches an Aurora secondary cluster from an Aurora global database cluster.	Write	cluster* (p. 1475)		
			global-cluster* (p. 1476)		
RemoveRoleFromDBCluster	Disassociates an AWS Identity and Access Management (IAM) role from an Amazon Aurora DB cluster.	Write	cluster* (p. 1475)		iam:PassRole
RemoveRoleFromDBInstance	Disassociates an AWS Identity and Access Management (IAM) role from a DB instance.	Write	db* (p. 1475)		iam:PassRole
RemoveSourceIdentifier	Removes a source identifier from an existing RDS event notification subscription.	Write	es* (p. 1475)		
RemoveTagsFromAmazonRDS	Removes metadata tags from an Amazon RDS resource.	Tagging	db (p. 1475)		
			es (p. 1475)		
			og (p. 1476)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			pg (p. 1476)		
			ri (p. 1476)		
			secgrp (p. 1476)		
			snapshot (p. 1476)		
			subgrp (p. 1476)		
				aws:RequestTag/ \${TagKey} (p. 1477)	
				aws:TagKeys (p. 1477)	
				rds:req- tag/ \${TagKey} (p. 1478)	
ResetDBClusterParameterGroup	Modifies the parameters of a DB cluster parameter group to the default value.	Write	cluster- pg* (p. 1475)		
ResetDBParameterGroup	Modifies the parameters of a DB parameter group to the engine/system default value.	Write	pg* (p. 1476)		
RestoreDBClusterFromClusterSnapshot	Creates an Amazon Aurora DB cluster from data stored in an Amazon S3 bucket.	Write	cluster* (p. 1475)		iam:PassRole

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:RequestTag/\${TagKey} (p. 1477) aws:TagKeys (p. 1477) rds:req-tag/\${TagKey} (p. 1478) rds:DatabaseEngine (p. 1477) rds:DatabaseName (p. 1477) rds:StorageEncrypted (p. 1477)	
RestoreDBClusterFromDBClusterSnapshot	Creates a new DB cluster from a DB cluster snapshot.	Write	cluster* (p. 1475) cluster-snapshot* (p. 1475) og* (p. 1476)		iam:PassRole
				aws:RequestTag/\${TagKey} (p. 1477) aws:TagKeys (p. 1477) rds:req-tag/\${TagKey} (p. 1478)	
RestoreDBClusterToPointInTime	Restores a DB cluster to an arbitrary point in time.	Write	cluster* (p. 1475) og* (p. 1476) subgrp* (p. 1476)		iam:PassRole

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:RequestTag/\${TagKey} (p. 1477) aws:TagKeys (p. 1477) rds:req-tag/\${TagKey} (p. 1478)	
RestoreDBInstanceFromDBSnapshot	Creates a new DB instance from a DB snapshot.	Write	db* (p. 1475) og* (p. 1476) snapshot* (p. 1476) subgrp* (p. 1476)	aws:RequestTag/\${TagKey} (p. 1477) aws:TagKeys (p. 1477) rds:req-tag/\${TagKey} (p. 1478)	iam:PassRole
RestoreDBInstanceFromAmazonS3	Creates a new DB instance from an Amazon S3 bucket.	Write	db* (p. 1475) aws:RequestTag/\${TagKey} (p. 1477) aws:TagKeys (p. 1477) rds:req-tag/\${TagKey} (p. 1478)	aws:RequestTag/\${TagKey} (p. 1477) aws:TagKeys (p. 1477) rds:req-tag/\${TagKey} (p. 1478)	iam:PassRole
RestoreDBInstanceToPointInTime	Restores a DB instance to an arbitrary point in time.	Write	db* (p. 1475)		iam:PassRole

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			og* (p. 1476)		
			snapshot* (p. 1476)		
			subgrp* (p. 1476)		
	Revokes ingress from a DB Security Group for previously authorized IP ranges or EC2 or VPC Security Groups.	Write	secgrp* (p. 1476)		
StartActivityStream	Enables the user to start Activity Stream.	Write	cluster* (p. 1475)		
StartDBCluster	Starts the DB cluster.	Write	cluster* (p. 1475)		
StartDBInstance	Starts the DB instance.	Write	db* (p. 1475)		
StartExportTask	Starts a new Export task for a DB snapshot.	Write			iam:PassRole
StopActivityStream	Enables the user to stop Activity Stream.	Write	cluster* (p. 1475)		
StopDBCluster	Stops the DB cluster.	Write	cluster* (p. 1475)		
StopDBInstance	Stops the DB instance.	Write	db* (p. 1475)		

Resource Types Defined by Amazon RDS

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1458\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
cluster	arn:\${Partition}:rds:\${Region}: \${Account}:cluster:\${DbClusterInstanceName}	aws:ResourceTag/\${TagKey} (p. 1477) rds:cluster-tag/\${TagKey} (p. 1477)
cluster-endpoint	arn:\${Partition}:rds:\${Region}: \${Account}:cluster-endpoint: \${DbClusterEndpoint}	aws:ResourceTag/\${TagKey} (p. 1477)
cluster-pg	arn:\${Partition}:rds:\${Region}: \${Account}:cluster-pg: \${ClusterParameterGroupName}	aws:ResourceTag/\${TagKey} (p. 1477) rds:cluster-pg-tag/\${TagKey} (p. 1477)
cluster-snapshot	arn:\${Partition}:rds:\${Region}: \${Account}:cluster-snapshot: \${ClusterSnapshotName}	aws:ResourceTag/\${TagKey} (p. 1477) rds:cluster-snapshot-tag/\${TagKey} (p. 1477)
db	arn:\${Partition}:rds:\${Region}: \${Account}:db:\${DbInstanceName}	aws:ResourceTag/\${TagKey} (p. 1477) rds:DatabaseClass (p. 1477) rds:DatabaseEngine (p. 1477) rds:DatabaseName (p. 1477) rds:MultiAz (p. 1477) rds:Piops (p. 1477) rds:StorageEncrypted (p. 1477) rds:StorageSize (p. 1477) rds:Vpc (p. 1477) rds:db-tag/\${TagKey} (p. 1477)
es	arn:\${Partition}:rds:\${Region}: \${Account}:es:\${SubscriptionName}	aws:ResourceTag/\${TagKey} (p. 1477) rds:es-tag/\${TagKey} (p. 1477)

Resource Types	ARN	Condition Keys
global-cluster	arn:\${Partition}:rds:\${Account}:global-cluster:\${GlobalCluster}	
og	arn:\${Partition}:rds:\${Region}:\${Account}:og:\${OptionGroupName}	aws:ResourceTag/\${TagKey} (p. 1477) rds:og-tag/\${TagKey} (p. 1477)
pg	arn:\${Partition}:rds:\${Region}:\${Account}:pg:\${ParameterGroupName}	aws:ResourceTag/\${TagKey} (p. 1477) rds:pg-tag/\${TagKey} (p. 1478)
proxy	arn:\${Partition}:rds:\${Region}:\${Account}:db-proxy:\${DbProxyId}	
ri	arn:\${Partition}:rds:\${Region}:\${Account}:ri:\${ReservedDbInstanceName}	aws:ResourceTag/\${TagKey} (p. 1477) rds:ri-tag/\${TagKey} (p. 1478)
secgrp	arn:\${Partition}:rds:\${Region}:\${Account}:secgrp:\${SecurityGroupName}	aws:ResourceTag/\${TagKey} (p. 1477) rds:secgrp-tag/\${TagKey} (p. 1478)
snapshot	arn:\${Partition}:rds:\${Region}:\${Account}:snapshot:\${SnapshotName}	aws:ResourceTag/\${TagKey} (p. 1477) rds:snapshot-tag/\${TagKey} (p. 1478)
subgrp	arn:\${Partition}:rds:\${Region}:\${Account}:subgrp:\${SubnetGroupName}	aws:ResourceTag/\${TagKey} (p. 1477) rds:subgrp-tag/\${TagKey} (p. 1478)
target	arn:\${Partition}:rds:\${Region}:\${Account}:target:\${TargetId}	
target-group	arn:\${Partition}:rds:\${Region}:\${Account}:target-group:\${TargetGroupId}	

Condition Keys for Amazon RDS

Amazon RDS defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Filters actions based on the presence of tag key-value pairs in the request	String
aws:ResourceTag/\${TagKey}	Filters actions based on tag key-value pairs attached to the resource	String
aws:TagKeys	Filters actions based on the presence of tag keys in the request	String
rds:DatabaseClass	A type of DB instance class.	String
rds:DatabaseEngine	A database engine, such as MySQL.	String
rds:DatabaseName	The user-defined name of the database on the DB instance.	String
rds:EndpointType	The type of the endpoint. One of: READER, WRITER, CUSTOM.	String
rds:MultiAz	A value that specifies whether the DB instance runs in multiple Availability Zones. To indicate that the DB instance is using Multi-AZ, specify true.	Boolean
rds:Piops	A value that contains the number of Provisioned IOPS (PIOPS) that the instance supports. To indicate a DB instance that does not have PIOPS enabled, specify 0.	Numeric
rds:StorageEncrypted	A value that specifies whether the DB instance storage should be encrypted. To enforce storage encryption, specify true.	Boolean
rds:StorageSize	The storage volume size (in GB).	Numeric
rds:Vpc	A value that specifies whether the DB instance runs in an Amazon Virtual Private Cloud (Amazon VPC). To indicate that the DB instance runs in an Amazon VPC, specify true.	Boolean
rds:cluster-pg-tag/\${TagKey}	A tag attached to a DB cluster parameter group.	String
rds:cluster-snapshot-tag/\${TagKey}	A tag attached to a DB cluster snapshot.	String
rds:cluster-tag/\${TagKey}	A tag attached to a DB cluster.	String
rds:db-tag/\${TagKey}	A tag attached to a DB instance.	String
rds:es-tag/\${TagKey}	A tag attached to an event subscription.	String
rds:og-tag/\${TagKey}	A tag attached to a DB option group.	String

Condition Keys	Description	Type
rds:pg-tag/\${TagKey}	A tag attached to a DB parameter group.	String
rds:req-tag/\${TagKey}	Limits the set of tag keys and values that can be used to tag a resource.	String
rds:ri-tag/\${TagKey}	A tag attached to a reserved DB instance.	String
rds:secgrp-tag/\${TagKey}	A tag attached to a DB security group.	String
rds:snapshot-tag/\${TagKey}	A tag attached to a DB snapshot.	String
rds:subgrp-tag/\${TagKey}	A tag attached to a DB subnet group.	String

Actions, Resources, and Condition Keys for Amazon RDS Data API

Amazon RDS Data API (service prefix: rds-data) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon RDS Data API \(p. 1478\)](#)
- [Resource Types Defined by Amazon RDS Data API \(p. 1479\)](#)
- [Condition Keys for Amazon RDS Data API \(p. 1479\)](#)

Actions Defined by Amazon RDS Data API

You can specify the following actions in the Action element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the Resource element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
BatchExecuteStatement	Runs a batch SQL statement over an array of data.	Write			
BeginTransaction	Starts a SQL transaction.	Write			
CommitTransaction	Ends a SQL transaction started with the BeginTransaction operation and commits the changes.	Write			rds-data:BeginTransaction
ExecuteSql	Runs one or more SQL statements. This operation is deprecated. Use the BatchExecuteStatement or ExecuteStatement operation.	Write			
ExecuteStatement	Runs a SQL statement against a database.	Write			
RollbackTransaction	Performs a rollback of a transaction. Rolling back a transaction cancels its changes.	Write			rds-data:BeginTransaction

Resource Types Defined by Amazon RDS Data API

Amazon RDS Data API does not support specifying a resource ARN in the Resource element of an IAM policy statement. To allow access to Amazon RDS Data API, specify “Resource”: “*” in your policy.

Condition Keys for Amazon RDS Data API

RDS Data API has no service-specific context keys that can be used in the Condition element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Amazon RDS IAM Authentication

Amazon RDS IAM Authentication (service prefix: rds-db) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon RDS IAM Authentication \(p. 1480\)](#)
- [Resource Types Defined by Amazon RDS IAM Authentication \(p. 1480\)](#)
- [Condition Keys for Amazon RDS IAM Authentication \(p. 1480\)](#)

Actions Defined by Amazon RDS IAM Authentication

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
<code>connect</code>	Allows IAM role or user to connect to RDS database	Permissions management	db-user* (p. 1480)		

Resource Types Defined by Amazon RDS IAM Authentication

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1480\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
db-user	<code>arn:\${Partition}:rds-db:\${Region}: \${Account}:dbuser:\${DbiResourceId}/ \${DbUserName}</code>	

Condition Keys for Amazon RDS IAM Authentication

RDS IAM Authentication has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Amazon Redshift

Amazon Redshift (service prefix: `redshift`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).

- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Redshift \(p. 1481\)](#)
- [Resource Types Defined by Amazon Redshift \(p. 1489\)](#)
- [Condition Keys for Amazon Redshift \(p. 1491\)](#)

Actions Defined by Amazon Redshift

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AcceptReservedNodeExchange	Grants permission to exchange a DC1 reserved node for a DC2 reserved node with no changes to the configuration	Write			
AuthorizeClusterSubnetIngress	Grants permission to add an <code>AuthorizerSubnetIngress</code> rule to an Amazon Redshift security group	Permissions management	securitygroup* (p. 1490)		
			securitygroupingress-ec2securitygroup* (p. 1490)		
AuthorizeSnapshotRestore	Grants permission to the specified AWS account to restore a snapshot	Permissions management	snapshot* (p. 1490)		
BatchDeleteClusterSnapshots	Grants permission to delete snapshots in a batch of size upto 100	Write	snapshot* (p. 1490)		
BatchModifyClusterSettings	Grants permission to modify settings for a list of snapshots	Write	snapshot* (p. 1490)		
CancelQuery [permission only]	Grants permission to cancel a query through the Amazon Redshift console	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CancelQuerySession [permission only]	Grants permission to see queries in the Amazon Redshift console	Write			
CancelResize	Grants permission to cancel a resize operation	Write	cluster* (p. 1490)		
CopyClusterSnapshot	Grants permission to copy a cluster snapshot	Write	snapshot* (p. 1490)		
CreateCluster	Grants permission to create a cluster	Write	cluster* (p. 1490)		
CreateClusterParameterGroup	Grants permission to create an Amazon Redshift parameter group	Write	parametergroup* (p. 1490)		
CreateClusterSecurityGroup	Grants permission to create an Amazon Redshift security group	Write	securitygroup* (p. 1490)		
CreateClusterSnapshot	Grants permission to create a manual snapshot of the specified cluster	Write	snapshot* (p. 1490)		
CreateClusterSubnetGroup	Grants permission to create an Amazon Redshift subnet group	Write	subnetgroup* (p. 1490)		
CreateClusterUser	Grants permission to automatically create the specified Amazon Redshift user if it does not exist	Permissions management (p. 1490)	dbuser* (p. 1490)		
			redshift:DbUser (p. 1491)		
CreateEventSubscription	Grants permission to create an Amazon Redshift event notification subscription	Write	eventsSubscription* (p. 1490)		
CreateHsmClientCertificate	Grants permission to create an HSM client certificate that a cluster uses to connect to an HSM	Write	hsmClientCertificate* (p. 1490)		
CreateHsmConfiguration	Grants permission to create an HSM configuration that contains information required by a cluster to store and use database encryption keys in a hardware security module (HSM)	Write	hsmConfiguration* (p. 1490)		
CreateSavedQuery [permission only]	Grants permission to create saved SQL queries through the Amazon Redshift console	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateScheduledAction	Grants permission to create an Amazon Redshift scheduled action	Write			
CreateSnapshotCopyGrant	Grants permission to create a snapshot copy grant and encrypt copied snapshots in a destination AWS Region	Permissions management	snapshotcopygrant* (p. 1490)		
CreateSnapshotSchedule	Grants permission to create a snapshot schedule	Write	snapshotschedule* (p. 1490)		
CreateTags	Grants permission to add one or more tags to a specified resource	Tagging			
DeleteCluster	Grants permission to delete a previously provisioned cluster	Write	cluster* (p. 1490)		
DeleteClusterParameterGroup	Grants permission to delete an Amazon Redshift parameter group	Write	parametergroup* (p. 1490)		
DeleteClusterSecurityGroup	Grants permission to delete an Amazon Redshift security group	Write	securitygroup* (p. 1490)		
DeleteClusterSnapshot	Grants permission to delete a manual snapshot	Write	snapshot* (p. 1490)		
DeleteClusterSubnetGroup	Grants permission to delete a subnet group	Write	subnetgroup* (p. 1490)		
DeleteEventSubscription	Grants permission to delete an Amazon Redshift event notification subscription	Write	events_subscription* (p. 1490)		
DeleteHsmClientCertificate	Grants permission to delete an HSM client certificate	Write	hsmclientcertificate* (p. 1490)		
DeleteHsmConfiguration	Grants permission to delete an Amazon Redshift HSM configuration	Write	hsmconfiguration* (p. 1490)		
DeleteSavedQuery [permission only]	Grants permission to delete saved SQL queries through the Amazon Redshift console	Write			
DeleteScheduledAction	Grants permission to delete an Amazon Redshift scheduled action	Write			
DeleteSnapshotCopyGrant	Grants permission to delete a snapshot copy grant	Write	snapshotcopygrant* (p. 1490)		
DeleteSnapshotSchedule	Grants permission to delete a snapshot schedule	Write	snapshotschedule* (p. 1490)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteTags	Grants permission to delete a tag or tags from a resource	Tagging			
DescribeAccountAttributes	Grants permission to describe Attributes attached to the specified AWS account	Read			
DescribeClusterDatabaseRevisions	Grants permission to describe Database revisions for a cluster	List			
DescribeClusterParameterGroups	Grants permission to describe Amazon Redshift parameter groups, including parameter groups you created and the default parameter group	Read			
DescribeClusterParameters	Grants permission to describe parameters contained within an Amazon Redshift parameter group	Read	parametergroup* (p. 1490)		
DescribeClusterSecurityGroups	Grants permission to describe Amazon Redshift security groups	Read			
DescribeClusterSnapshots	Grants permission to describe snapshots , which contain metadata about your cluster snapshots	Read			
DescribeClusterSubnetGroups	Grants permission to describe one or more cluster subnet group objects, which contain metadata about your cluster subnet groups	Read			
DescribeClusterTracks	Grants permission to describe available maintenance tracks	List			
DescribeClusterVersions	Grants permission to describe available Amazon Redshift cluster versions	Read			
DescribeClusters	Grants permission to describe properties of provisioned clusters	List			
DescribeDefaultClusterParameterGroupSettings	Grants permission to describe parameter settings for a parameter group family	Read			
DescribeEventCategories	Grants permission to describe event categories for all event source types, or for a specified source type	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeEventSubscriptions	Grants permission to describe Amazon Redshift event notification subscriptions for the specified AWS account	Read			
DescribeEvents	Grants permission to describe events related to clusters, security groups, snapshots, and parameter groups for the past 14 days	List			
DescribeHsmClientCertificates	Grants permission to describe HSM client certificates	Read			
DescribeHsmConfigurations	Grants permission to describe Amazon Redshift HSM configurations	Read			
DescribeLoggingStatus	Grants permission to describe whether information, such as queries and connection attempts, is being logged for a cluster	Read	cluster* (p. 1490)		
DescribeNodeConfigurationOptions	Grants permission to describe possible node configurations such as node type, number of nodes, and disk usage for the specified action type	List			
DescribeOrderableClusterOptions	Grants permission to describe orderable cluster options	Read			
DescribeQuery [permission only]	Grants permission to describe a query through the Amazon Redshift console	Read			
DescribeReservedNodeOfferings	Grants permission to describe available reserved node offerings by Amazon Redshift	Read			
DescribeReservedNodes	Grants permission to describe the reserved nodes	Read			
DescribeResize	Grants permission to describe the last resize operation for a cluster	Read	cluster* (p. 1490)		
DescribeSavedQueries [permission only]	Grants permission to describe saved queries through the Amazon Redshift console	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeScheduledActions	Grants permission to describe Amazon Redshift scheduled actions	Read			
DescribeSnapshotCopyJobs	Grants permission to describe snapshot copy grants owned by the specified AWS account in the destination AWS Region	Read			
DescribeSnapshotSchedules	Grants permission to describe snapshot schedules	Read	snapshotschedule* (p. 1490)		
DescribeStorage	Grants permission to describe account level backups storage size and provisional storage	Read			
DescribeTable [permission only]	Grants permission to describe a table through the Amazon Redshift console	Read			
DescribeTableRestoreStatus	Grants permission to describe one or more table restore requests made using the RestoreTableFromClusterSnapshot API action	Read			
DescribeTags	Grants permission to describe tags	Read			
DisableLogging	Grants permission to disable logging information, such as queries and connection attempts, for a cluster	Write	cluster* (p. 1490)		
DisableSnapshotCopy	Grants permission to disable the automatic copy of snapshots for a cluster	Write	cluster* (p. 1490)		
EnableLogging	Grants permission to enable logging information, such as queries and connection attempts, for a cluster	Write	cluster* (p. 1490)		
EnableSnapshotCopy	Grants permission to enable the automatic copy of snapshots for a cluster	Write	cluster* (p. 1490)		
ExecuteQuery [permission only]	Grants permission to execute a query through the Amazon Redshift console	Write			
FetchResults [permission only]	Grants permission to fetch query results through the Amazon Redshift console	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetClusterCredentials	Grants permission to get temporary credentials to access an Amazon Redshift database by the specified AWS account	Write	dbuser* (p. 1490)		
	dbgrou (p. 1490)				
	dbname (p. 1490)				
GetReservedNodeOfferings	Get an array of DC2	Read			
	ReservedNodeOfferings that matches the payment type, term, and usage price of the given DC1 reserved node				
JoinGroup	Grants permission to join the specified Amazon Redshift group	Permissions management	dbgrou* (p. 1490)		
ListDatabases [permission only]	Grants permission to list databases through the Amazon Redshift console	List			
ListSavedQueries [permission only]	Grants permission to list saved queries through the Amazon Redshift console	List			
ListSchemas [permission only]	Grants permission to list schemas through the Amazon Redshift console	List			
ListTables [permission only]	Grants permission to list tables through the Amazon Redshift console	List			
ModifyCluster	Grants permission to modify the settings of a cluster	Write	cluster* (p. 1490)		
ModifyClusterDbParameter	Grants permission to modify the database revision of a cluster	Write	cluster* (p. 1490)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ModifyClusterIamRoles	Grants permission to modify the list of AWS Identity and Access Management (IAM) roles that can be used by a cluster to access other AWS services	Permissions management	cluster* (p. 1490)		
ModifyClusterMaintenance	Grants permission to modify the maintenance settings of a cluster	Write			
ModifyClusterParameters	Grants permission to modify the parameters of a parameter group	Write	parametergroup* (p. 1490)		
ModifyClusterSnapshotSettings	Grants permission to modify the settings of a snapshot	Write	snapshot* (p. 1490)		
ModifyClusterSnapshotSchedule	Grants permission to modify a snapshot schedule for a cluster	Write	cluster* (p. 1490)		
ModifyClusterSubnetGroup	Grants permission to modify a subnet group to include the specified list of VPC subnets	Write	subnetgroup* (p. 1490)		
ModifyEventSubscription	Grants permission to modify an existing Amazon Redshift event notification subscription	Write	eventsSubscription* (p. 1490)		
ModifySavedQuery [permission only]	Grants permission to modify an existing saved query through the Amazon Redshift console	Write			
ModifyScheduledAction	Grants permission to modify an existing Amazon Redshift scheduled action	Write			
ModifySnapshotCopyRequest	Grants permission to modify the number of days to retain snapshots in the destination AWS Region after they are copied from the source AWS Region	Write	cluster* (p. 1490)		
ModifySnapshotSchedule	Grants permission to modify a snapshot schedule	Write	snapshotSchedule* (p. 1490)		
PurchaseReservedNodeOfferings	Grants permission to purchase a reserved node	Write			
RebootCluster	Grants permission to reboot a cluster	Write	cluster* (p. 1490)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ResetClusterParameters <small>[more parameters]</small>	Grants permission to set one or more parameters of a parameter group to their default values and set the source values of the parameters to "engine-default"	Write	parametergroup* (p. 1490)		
ResizeCluster	Grants permission to change the size of a cluster	Write	cluster* (p. 1490)		
RestoreFromClusterSnapshot	Grants permission to create a cluster from a snapshot	Write	snapshot* (p. 1490)		
RestoreTableFromClusterSnapshot	Grants permission to create a table from a snapshot in an Amazon Redshift cluster snapshot	Write	cluster* (p. 1490)		
			snapshot* (p. 1490)		
RevokeClusterSecurityGroupIngress	Grants permission to revoke security group ingress from an Amazon Redshift security group for a previously authorized IP range or Amazon EC2 security group	Permissions management	securitygroup* (p. 1490)		
			securitygroupingress-ec2securitygroup* (p. 1490)		
RevokeSnapshotAccess	Grants permission to revoke access from the specified AWS account to restore a snapshot	Permissions management	snapshot* (p. 1490)		
RotateEncryptionKey	Grants permission to rotate an encryption key for a cluster	Permissions management	cluster* (p. 1490)		
ViewQueriesFromResults [permission only]	Grants permission to view query results through the Amazon Redshift console	List			
ViewQueriesInConsole [permission only]	Grants permission to terminate running queries and loads through the Amazon Redshift console	List			

Resource Types Defined by Amazon Redshift

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1481\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
cluster	arn:\${Partition}:redshift:\${Region}: \${Account}:cluster:\${ClusterName}	
dbgroup	arn:\${Partition}:redshift:\${Region}: \${Account}:dbgroup:\${ClusterName}/\${DbGroup}	
dbname	arn:\${Partition}:redshift:\${Region}: \${Account}:dbname:\${ClusterName}/\${DbName}	
dbuser	arn:\${Partition}:redshift:\${Region}: \${Account}:dbuser:\${ClusterName}/\${DbUser}	
eventsSubscription	arn:\${Partition}:redshift:\${Region}: \${Account}:eventsSubscription: \${EventSubscriptionName}	
hsmclientcertificate	arn:\${Partition}:redshift:\${Region}: \${Account}:hsmclientcertificate: \${HSMClientCertificateId}	
hsmconfiguration	arn:\${Partition}:redshift:\${Region}: \${Account}:hsmconfiguration: \${HSMConfigurationId}	
parametergroup	arn:\${Partition}:redshift: \${Region}:\${Account}:parametergroup: \${ParameterGroupName}	
securitygroup	arn:\${Partition}:redshift: \${Region}:\${Account}:securitygroup: \${SecurityGroupName}/ec2securitygroup/ \${Owner}/\${Ec2SecurityGroupId}	
securitygroupingresscidr	arn:\${Partition}:redshift:\${Region}: \${Account}:securitygroupingress: \${SecurityGroupName}/cidrip/\${IpRange}	
securitygroupingressec2securitygroup	arn:\${Partition}:redshift:\${Region}: \${Account}:securitygroupingress: \${SecurityGroupName}/ec2securitygroup/ \${Owner}/\${Ece2SecuritygroupId}	
snapshot	arn:\${Partition}:redshift:\${Region}: \${Account}:snapshot:\${ClusterName}/ \${SnapshotName}	
snapshotcopygrant	arn:\${Partition}:redshift:\${Region}: \${Account}:snapshotcopygrant: \${SnapshotCopyGrantName}	
snapshotschedule	arn:\${Partition}:redshift:\${Region}: \${Account}:snapshotschedule: \${ParameterGroupName}	
subnetgroup	arn:\${Partition}:redshift:\${Region}: \${Account}:subnetgroup:\${SubnetGroupName}	

Condition Keys for Amazon Redshift

Amazon Redshift defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
<code>redshift:DbName</code>	Filters access by the database name	String
<code>redshift:DbUser</code>	Filters access by the database user name	String
<code>redshift:DurationSeconds</code>	Filters access by the number of seconds until a temporary credential set expires	String

Actions, Resources, and Condition Keys for Amazon Rekognition

Amazon Rekognition (service prefix: `rekognition`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Rekognition \(p. 1491\)](#)
- [Resource Types Defined by Amazon Rekognition \(p. 1495\)](#)
- [Condition Keys for Amazon Rekognition \(p. 1495\)](#)

Actions Defined by Amazon Rekognition

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CompareFaces	Compares a face in source input image with each face detected in the target input image.	Read			
CreateCollection	Creates a collection in an AWS region. You can then add faces to the collection using the IndexFaces API.	Write	collection* (p. 1495)		
CreateProject	Creates a new Amazon Rekognition Custom Labels project.	Write	project* (p. 1495)		
CreateProjectVersion	Creates a new version of a model and begins training.	Write	project* (p. 1495)		
			projectversion* (p. 1495)		
CreateStreamProcessor	Creates an Amazon Rekognition stream processor that you can use to detect and recognize faces in a streaming video.	Write	collection* (p. 1495)		
			streamprocessor* (p. 1495)		
DeleteCollection	Deletes the specified collection. Note that this operation removes all faces in the collection.	Write	collection* (p. 1495)		
DeleteFaces	Deletes faces from a collection.	Write	collection* (p. 1495)		
DeleteStreamProcessor	Deletes the stream processor identified by Name.	Write	streamprocessor* (p. 1495)		
DescribeCollection	Describes the specified collection.	Read	collection* (p. 1495)		
DescribeProjectVersion	Lists and describes the model versions in an Amazon Rekognition Custom Labels project.	Read	project* (p. 1495)		
DescribeProjects	Lists and gets information about your Amazon Rekognition Custom Labels projects.	Read			
DescribeStreamProcessor	Provides information about a stream processor created by CreateStreamProcessor.	Read	streamprocessor* (p. 1495)		
DetectCustomLabel	Detects custom labels in a supplied image by using an	Read	projectversion* (p. 1495)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
	Amazon Rekognition Custom Labels model version.				
DetectFaces	Detects human faces within an image (JPEG or PNG) provided as input.	Read			
DetectLabels	Detects instances of real-world labels within an image (JPEG or PNG) provided as input.	Read			
DetectModerationLabels	Detects moderation labels within input image.	Read			
DetectText	Detects text in the input image and converts it into machine-readable text.	Read			
GetCelebrityInfo	Gets the name and additional information about a celebrity based on his or her Rekognition ID.	Read			
GetCelebrityRecognitionResults	Gets the celebrity recognition results for a Rekognition Video analysis started by StartCelebrityRecognition.	Read			
GetContentModerationResults	Gets the content moderation analysis results for a Rekognition Video analysis started by StartContentModeration.	Read			
GetFaceDetection	Gets face detection results for a Rekognition Video analysis started by StartFaceDetection.	Read			
GetFaceSearch	Gets the face search results for Rekognition Video face search started by StartFaceSearch.	Read			
GetLabelDetectionResults	Gets the label detection results of a Rekognition Video analysis started by StartLabelDetection.	Read			
GetPersonTracking	Gets information about people detected within a video.	Read			
IndexFaces	Detects faces in the input image and adds them to the specified collection.	Write	collection* (p. 1495)		
ListCollections	Returns a list of collection IDs in your account.	Read	collection* (p. 1495)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListFaces	Returns metadata for faces in the specified collection.	Read	collection* (p. 1495)		
ListStreamProcessors	Gets a list of stream processors that you have created with CreateStreamProcessor.	List	streamprocessor* (p. 1495)		
RecognizeCelebrities	Returns an array of celebrities recognized in the input image.	Read			
SearchFaces	For a given input face ID, searches the specified collection for matching faces.	Read	collection* (p. 1495)		
SearchFacesByImage	For a given input image, first detects the largest face in the image, and then searches the specified collection for matching faces.	Read	collection* (p. 1495)		
StartCelebrityRecognition	Starts asynchronous recognition of celebrities in a video.	Write			
StartContentModeration	Starts asynchronous detection of explicit or suggestive adult content in a video.	Write			
StartFaceDetection	Starts asynchronous detection of faces in a video.	Write			
StartFaceSearch	Starts the asynchronous search for faces in a collection that match the faces of persons detected in a video.	Write	collection* (p. 1495)		
StartLabelDetection	Starts asynchronous detection of labels in a video.	Write			
StartPersonTracking	Starts the asynchronous tracking of persons in a video.	Write			
StartProjectVersion	Starts the deployment of a model version.	Write	projectversion* (p. 1495)		
StartStreamProcessor	Starts processing a stream processor.	Write	streamprocessor* (p. 1495)		
StopProjectVersion	Stops a deployed model version.	Write	projectversion* (p. 1495)		
StopStreamProcessor	Stops a running stream processor that was created by CreateStreamProcessor.	Write	streamprocessor* (p. 1495)		

Resource Types Defined by Amazon Rekognition

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 1491\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
collection	arn:\${Partition}:rekognition:\${Region}: \${Account}:collection/\${CollectionId}	
streamprocessor	arn:\${Partition}:rekognition: \${Region}: \${Account}:streamprocessor/ \${StreamprocessorId}	
project	arn:\${Partition}:rekognition:\${Region}: \${Account}:project/\${ProjectName}/ \${CreationTimestamp}	
projectversion	arn:\${Partition}:rekognition:\${Region}: \${Account}:project/\${ProjectName}/version/ \${VersionName}/\${CreationTimestamp}	

Condition Keys for Amazon Rekognition

Rekognition has no service-specific context keys that can be used in the Condition element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for AWS Resource Access Manager

AWS Resource Access Manager (service prefix: `ram`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Resource Access Manager \(p. 1495\)](#)
- [Resource Types Defined by AWS Resource Access Manager \(p. 1500\)](#)
- [Condition Keys for AWS Resource Access Manager \(p. 1501\)](#)

Actions Defined by AWS Resource Access Manager

You can specify the following actions in the Action element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually

allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AcceptResourceShareInvitation	Accept the specified resource share invitation	Write	resource-share-invitation* (p. 1501)		
				ram:ShareOwnerAccountId (p. 1502)	
AssociateResourcePrincipals	Associates resource(s) and/or principal(s) to a resource share	Write	resource-share* (p. 1500)		
AssociateResourceSharePermissions	Associate a Permission with a Resource Share	Write	permission* (p. 1501)		
			resource-share* (p. 1500)		
				aws:ResourceTag/ {\$TagKey} (p. 1501)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				ram:AllowsExternalPrincipals (p. 1501)	ram:ResourceShareName (p. 1502)
	CreateResourceShare Create resource share with provided resource(s) and/or principal(s)	Write		aws:RequestTag/ \${TagKey} (p. 1501)	ram:RequestedResourceType (p. 1501)
	DeleteResourceShare Delete resource share	Write	resource-share* (p. 1500)		
	DisassociateResourcePrincipals Disassociates resource(s) and/or principal(s) from a resource share	Write	resource-share* (p. 1500)		ram:AllowsExternalPrincipals (p. 1501)

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:ResourceTag/\${TagKey} (p. 1501) ram:ResourceShareName (p. 1502) ram:AllowsExternalPrincipals (p. 1501) ram:Principal (p. 1501) ram:RequestedResourceType (p. 1501) ram:ResourceArn (p. 1502)	
DisassociateResourceShare	Disassociate a Permission from a Resource Share	Write	permission* (p. 1501)		
			resource-share* (p. 1500)		
				aws:ResourceTag/\${TagKey} (p. 1501) ram:AllowsExternalPrincipals (p. 1501) ram:ResourceShareName (p. 1502) ram:PermissionArn (p. 1501)	
EnableSharingWithCustomerOrganization	Grants permission to access customer organization and create a SLR in the customer's account	Write			
GetPermission	Gets the contents of an AWS RAM permission	Read	permission* (p. 1501)		
				ram:PermissionArn (p. 1501)	
GetResourcePolicies	Gets the policies for the specified resources that you own and have shared	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetResourceShares	Get a set of resource share associations from a provided list or with a specified status of the specified type	Read			
GetResourceSharesByInvitationArn	Get resource share invitations by the specified invitation arn or those for the resource share	Read			
GetResourceShares	Get a set of resource shares from a provided list or with a specified status	Read			
ListPendingInvitations	Lists the resources in a resource share that is shared with you but that the invitation is still pending for	Read	resource-share-invitation* (p. 1501)		
ListPermissions	Lists the AWS RAM permissions	List			
ListPrincipals	Lists the principals that you have shared resources with or that have shared resources with you	List			
ListResourceShares	List the Permissions associated with a Resource Share	List	resource-share* (p. 1500)		
				aws:ResourceTag/\${TagKey} (p. 1501) ram:ResourceShareName (p. 1502) ram:AllowsExternalPrincipals (p. 1501)	
ListResources	Lists the resources that you added to a resource shares or the resources that are shared with you	List			
RejectResourceShareInvitation	Reject the specified resource share invitation	Write	resource-share-invitation* (p. 1501)		
TagResource	Tag the specified resources share	Write	resource-share* (p. 1500)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:RequestTag/ \${TagKey} (p. 1501) aws:TagKeys (p. 1501)	
UntagResource	Untag the specified resource share	Write	resource-share* (p. 1500)		
				aws:RequestTag/ \${TagKey} (p. 1501) aws:TagKeys (p. 1501)	
UpdateResourceShare	Update attributes of the resource share	Write	resource-share* (p. 1500)		
				aws:ResourceTag/ \${TagKey} (p. 1501) ram:ResourceShareName (p. 1502) ram:AllowsExternalPrincipals (p. 1501) ram:RequestedAllowsExternalPrincipals (p. 1501)	

Resource Types Defined by AWS Resource Access Manager

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1495\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
resource-share	arn:\${Partition}:ram:\${Region}: \${Account}:resource-share/\${ResourcePath}	aws:ResourceTag/ \${TagKey} (p. 1501) ram:AllowsExternalPrincipals (p. 1501)

Resource Types	ARN	Condition Keys
		ram:ResourceShareName (p. 1502)
resource-share-invitation	arn:\${Partition}:ram:\${Region}: \${Account}:resource-share-invitation/ \${ResourcePath}	
permission	arn:\${Partition}:ram::\${Account}:permission/ \${ResourcePath}	ram:PermissionArn (p. 1501)

Condition Keys for AWS Resource Access Manager

AWS Resource Access Manager defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Specifies a tag key and value pair that must be used when creating or tagging a resource share. If users don't pass these specific tags, or if they don't specify tags at all, the request fails.	String
aws:ResourceTag/\${TagKey}	Indicates that the action can only be performed on resources that have the specified tag key and value pair.	String
aws:TagKeys	Specifies the tag keys that can be used when creating or tagging a resource share	String
ram:AllowsExternalPrincipals	Indicates that the action can only be performed on resource shares that allow or deny sharing with external principals. For example, specify true if the action can only be performed on resource shares that allow sharing with external principals. External principals are AWS accounts that are outside of its AWS organization	Bool
ram:PermissionArn	Indicates that the action can only be performed on a resource using the specified Permission ARN.	Arn
ram:Principal	Principals with the specified format can be associated to or disassociated from a resource share	String
ram:RequestedAllowsExternalPrincipals	The request must have the specified value for 'AllowsExternalPrincipals'. External principals are AWS accounts that are outside of its AWS Organization	Bool
ram:RequestedResourceType	Indicates that the action can only be performed on the specified resource type	String

Condition Keys	Description	Type
ram:ResourceArn	Indicates that the action can only be performed on a resource with the specified ARN.	Arn
ram:ResourceShareName	Indicates that the action can only be performed on a resource share with the specified name.	String
ram:ShareOwnerAccountId	Indicates that the action can only be performed on resource shares owned by a specific account. For example, you can use this condition key to specify which resource share invitations can be accepted or rejected based on the resource share owner's account ID.	String

Actions, Resources, and Condition Keys for Amazon Resource Group Tagging API

Amazon Resource Group Tagging API (service prefix: tag) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Resource Group Tagging API \(p. 1502\)](#)
- [Resource Types Defined by Amazon Resource Group Tagging API \(p. 1503\)](#)
- [Condition Keys for Amazon Resource Group Tagging API \(p. 1503\)](#)

Actions Defined by Amazon Resource Group Tagging API

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeReportCreation	Describe the status of the StartReportCreation operation.	Read			
GetComplianceSummary	Get a table that shows counts of resources that are noncompliant with their effective tag policies.	Read			
GetResources	Get tagged AWS resources that match the given tag filters	Read			
GetTagKeys	Get all tagKeys for the account in the specific region	Read			
GetTagValues	Get all tagValues for the account in the specific region	Read			
StartReportCreation	Generate a report that lists all tagged resources in accounts across your organization, and whether each resource is compliant with the effective tag policy.	Write			
TagResources	Add tags to AWS resources	Tagging			
UntagResources	Remove tags from AWS resources	Tagging			

Resource Types Defined by Amazon Resource Group Tagging API

Amazon Resource Group Tagging API does not support specifying a resource ARN in the `Resource` element of an IAM policy statement. To allow access to Amazon Resource Group Tagging API, specify `"Resource" : "*"` in your policy.

Condition Keys for Amazon Resource Group Tagging API

Resource Group Tagging has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for AWS Resource Groups

AWS Resource Groups (service prefix: `resource-groups`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Resource Groups \(p. 1504\)](#)
- [Resource Types Defined by AWS Resource Groups \(p. 1505\)](#)
- [Condition Keys for AWS Resource Groups \(p. 1505\)](#)

Actions Defined by AWS Resource Groups

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateGroup	Creates a group with a specified name, description, and resource query.	Tagging	group* (p. 1505)		
				aws:RequestTag/\${TagKey} (p. 1506) aws:TagKeys (p. 1506)	
DeleteGroup	Deletes a specified resource group	Write	group* (p. 1505)		
GetGroup	Gets information of a specified resource group	Read	group* (p. 1505)		
GetGroupQuery	Gets the query associated with a specified resource group	Read	group* (p. 1505)		
GetTags	Gets the tags associated with a specified resource group	Read	group* (p. 1505)		
ListGroupResources	Lists the resources that are member of a specified resource group	List	group* (p. 1505)		
ListGroups	Lists all resource groups	List	group* (p. 1505)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
SearchResources	Returns a list of AWS resource identifiers matching the given query	List			
Tag	Tags a specified resource group	Tagging	group* (p. 1505)		
				aws:RequestTag/ \${TagKey} (p. 1506) aws:TagKeys (p. 1506)	
Untag	Removes tags associated with a specified resource group	Tagging	group* (p. 1505)		
				aws:TagKeys (p. 1506)	
UpdateGroup	Updates a specified resource group	Write	group* (p. 1505)		
UpdateGroupQuery	Updates the query associated with a specified resource group	Write	group* (p. 1505)		

Resource Types Defined by AWS Resource Groups

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1504\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
group	<code>arn:\${Partition}:resource-groups:\${Region}:\${Account}:group/\${GroupName}</code>	aws:ResourceTag/ \${TagKey} (p. 1506)

Condition Keys for AWS Resource Groups

AWS Resource Groups defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
<code>aws:RequestTag/\${TagKey}</code>	Filters actions based on the presence of tag key-value pairs in the request	String
<code>aws:ResourceTag/\${TagKey}</code>	Filters actions based on tag key-value pairs attached to the resource	String
<code>aws:TagKeys</code>	Filters actions based on the presence of tag keys in the request	String

Actions, Resources, and Condition Keys for AWS RoboMaker

AWS RoboMaker (service prefix: `robomaker`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS RoboMaker \(p. 1506\)](#)
- [Resource Types Defined by AWS RoboMaker \(p. 1510\)](#)
- [Condition Keys for AWS RoboMaker \(p. 1511\)](#)

Actions Defined by AWS RoboMaker

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
BatchDescribeSimulationJobs	Describe multiple simulation jobs	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CancelDeploymentJob	Cancel a deployment job	Write	deploymentJob* (p. 1510)		
CancelSimulationJob	Cancel a simulation job	Write	simulationJob* (p. 1510)		
CreateDeploymentJob	Create a deployment job	Write		aws:TagKeys iam:CreateServiceLinkedRole* (p. 1511) aws:RequestTag/ \${TagKey} (p. 1511)	
CreateFleet	Create a deployment fleet that represents a logical group of robots running the same robot application	Write		aws:TagKeys (p. 1511) aws:RequestTag/ \${TagKey} (p. 1511)	
CreateRobot	Create a robot that can be registered to a fleet	Write		aws:TagKeys iam:CreateServiceLinkedRole* (p. 1511) aws:RequestTag/ \${TagKey} (p. 1511)	
CreateRobotApplication	Create a robot application	Write		aws:TagKeys (p. 1511) aws:RequestTag/ \${TagKey} (p. 1511)	
CreateRobotApplicationSnapshot	Create a snapshot of a robot application	Write	robotApplication* (p. 1510)		s3:GetObject
CreateSimulationApplication	Create a simulation application	Write		aws:TagKeys (p. 1511) aws:RequestTag/ \${TagKey} (p. 1511)	
CreateSimulationApplicationSnapshot	Create a snapshot of a simulation application	Write	simulationApplication* (p. 1510)		s3:GetObject
CreateSimulationJob	Create a simulation job	Write		aws:TagKeys iam:CreateServiceLinkedRole* (p. 1511) aws:RequestTag/ \${TagKey} (p. 1511)	
DeleteFleet	Delete a deployment fleet	Write	deploymentFleet* (p. 1510)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteRobot	Delete a robot	Write	robot* (p. 1510)		
DeleteRobotApplication	Delete a robot application	Write	robotApplication* (p. 1510)		
DeleteSimulationApplication	Delete a simulation application	Write	simulationApplication* (p. 1510)		
DeregisterRobot	Deregister a robot from a fleet	Write	deploymentFleet* (p. 1510)		
			robot* (p. 1510)		
DescribeDeploymentJob	Describe a deployment job	Read	deploymentJob* (p. 1510)		
DescribeFleet	Describe a deployment fleet	Read	deploymentFleet* (p. 1510)		
DescribeRobot	Describe a robot	Read	robot* (p. 1510)		
DescribeRobotApplication	Describe a robot application	Read	robotApplication* (p. 1510)		
DescribeSimulationApplication	Describe a simulation application	Read	simulationApplication* (p. 1510)		
DescribeSimulationJob	Describe a simulation job	Read	simulationJob* (p. 1510)		
ListDeploymentJobs	List deployment jobs	List			
ListFleets	List fleets	List			
ListRobotApplications	List robot applications	List			
ListRobots	List robots	List			
ListSimulationApplications	List simulation applications	List			
ListSimulationJobs	List simulation jobs	List			
ListTagsForResource	List tags for a RoboMaker resource.	List	deploymentFleet (p. 1510)		
			deploymentJob (p. 1510)		
			robot (p. 1510)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			robotApplication (p. 1510)		
			simulationApplication (p. 1510)		
			simulationJob (p. 1510)		
RegisterRobot	Register a robot to a fleet	Write	deploymentFleet* (p. 1510)		
	robot* (p. 1510)				
RestartSimulationJob	Restart a running simulation job	Write	simulationJob* (p. 1510)		
SyncDeploymentJobs	Ensures the most recently deployed robot application is deployed to all robots in the fleet	Write	deploymentFleet* (p. 1510)		iam:CreateServiceLinkedRole
TagResource	Add tags to a RoboMaker resource	Write	deploymentFleet (p. 1510)		
	deploymentJob (p. 1510)				
	robot (p. 1510)				
	robotApplication (p. 1510)				
	simulationApplication (p. 1510)				
	simulationJob (p. 1510)				
			aws:TagKeys (p. 1511) aws:RequestTag/ {\$TagKey} (p. 1511)		
UntagResource	Remove tags from a RoboMaker resource	Write	deploymentFleet (p. 1510)		
	deploymentJob (p. 1510)				
	robot (p. 1510)				

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			robotApplication (p. 1510)		
			simulationApplication (p. 1510)		
			simulationJob (p. 1510)		
				aws:TagKeys (p. 1511)	
UpdateRobotApplication	Update a robot application	Write	robotApplication*(p. 1510)		
UpdateSimulationApplication	Update a simulation application	Write	simulationApplication*(p. 1510)		

Resource Types Defined by AWS RoboMaker

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1506\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
robotApplication	arn:\${Partition}:robomaker:\${Region}: \${Account}:robot-application/ \${ApplicationName}/\${CreatedOnEpoch}	aws:ResourceTag/ \${TagKey} (p. 1511)
simulationApplication	arn:\${Partition}:robomaker:\${Region}: \${Account}:simulation-application/ \${ApplicationName}/\${CreatedOnEpoch}	aws:ResourceTag/ \${TagKey} (p. 1511)
simulationJob	arn:\${Partition}:robomaker:\${Region}: \${Account}:simulation-job/\${SimulationJobId}	aws:ResourceTag/ \${TagKey} (p. 1511)
deploymentJob	arn:\${Partition}:robomaker:\${Region}: \${Account}:deployment-job/\${DeploymentJobId}	aws:ResourceTag/ \${TagKey} (p. 1511)
robot	arn:\${Partition}:robomaker:\${Region}: \${Account}:robot/\${RobotName}/ \${CreatedOnEpoch}	aws:ResourceTag/ \${TagKey} (p. 1511)
deploymentFleet	arn:\${Partition}:robomaker:\${Region}: \${Account}:deployment-fleet/\${FleetName}/ \${CreatedOnEpoch}	aws:ResourceTag/ \${TagKey} (p. 1511)

Condition Keys for AWS RoboMaker

AWS RoboMaker defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
<code>aws:RequestTag/\${TagKey}</code>		String
<code>aws:ResourceTag/\${TagKey}</code>		String
<code>aws:TagKeys</code>		String

Actions, Resources, and Condition Keys for Amazon Route 53

Amazon Route 53 (service prefix: `route53`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Route 53 \(p. 1511\)](#)
- [Resource Types Defined by Amazon Route 53 \(p. 1517\)](#)
- [Condition Keys for Amazon Route 53 \(p. 1518\)](#)

Actions Defined by Amazon Route 53

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AssociateVPCWithHostedZone	Grants permission to associate an additional Amazon VPC with a private hosted zone	Write	hostedzone* (p. 1518)		
ChangeResourceRecordSets	Grants permission to create, update, or delete a record, which contains authoritative DNS information for a specified domain or subdomain name	Write	hostedzone* (p. 1518)		
ChangeTagsForResource	Grants permission to add, edit, or delete tags for a health check or a hosted zone	Tagging	healthcheck* (p. 1518)		
			hostedzone* (p. 1518)		
CreateHealthCheck	Grants permission to create a new health check, which monitors the health and performance of your web applications, web servers, and other resources	Write			
CreateHostedZone	Grants permission to create a public hosted zone, which you use to specify how the Domain Name System (DNS) routes traffic on the Internet for a domain, such as example.com, and its subdomains	Write			
CreateQueryLoggingConfiguration	Grants permission to create a configuration for DNS query logging	Write	hostedzone* (p. 1518)		
CreateReusableDelegationSet	Grants permission to create a delegation set (a group of four name servers) that can be reused by multiple hosted zones	Write			
CreateTrafficPolicy	Grants permission to create a traffic policy, which you use to create multiple DNS records for one domain name (such as example.com) or one subdomain name (such as www.example.com)	Write			
CreateTrafficPolicyVersion	Grants permission to create records in a specified hosted zone based on the settings in a specified traffic policy version	Write	hostedzone* (p. 1518)		
			trafficpolicy* (p. 1518)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateTrafficPolicy	Grants permission to create a new version of an existing traffic policy	Write	trafficpolicy* (p. 1518)		
CreateVPCAssociationAuthorization	Grants permission to authorize another AWS account that created a specified VPC to submit an AssociateVPCWithHostedZone request, which associates the VPC with a specified hosted zone that was created by a different account	Write	hostedzone* (p. 1518)		
DeleteHealthCheck	Grants permission to delete a health check	Write	healthcheck* (p. 1518)		
DeleteHostedZone	Grants permission to delete a hosted zone	Write	hostedzone* (p. 1518)		
DeleteQueryLoggingConfiguration	Grants permission to delete a configuration for DNS query logging	Write	queryloggingconfig* (p. 1518)		
DeleteReusableDelegationSet	Grants permission to delete a reusable delegation set	Write	delegationset* (p. 1518)		
DeleteTrafficPolicy	Grants permission to delete a traffic policy	Write	trafficpolicy* (p. 1518)		
DeleteTrafficPolicyInstance	Grants permission to delete a traffic policy instance and all the records that Route 53 created when you created the instance	Write	trafficpolicyinstance* (p. 1518)		
DeleteVPCAssociationAuthorization	Grants permission to remove authorization for associating an Amazon Virtual Private Cloud with a Route 53 private hosted zone	Write	hostedzone* (p. 1518)		
DisassociateVPCFromHostedZone	Grants permission to disassociate an Amazon Virtual Private Cloud from a Route 53 private hosted zone	Write			
	SCENARIO: Disassociate by the VPC owner		hostedzone* (p. 1518) hostedzone (p. 1518)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
	SCENARIO: Disassociate by the hosted zone owner		hostedzone* (p. 1518) hostedzone (p. 1518)		
GetAccountLimit	Grants permission to get the specified limit for the current account, for example, the maximum number of health checks that you can create using the account	Read			
GetChange	Grants permission to get the current status of a request to create, update, or delete one or more records	List	change* (p. 1517)		
GetCheckerIpRanges	Grants permission to get a list of the IP ranges that are used by Route 53 health checkers to check the health of your resources	List			
GetGeoLocation	Grants permission to get information about whether a specified geographic location is supported for Route 53 geolocation records	List			
GetHealthCheck	Grants permission to get information about a specified health check	Read	healthcheck* (p. 1518)		
GetHealthCheckCount	Grants permission to get the number of health checks that are associated with the current AWS account	List			
GetHealthCheckLastFailureReason	Grants permission to get the reason that a specified health check failed most recently	List	healthcheck* (p. 1518)		
GetHealthCheckStatus	Grants permission to get the status of a specified health check	List	healthcheck* (p. 1518)		
GetHostedZone	Grants permission to get information about a specified hosted zone including the four name servers that Route 53 assigned to the hosted zone	List	hostedzone* (p. 1518)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetHostedZoneCount	Grants permission to get the number of hosted zones that are associated with the current AWS account	List			
GetHostedZoneLimit	Grants permission to get the specified limit for a specified hosted zone	Read	hostedzone* (p. 1518)		
GetQueryLoggingInfo	Grants permission to get information about a specified configuration for DNS query logging	Read	queryloggingconfig* (p. 1518)		
GetReusableDelegationSetInformation	Grants permission to get information about a specified reusable delegation set, including the four name servers that are assigned to the delegation set	List	delegationset* (p. 1518)		
GetReusableDelegationSetLimit	Grants permission to get the maximum number of hosted zones that you can associate with the specified reusable delegation set	Read	delegationset* (p. 1518)		
GetTrafficPolicy	Grants permission to get information about a specified traffic policy version	Read	trafficpolicy* (p. 1518)		
GetTrafficPolicyInstanceInformation	Grants permission to get information about a specified traffic policy instance	Read	trafficpolicyinstance* (p. 1518)		
GetTrafficPolicyInstancesCount	Grants permission to get the number of traffic policy instances that are associated with the current AWS account	Read			
ListGeoLocations	Grants permission to get a list of geographic locations that Route 53 supports for geolocation	List			
ListHealthChecks	Grants permission to get a list of the health checks that are associated with the current AWS account	List			
ListHostedZones	Grants permission to get a list of the public and private hosted zones that are associated with the current AWS account	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListHostedZones	Grants permission to get a list of your hosted zones in lexicographic order. Hosted zones are sorted by name with the labels reversed, for example, com.example.www.	List			
ListQueryLoggingConfigurations	Grants permission to list the configurations for DNS query logging that are associated with the current AWS account or the configuration that is associated with a specified hosted zone.	List	queryloggingconfig* (p. 1518)		
ListResourceRecords	Grants permission to list the records in a specified hosted zone	List	hostedzone* (p. 1518)		
ListReusableDelegationSets	Grants permission to list the delegation sets that are associated with the current AWS account.	List			
ListTagsForResource	Grants permission to list tags for one health check or hosted zone	Read	healthcheck (p. 1518)		
			hostedzone (p. 1518)		
ListTagsForResources	Grants permission to list tags for up to 10 health checks or hosted zones	Read	healthcheck (p. 1518)		
			hostedzone (p. 1518)		
ListTrafficPolicies	Grants permission to get information about the latest version for every traffic policy that is associated with the current AWS account. Policies are listed in the order in which they were created.	Read			
ListTrafficPolicyInstances	Grants permission to get information about the traffic policy instances that you created by using the current AWS account	Read			
ListTrafficPolicyInstancesInHostedZone	Grants permission to get information about the traffic policy instances that you created in a specified hosted zone	Read	hostedzone* (p. 1518)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListTrafficPolicyInformation	Grants permission to get information about the traffic policy instances that you created using a specified traffic policy version	Read	trafficpolicy* (p. 1518)		
ListTrafficPolicyVersions	Grants permission to get information about all the versions for a specified traffic policy	Read	trafficpolicy* (p. 1518)		
ListVPCAssociations	Grants permission to get a list of the VPCs that were created by other accounts and that can be associated with a specified hosted zone	Read	hostedzone* (p. 1518)		
TestDNSAnswer	Grants permission to get the value that Route 53 returns in response to a DNS query for a specified record name and type	Read			
UpdateHealthCheck	Grants permission to update an existing health check	Write	healthcheck* (p. 1518)		
UpdateHostedZoneComment	Grants permission to update the comment for a specified hosted zone	Write	hostedzone* (p. 1518)		
UpdateTrafficPolicyComment	Grants permission to update the comment for a specified traffic policy version	Write	trafficpolicy* (p. 1518)		
UpdateTrafficPolicyRecords	Grants permission to update the records in a specified hosted zone that were created based on the settings in a specified traffic policy version	Write	trafficpolicyinstance* (p. 1518)		

Resource Types Defined by Amazon Route 53

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1511\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
change	<code>arn:\${Partition}:route53:::change/\${Id}</code>	

Resource Types	ARN	Condition Keys
delegationset	arn:\${Partition}:route53::::delegationset/\${Id}	
healthcheck	arn:\${Partition}:route53::::healthcheck/\${Id}	
hostedzone	arn:\${Partition}:route53::::hostedzone/\${Id}	
trafficpolicy	arn:\${Partition}:route53::::trafficpolicy/\${Id}	
trafficpolicyinstance	arn:\${Partition}:route53::::trafficpolicyinstance/\${Id}	
queryloggingconfig	arn:\${Partition}:route53::::queryloggingconfig/\${Id}	

Condition Keys for Amazon Route 53

Route 53 has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Amazon Route 53 Resolver

Amazon Route 53 Resolver (service prefix: `route53resolver`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Route 53 Resolver \(p. 1518\)](#)
- [Resource Types Defined by Amazon Route 53 Resolver \(p. 1521\)](#)
- [Condition Keys for Amazon Route 53 Resolver \(p. 1522\)](#)

Actions Defined by Amazon Route 53 Resolver

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your

policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AssociateResolverEndpointWithAddress	Grants permission to associate a specified IP address with a resolver endpoint. This is an IP address that DNS queries pass through on the way to your network (outbound) or your VPCs (inbound).	Write	resolver-endpoint* (p. 1521)		
AssociateResolverRuleWithVpc	Grants permission to associate a specified resolver rule with a specified VPC.	Write	resolver-rule* (p. 1521)		
CreateResolverEndpoint	Grants permission to create a resolver endpoint. There are two types of resolver endpoints, inbound and outbound.	Write	resolver-endpoint* (p. 1521)		
CreateResolverRule	For DNS queries that originate in your VPC, grants permission to define how to route the queries out of the VPC.	Write	resolver-rule* (p. 1521)		
DeleteResolverEndpoint	Grants permission to delete a resolver endpoint. The effect of deleting a resolver endpoint depends on whether it's an inbound or an outbound resolver endpoint.	Write	resolver-endpoint* (p. 1521)		
DeleteResolverRule	Grants permission to delete a resolver rule.	Write	resolver-rule* (p. 1521)		
DisassociateResolverEndpointWithAddress	Grants permission to remove a specified IP address from a resolver endpoint. This is an IP address that DNS queries pass through on the way to your network (outbound) or your VPCs (inbound).	Write	resolver-endpoint* (p. 1521)		
DisassociateResolverRuleWithVpc	Grants permission to remove the association between a specified resolver rule and a specified VPC.	Write	resolver-rule* (p. 1521)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetResolverEndpoint	Grants permission to get information about a specified resolver endpoint, such as whether it's an inbound or an outbound resolver endpoint, and the IP addresses in your VPC that DNS queries are forwarded to on the way into or out of your VPC.	Read	resolver-endpoint* (p. 1521)		
GetResolverRule	Grants permission to get information about a specified resolver rule, such as the domain name that the rule forwards DNS queries for and the IP address that queries are forwarded to.	Read	resolver-rule* (p. 1521)		
GetResolverRuleAssociation	Grants permission to get information about an association between a specified resolver rule and a VPC.	Read	resolver-rule* (p. 1521)		
GetResolverRulePolicy	Grants permission to get information about a resolver rule policy.	Read	resolver-rule* (p. 1521)		
ListResolverEndpointAssociations	For a specified resolver endpoint, grants permission to list the IP addresses that DNS queries pass through on the way to your network (outbound) or your VPCs (inbound).	List	resolver-endpoint* (p. 1521)		
ListResolverEndpoints	Grants permission to list all the resolver endpoints that were created using the current AWS account.	List	resolver-endpoint* (p. 1521)		
ListResolverRuleAssociations	Grants permission to list the associations that were created between resolver rules and VPCs using the current AWS account.	List	resolver-rule* (p. 1521)		
ListResolverRules	Grants permission to list the resolver rules that were created using the current AWS account.	List	resolver-rule* (p. 1521)		
ListTagsForResource	Grants permission to list the tags that you associated with the specified resource.	Read	resolver-endpoint (p. 1521)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			resolver-rule (p. 1521)		
PutResolverRule	Grants permission to specify the Resolver operations and resources that you want to allow another AWS account to use.	Write	resolver-rule* (p. 1521)		
TagResource	Grants permission to add one or more tags to a specified resource.	Tagging	resolver-endpoint (p. 1521)		
			resolver-rule (p. 1521)		
UntagResource	Grants permission to remove one or more tags from a specified resource.	Tagging	resolver-endpoint (p. 1521)		
			resolver-rule (p. 1521)		
UpdateResolverEndpoint	Grants permission to update selected settings for an inbound or an outbound resolver endpoint.	Write	resolver-endpoint* (p. 1521)		
UpdateResolverRule	Grants permission to update settings for a specified resolver rule.	Write	resolver-rule* (p. 1521)		

Resource Types Defined by Amazon Route 53 Resolver

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1518\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
resolver-rule	arn:\${Partition}:route53resolver:\${Region}: \${Account}:resolver-rule/\${ResourceId}	aws:ResourceTag/ \${TagKey} (p. 1522)
resolver-endpoint	arn:\${Partition}:route53resolver:\${Region}: \${Account}:resolver-endpoint/\${ResourceId}	aws:ResourceTag/ \${TagKey} (p. 1522)

Condition Keys for Amazon Route 53 Resolver

Amazon Route 53 Resolver defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
<code>aws:RequestTag/\${TagKey}</code>	Filters actions based on the presence of tag key-value pairs in the request	String
<code>aws:ResourceTag/\${TagKey}</code>	Filters actions based on tag key-value pairs attached to the resource	String
<code>aws:TagKeys</code>	Filters actions based on the presence of tag keys in the request	String

Actions, Resources, and Condition Keys for Amazon Route53 Domains

Amazon Route53 Domains (service prefix: `route53domains`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Route53 Domains \(p. 1522\)](#)
- [Resource Types Defined by Amazon Route53 Domains \(p. 1525\)](#)
- [Condition Keys for Amazon Route53 Domains \(p. 1525\)](#)

Actions Defined by Amazon Route53 Domains

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you

specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CheckDomainAvailability	Grants permission to check the availability of one domain name	Read			
DeleteTagsForDomain	Grants permission to delete the specified tags for a domain	Tagging			
DisableDomainAutoRenew	Grants permission to configure Amazon Route 53 to automatically renew the specified domain before the domain registration expires	Write			
DisableDomainTransfer	Grants permission to remove the transfer lock on the domain (specifically the clientTransferProhibited status) to allow domain transfers	Write			
EnableDomainAutoRenew	Grants permission to configure Amazon Route 53 to automatically renew the specified domain before the domain registration expires	Write			
EnableDomainTransfer	Grants permission to set the transfer lock on the domain (specifically the clientTransferProhibited status) to prevent domain transfers	Write			
GetContactReachabilityInformation	For operations that require confirmation that the email address for the registrant contact is valid, such as registering a new domain, grants permission to get information about whether the registrant contact has responded	Read			
GetDomainDetail	Grants permission to get detailed information about a domain	Read			
GetDomainSuggestions	Grants permission to get a list of suggested domain names given a string, which can either be a domain name or simply a word or phrase (without spaces)	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetOperationDetail	Grants permission to get the current status of an operation that is not completed	Read			
ListDomains	Grants permission to list all the domain names registered with Amazon Route 53 for the current AWS account	List			
ListOperations	Grants permission to list the operation IDs of operations that are not yet complete	List			
ListTagsForDomain	Grants permission to list all the tags that are associated with the specified domain	List			
RegisterDomain	Grants permission to register domains	Write			
RenewDomain	Grants permission to renew domains for the specified number of years	Write			
ResendContactReconfirmationEmail	For operations that require confirmation that the email address for the registrant contact is valid, such as registering a new domain, grants permission to resend the confirmation email to the current email address for the registrant contact	Write			
RetrieveDomainAuthCode	Grants permission to get the AuthCode for the domain	Write			
TransferDomain	Grants permission to transfer a domain from another registrar to Amazon Route 53	Write			
UpdateDomainContact	Grants permission to update the contact information for domain	Write			
UpdateDomainContactPrivacy	Grants permission to update the domain contact privacy setting	Write			
UpdateDomainNameServers	Grants permission to replace the current set of name servers for a domain with the specified set of name servers	Write			
UpdateTagsForDomain	Grants permission to add or update tags for a specified domain	Tagging			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ViewBilling	Grants permission to get all the domain-related billing records for the current AWS account for a specified period	Read			

Resource Types Defined by Amazon Route53 Domains

Amazon Route53 Domains does not support specifying a resource ARN in the Resource element of an IAM policy statement. To allow access to Amazon Route53 Domains, specify “Resource”: “*” in your policy.

Condition Keys for Amazon Route53 Domains

Route53 Domains has no service-specific context keys that can be used in the Condition element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Amazon S3

Amazon S3 (service prefix: s3) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon S3 \(p. 1525\)](#)
- [Resource Types Defined by Amazon S3 \(p. 1576\)](#)
- [Condition Keys for Amazon S3 \(p. 1576\)](#)

Actions Defined by Amazon S3

You can specify the following actions in the Action element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources (“*”) in the Resource element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
	Aborts a multipart upload. AbortMultipartUpload	Write	object* (p. 1576)	s3:DataAccessPointArn (p. 1577)	s3:DataAccessPointAccount (p. 1577)
	Allows circumvention of Retention -mode object retention settings	Permissions management	object* (p. 1576)	s3:DataAccessPointNetworkOrigin (p. 1577)	s3:authtype (p. 1577)

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				s3:x-amz-acl (p. 1578) s3:x-amz-content-sha256 (p. 1578) s3:x-amz-copy-source (p. 1578) s3:x-amz-grant-full-control (p. 1578) s3:x-amz-grant-read (p. 1578) s3:x-amz-grant-read-acp (p. 1578) s3:x-amz-grant-write (p. 1578) s3:x-amz-grant-write-acp (p. 1578) s3:x-amz-metadata-directive (p. 1578) s3:x-amz-server-side-encryption (p. 1578) s3:x-amz-server-side-encryption-aws-kms-	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				key-id (p. 1578) s3:x-amz-storage-class (p. 1579) s3:x-amz-website-redirect-location (p. 1579) s3:object-lock-mode (p. 1578) s3:object-lock-retain-until-date (p. 1578) s3:object-lock-remaining-retention-days (p. 1578) s3:object-lock-legal-hold (p. 1578)	
CreateAccessPoint	Creates a new access point.	Write	accesspoint* (p. 1576)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				s3:DataAccessPointAccount (p. 1577) s3:DataAccessPointArn (p. 1577) s3:AccessPointNetworkOrigin (p. 1577) s3:authtype (p. 1577) s3:locationconstraint (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-acl (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
CreateBucket	Creates a new bucket.	Write	bucket* (p. 1576)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				s3:authtype (p. 1577) s3:locationconstraint (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-acl (p. 1578) s3:x-amz-content-sha256 (p. 1578) s3:x-amz-grant-full-control (p. 1578) s3:x-amz-grant-read (p. 1578) s3:x-amz-grant-read-acp (p. 1578) s3:x-amz-grant-write (p. 1578) s3:x-amz-grant-write-acp (p. 1578)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateJob	Creates a new Amazon S3 Batch Operations job.	Write		s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578) s3:RequestJobPriority (p. 1577) s3:RequestJobOperation (p. 1577)	
DeleteAccessPoint	Deletes the access point named <code>AccessPointName</code> in the URI	Write	accesspoint* (p. 1576)	s3:DataAccessPointArn (p. 1577) s3:DataAccessPointAccount (p. 1577) s3:AccessPointNetworkOrigin (p. 1577) s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
DeleteAccessPointPolicy	Delete the policy on a specified access point	Permissions management	accesspoint* (p. 1576)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				s3:DataAccessPointArn (p. 1577) s3:DataAccessPointAccount (p. 1577) s3:AccessPointNetworkOrigin (p. 1577) s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
DeleteBucket	Deletes the bucket named in the URI	Write	bucket* (p. 1576)		
				s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
DeleteBucketPolicy	Delete the policy on a specified bucket	Permissions management	bucket* management (p. 1576)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
DeleteBucketWebsiteConfiguration	Removes the website configuration for a bucket.	Write	bucket* (p. 1576)		
				s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteObject	Removes the null version (if there is one) of an object and inserts a delete marker, which becomes the current version of the object.	Write	object* (p. 1576)		s3:DataAccessPointAccount (p. 1577) s3:DataAccessPointArn (p. 1577) s3:AccessPointNetworkOrigin (p. 1577) s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)
DeleteObjectTagging	This implementation of the <code>DELETE</code> operation uses the tagging subresource to remove the entire tag set from the specified object.	Tagging	object* (p. 1576)		s3:DataAccessPointAccount (p. 1577) s3:DataAccessPointArn (p. 1577) s3:AccessPointNetworkOrigin (p. 1577) s3:ExistingObjectTag/<key> (p. 1577) s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteObjectVersion	To remove a specific version of a object , you must be the bucket owner and you must use the <code>versionId</code> subresource.	Write	object* (p. 1576)		s3:DataAccessPointAccount (p. 1577) s3:DataAccessPointArn (p. 1577) s3:AccessPointNetworkOrigin (p. 1577) s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:versionid (p. 1578) s3:x-amz-content-sha256 (p. 1578)
DeleteObjectVersion	DELETE Object tagging (for a Specific Version of the Object)	Tagging	object* (p. 1576)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				s3:DataAccessPointAccount (p. 1577) s3:DataAccessPointArn (p. 1577) s3:AccessPointNetworkOrigin (p. 1577) s3:ExistingObjectTag/<key> (p. 1577) s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:versionid (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
DescribeJob	Retrieves the configuration parameters and status for an Amazon S3 batch operations job.	Read	job* (p. 1576)		
				s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetAccelerateConfiguration	This implementation of the GET <code>Configuration</code> operation uses the accelerate subresource to return the Transfer Acceleration state of a bucket, which is either Enabled or Suspended.	Read	bucket* (p. 1576)	s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
GetAccessPoint	Retrieve access point metadata	Read		s3:DataAccessPointAccount (p. 1577) s3:DataAccessPointArn (p. 1577) s3:AccessPointNetworkOrigin (p. 1577) s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
GetAccessPointPolicy	Return the policy of a specified access point.	Read	accesspoint* (p. 1576)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				s3:DataAccessPointAccount (p. 1577) s3:DataAccessPointArn (p. 1577) s3:AccessPointNetworkOrigin (p. 1577) s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
GetAccessPointPolicy	Retrieve the policy status for an specified access point's policy	Read	accesspoint* (p. 1576)	s3:DataAccessPointAccount (p. 1577) s3:DataAccessPointArn (p. 1577) s3:AccessPointNetworkOrigin (p. 1577) s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetAccountPublicAccessBlockConfiguration	Retrieve the PublicAccessBlock Configuration for an AWS account	Read		s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
GetAnalyticsConfiguration	This implementation of the GET operation returns an analytics configuration (identified by the analytics configuration ID) from the bucket.	Read	bucket* (p. 1576)		
				s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
GetBucketAcl	Return the access control list (ACL) of a bucket.	Read	bucket* (p. 1576)		
GetBucketCORS	Returns the CORS configuration information set for the bucket.	Read	bucket* (p. 1576)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
	Return a bucket's region. GetBucketLocation	Read	bucket* (p. 1576)		
GetBucketLogging	Return the logging status of a bucket and the permissions users have to view and modify that status.	Read	bucket* (p. 1576)		
				s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
GetBucketNotificationConfiguration	Return the notification configuration of a bucket.	Read	bucket* (p. 1576)		
				s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
GetBucketObjectLockConfiguration for a specific bucket	GET Object Lock configuration for a specific bucket	Read	bucket* (p. 1576)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578)	
GetBucketPolicy	Return the policy of a specified bucket.	Read	bucket* (p. 1576)		
				s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
GetBucketPolicyStatus	Retrieve the policy status for an specific S3 bucket, indicating whether the bucket is public.	Read	bucket* (p. 1576)		
				s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
GetBucketPublicAccessConfiguration	Retrieve the PublicAccessBlock configuration for a specific S3 bucket.	Read	bucket* (p. 1576)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
GetBucketRequestPaymentConfiguration	Return the request payment configuration of a bucket.	Read	bucket* (p. 1576)		
				s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
GetBucketTagging	Return the tag set associated with the bucket.	Read	bucket* (p. 1576)		
				s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
GetBucketVersioning	Return the versioning state of a bucket.	Read	bucket* (p. 1576)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
GetBucketWebsiteConfiguration	Returns the website configuration associated with a bucket.	Read	bucket* (p. 1576)		
				s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
GetEncryptionConfiguration	Returns the encryption configuration information set on the bucket.	Read	bucket* (p. 1576)		
				s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetInventoryConfiguration	This implementation of the GET operation returns an inventory configuration (identified by the inventory configuration ID) from the bucket.	Read	bucket* (p. 1576)		
		Read	bucket* (p. 1576)		
	Gets a metrics configuration for the CloudWatch request metrics (specified by the metrics configuration ID) from the bucket. Note that this doesn't include the daily storage metrics.	Read	bucket* (p. 1576)		
GetObject	Retrieves objects from Amazon S3.	Read	object* (p. 1576)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				s3:DataAccessPointAccount (p. 1577) s3:DataAccessPointArn (p. 1577) s3:AccessPointNetworkOrigin (p. 1577) s3:ExistingObjectTag/ <key> (p. 1577) s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
GetObjectAcl	Return the access control list (ACL) of an object.	Read	object* (p. 1576)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				s3:DataAccessPointAccount (p. 1577) s3:DataAccessPointArn (p. 1577) s3:AccessPointNetworkOrigin (p. 1577) s3:ExistingObjectTag/<key> (p. 1577) s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
GetObjectLegalHold	GET Object Legal Hold for a specific object	Read	object* (p. 1576)	s3:DataAccessPointAccount (p. 1577) s3:DataAccessPointArn (p. 1577) s3:AccessPointNetworkOrigin (p. 1577) s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
GetObjectRetention	GET Object Legal Hold for a specific object	Read	object* (p. 1576)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				s3:DataAccessPointAccount (p. 1577) s3:DataAccessPointArn (p. 1577) s3:AccessPointNetworkOrigin (p. 1577) s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
GetObjectTagging	This implementation of the GET operation returns the tags associated with an object. You send the GET request against the tagging subresource associated with the object.	Read	object* (p. 1576)	s3:DataAccessPointAccount (p. 1577) s3:DataAccessPointArn (p. 1577) s3:AccessPointNetworkOrigin (p. 1577) s3:ExistingObjectTag/<key> (p. 1577) s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
GetObjectTorrent	return torrent files from a bucket.	Read	object* (p. 1576)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
GetObjectVersion	To return a different version, use the <code>versionId</code> subresource.	Read	object* (p. 1576)	s3:DataAccessPointAccount (p. 1577) s3:DataAccessPointArn (p. 1577) s3:AccessPointNetworkOrigin (p. 1577) s3:ExistingObjectTag/<key> (p. 1577) s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:versionid (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
GetObjectVersion	To return ACL information about a different version, use the <code>versionId</code> subresource.	Read	object* (p. 1576)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				s3:DataAccessPointAccount (p. 1577) s3:DataAccessPointArn (p. 1577) s3:AccessPointNetworkOrigin (p. 1577) s3:ExistingObjectTag/ <key> (p. 1577) s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:versionid (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
GetObjectVersionForReplication	Permission exercised by S3 For Replication	Read	object* (p. 1576)		
				s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
GetObjectVersionSpecific	GET Object tagging (for a Specific Version of the Object)	Read	object* (p. 1576)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				s3:DataAccessPointAccount (p. 1577) s3:DataAccessPointArn (p. 1577) s3:AccessPointNetworkOrigin (p. 1577) s3:ExistingObjectTag/ <key> (p. 1577) s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:versionid (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
GetObjectVersionTorrent	To return Torrent files about a different version, use the <code>versionId</code> subresource.	Read	object* (p. 1576)		
				s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:versionid (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
GetReplicationConfiguration	Returns the replication configuration information set on the bucket.	Read	bucket* (p. 1576)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
ListAccessPoints	Lists access points.	Read		s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
ListAllMyBuckets	Returns a list of all buckets owned by the authenticated sender of the request.	List		s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
ListBucket	Returns some or all (up to 1000) of the objects in a bucket.	List	bucket* (p. 1576)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				s3:DataAccessPointAccount (p. 1577) s3:DataAccessPointArn (p. 1577) s3:AccessPointNetworkOrigin (p. 1577) s3:authtype (p. 1577) s3:delimiter (p. 1577) s3:max-keys (p. 1577) s3:prefix (p. 1578) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
ListBucketMultipartUploads	Lists in-progress multipart uploads	Read	bucket* (p. 1576)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				s3:DataAccessPointAccount (p. 1577) s3:DataAccessPointArn (p. 1577) s3:AccessPointNetworkOrigin (p. 1577) s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
ListBucketVersion	Use the versions subresource to list metadata about all of the versions of objects in a bucket.	Read	bucket* (p. 1576)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				s3:DataAccessPointAccount (p. 1577) s3:DataAccessPointArn (p. 1577) s3:AccessPointNetworkOrigin (p. 1577) s3:authtype (p. 1577) s3:delimiter (p. 1577) s3:max-keys (p. 1577) s3:prefix (p. 1578) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
ListJobs	Lists current jobs and jobs that have ended recently.	Read		s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
ListMultipartUploadParts	Lists the parts that have been uploaded for a specific multipart upload.	Read	object* (p. 1576)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				s3:DataAccessPointAccount (p. 1577) s3:DataAccessPointArn (p. 1577) s3:AccessPointNetworkOrigin (p. 1577) s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
<code>ObjectOwnerOverrideBucketOwner</code>	Permission exercised by S3 BucketOwner	Permissions management	object* management (p. 1576) 	s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
PutAccelerateConfiguration	This implementation of the PUT <code>operation</code> uses the accelerate subresource to set the Transfer Acceleration state of an existing bucket.	Write	bucket* (p. 1576)	s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
PutAccessPointPolicy	Add to or replace a data policy on a access point.	Permissions management	accesspoint* (p. 1576)	s3:DataAccessPointAccount (p. 1577) s3:DataAccessPointArn (p. 1577) s3:AccessPointNetworkOrigin (p. 1577) s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
PutAccountPublicAccessBlock	Create or modify the PublicAccessBlock configuration for an AWS account.	Permissions management		s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
PutAnalyticsConfiguration	This implementation of the PUT operation adds an analytics configuration (identified by the analytics ID) to the bucket. You can have up to 1,000 analytics configurations per bucket.	Write	bucket* (p. 1576)		
				s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
PutBucketAcl	Set the permissions on an existing bucket using access control lists (ACL).	Permissions management	bucket* (p. 1576)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-acl (p. 1578) s3:x-amz-content-sha256 (p. 1578) s3:x-amz-grant-full-control (p. 1578) s3:x-amz-grant-read (p. 1578) s3:x-amz-grant-read-acp (p. 1578) s3:x-amz-grant-write (p. 1578) s3:x-amz-grant-write-acp (p. 1578)	
PutBucketCORS	Sets the CORS configuration for your bucket.	Write	bucket* (p. 1576)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
PutBucketLogging	Set the logging parameters for a bucket.	Write	bucket* (p. 1576)		
				s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
PutBucketNotification	Enables you to receive notifications when certain events happen in your bucket.	Write	bucket* (p. 1576)		
				s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
PutBucketObjectLockConfiguration	PUT Object Lock configuration on a specific bucket	Write	bucket* (p. 1576)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578)	
PutBucketPolicy	Add to or replace a policy on a bucket.	Permissions management	bucket* (p. 1576)		
				s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
	Permissions management	bucket* (p. 1576)			
		Create or modify the PublicAccessBlock configuration for an specific S3 bucket.			s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)
PutBucketRequestConfiguration	Set the request payment configuration of a bucket.	Write	bucket* (p. 1576)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
PutBucketTagging	Add a set of tags to an existing bucket.	Tagging	bucket* (p. 1576)		
				s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
PutBucketVersioning	Set the versioning state of an existing bucket.	Write	bucket* (p. 1576)		
				s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
PutBucketWebsite	Sets the configuration of the website that is specified in the website subresource.	Write	bucket* (p. 1576)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
	Sets the encryption configuration for the bucket.	Write	bucket* (p. 1576)	s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
	This implementation of the PUT operation adds an inventory configuration (identified by the inventory ID) to the bucket. You can have up to 1,000 inventory configurations per bucket.	Write	bucket* (p. 1576)	s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
	Creates a new lifecycle configuration for the bucket or replaces an existing lifecycle configuration.	Write	bucket* (p. 1576)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
PutMetricsConfig	Sets or updates a metrics configuration for the CloudWatch request metrics (specified by the metrics configuration ID) from the bucket.	Write	bucket* (p. 1576)		
				s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
PutObject	Adds an object to a bucket.	Write	object* (p. 1576)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				s3:DataAccessPointAccount (p. 1577) s3:DataAccessPointArn (p. 1577) s3:AccessPointNetworkOrigin (p. 1577) s3:RequestObjectTag/<key> (p. 1577) s3:RequestObjectTagKeys (p. 1577) s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-acl (p. 1578) s3:x-amz-content-sha256 (p. 1578) s3:x-amz-copy-source (p. 1578) s3:x-amz-grant-full-control (p. 1578) s3:x-amz-grant-read (p. 1578) s3:x-amz-grant-read-acp (p. 1578) s3:x-amz-grant-	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				write (p. 1578) s3:x-amz-grant-write-acp (p. 1578) s3:x-amz-metadata-directive (p. 1578) s3:x-amz-server-side-encryption (p. 1578) s3:x-amz-server-side-encryption-aws-kms-key-id (p. 1578) s3:x-amz-storage-class (p. 1579) s3:x-amz-website-redirect-location (p. 1579) s3:object-lock-mode (p. 1578) s3:object-lock-retain-until-date (p. 1578) s3:object-lock-remaining-retention-days (p. 1578)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				s3:object-lock-legal-hold (p. 1578)	
PutObjectAcl	Set the access control list (ACL) permissions for an object that already exists in a bucket.	Permissions management	object* (p. 1576)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				s3:DataAccessPointAccount (p. 1577) s3:DataAccessPointArn (p. 1577) s3:AccessPointNetworkOrigin (p. 1577) s3:ExistingObjectTag/<key> (p. 1577) s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-acl (p. 1578) s3:x-amz-content-sha256 (p. 1578) s3:x-amz-grant-full-control (p. 1578) s3:x-amz-grant-read (p. 1578) s3:x-amz-grant-read-acp (p. 1578) s3:x-amz-grant-write (p. 1578) s3:x-amz-grant-write-acp (p. 1578)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				s3:x-amz-storage-class (p. 1579)	
PutObjectLegalHold	PUT Object Legal Hold on a specific object	Write	object* (p. 1576)	s3:DataAccessPointAccount (p. 1577) s3:DataAccessPointArn (p. 1577) s3:AccessPointNetworkOrigin (p. 1577) s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578) s3:object-lock-legal-hold (p. 1578)	
PutObjectRetention	PUT Object Retention on a specific object	Write	object* (p. 1576)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				s3:DataAccessPointAccount (p. 1577) s3:DataAccessPointArn (p. 1577) s3:AccessPointNetworkOrigin (p. 1577) s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578) s3:object-lock-mode (p. 1578) s3:object-lock-retain-until-date (p. 1578) s3:object-lock-remaining-retention-days (p. 1578)	
PutObjectTagging	This implementation of the <code>PUT</code> operation uses the <code>tagging</code> subresource to add a set of tags to an existing object.	Tagging	object* (p. 1576)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				s3:DataAccessPointAccount (p. 1577) s3:DataAccessPointArn (p. 1577) s3:AccessPointNetworkOrigin (p. 1577) s3:ExistingObjectTag/<key> (p. 1577) s3:RequestObjectTag/<key> (p. 1577) s3:RequestObjectTagKeys (p. 1577) s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
PutObjectVersion	The ACL of an object is set at the object version level.	Permissions management	object* management (p. 1576)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				s3:DataAccessPointAccount (p. 1577) s3:DataAccessPointArn (p. 1577) s3:AccessPointNetworkOrigin (p. 1577) s3:ExistingObjectTag/<key> (p. 1577) s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:versionid (p. 1578) s3:x-amz-acl (p. 1578) s3:x-amz-content-sha256 (p. 1578) s3:x-amz-grant-full-control (p. 1578) s3:x-amz-grant-read (p. 1578) s3:x-amz-grant-read-acp (p. 1578) s3:x-amz-grant-write (p. 1578) s3:x-amz-grant-	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				write-acp (p. 1578) s3:x-amz-storage-class (p. 1579)	
PutObjectVersionTagging	PUT Object tagging (for a specific Version of the Object)	Tagging	object* (p. 1576)	s3:DataAccessPointAccount (p. 1577) s3:DataAccessPointArn (p. 1577) s3:AccessPointNetworkOrigin (p. 1577) s3:ExistingObjectTag/<key> (p. 1577) s3:RequestObjectTag/<key> (p. 1577) s3:RequestObjectTagKeys (p. 1577) s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:versionid (p. 1578) s3:x-amz-content-sha256 (p. 1578)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
PutReplicationConfiguration	In a versioning-enabled bucket, this operation creates a new replication configuration (or replaces an existing one, if present).	Write	bucket* (p. 1576)	s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
ReplicateDelete	Permission exercised by S3 replication	Write	object* (p. 1576)	s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
ReplicateObject	Permission exercised by S3 replication	Write	object* (p. 1576)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578) s3:x-amz-server-side-encryption (p. 1578) s3:x-amz-server-side-encryption-aws-kms-key-id (p. 1578)	
ReplicateTags	Permission exercised by S3 replication	Tagging	object* (p. 1576)		
				s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
RestoreObject	Restores a temporary copy of an archived object.	Write	object* (p. 1576)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				s3:DataAccessPointAccount (p. 1577) s3:DataAccessPointArn (p. 1577) s3:AccessPointNetworkOrigin (p. 1577) s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578)	
UpdateJobPriority	Updates an existing job's priority.	Write	job* (p. 1576) s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578) s3:RequestJobPriority (p. 1577) s3:ExistingJobPriority (p. 1577) s3:ExistingJobOperation (p. 1577)		
UpdateJobStatus	Updates the status for the specified job.	Write	job* (p. 1576)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				s3:authtype (p. 1577) s3:signatureage (p. 1578) s3:signatureversion (p. 1578) s3:x-amz-content-sha256 (p. 1578) s3:ExistingJobPriority (p. 1577) s3:ExistingJobOperation (p. 1577) s3:JobSuspendedCause (p. 1577)	

Resource Types Defined by Amazon S3

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 1525\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
accesspoint	arn:\${Partition}:s3:\${Region}: \${Account}:accesspoint/\${AccessPointName}	
bucket	arn:\${Partition}:s3::::\${BucketName}	
object	arn:\${Partition}:s3::::\${BucketName}/ \${ObjectName}	
job	arn:\${Partition}:s3:\${Region}: \${Account}:job/\${JobId}	

Condition Keys for Amazon S3

Amazon S3 defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
s3:AccessPointNetworkAccessPoint	The network type from which traffic may be received by the access point involved in the request	String
s3:DataAccessPointAccount	The AWS Account ID of the account that owns the data operations access point involved in the request	String
s3:DataAccessPointArn	The ARN of the data operations access point involved in the request	String
s3:ExistingJobOperation		String
s3:ExistingJobPriority		Numeric
s3:ExistingObjectTag<key>	Enables you to verify that an existing object tag has the specific tag key and value.	String
s3:JobSuspendedCause		String
s3:LocationConstraint	Enables you to restrict users to creating buckets in only a specific region.	String
s3:RequestJobOperation		String
s3:RequestJobPriority		Numeric
s3:RequestObjectTag<key>	Restrict the tag keys and values that you want to allow on objects.	String
s3:RequestObjectTagKeys	restrict the tag keys that you want to allow on objects.	String
s3:VersionId	Enables you to limit the permission for the PutObjectVersionTagging action to a specific object version.	String
s3:authtype		String
s3:delimiter	Enables you to require the user to specify the delimiter parameter in the GET Bucket Object versions request.	String
s3:locationconstraint	Enables you to restrict the user to creating a bucket in only a specific region.	String
s3:max-keys	Enables you to limit the number of keys Amazon S3 returns in response to ListBucket requests by requiring the user to specify the max-keys parameter.	Numeric

Condition Keys	Description	Type
s3:object-lock-legal-hold	Enables enforcement of the specified object legal hold status	String
s3:object-lock-mode	Enables enforcement of the specified object retention mode	String
s3:object-lock-remaining-retention-days	Enables enforcement of an object relative to the remaining retention days	String
s3:object-lock-retain-until-date	Enables enforcement of a specific retain-until-date	String
s3:prefix	Enables you to limit the response of the ListBucket API to key names with specific prefix.	String
s3:signatureage		Numeric
s3:signatureversion		String
s3:versionid		String
s3:x-amz-acl	Enables you to require specific access permissions when uploading an object.	String
s3:x-amz-content-sha256		String
s3:x-amz-copy-source	Enables you to restrict the copy source to a specific bucket, a specific folder in the bucket, or a specific object in a bucket.	String
s3:x-amz-grant-full-control		String
s3:x-amz-grant-read		String
s3:x-amz-grant-read-acp		String
s3:x-amz-grant-write		String
s3:x-amz-grant-write-acp		String
s3:x-amz-metadata-directive	Enables you to enforce certain behavior (COPY vs. REPLACE) when objects are uploaded.	String
s3:x-amz-server-side-encryption	Enables you to require the user to specify this header in the request to ensure that objects the user uploads are encrypted when they are saved.	String
s3:x-amz-server-side-encryption-aws-kms-key-id		String

Condition Keys	Description	Type
s3:x-amz-storage-class		String
s3:x-amz-website-redirect-location		String

Actions, Resources, and Condition Keys for Amazon SageMaker

Amazon SageMaker (service prefix: `sagemaker`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon SageMaker \(p. 1579\)](#)
- [Resource Types Defined by Amazon SageMaker \(p. 1604\)](#)
- [Condition Keys for Amazon SageMaker \(p. 1607\)](#)

Actions Defined by Amazon SageMaker

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AddTags	Adds or overwrites one or more tags for the specified Amazon SageMaker resource.	Tagging	app (p. 1605)		
			automl-job (p. 1606)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			domain (p. 1605)		
			endpoint (p. 1606)		
			endpoint-config (p. 1606)		
			experiment (p. 1606)		
			experiment-trial (p. 1607)		
			experiment-trial-component (p. 1607)		
			flow-definition (p. 1604)		
			human-task-ui (p. 1604)		
			hyper-parameter-tuning-job (p. 1606)		
			labeling-job (p. 1604)		
			model (p. 1606)		
			monitoring-schedule (p. 1606)		
			notebook-instance (p. 1605)		
			processing-job (p. 1605)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			training-job (p. 1605)		
			transform-job (p. 1606)		
			user-profile (p. 1605)		
			workteam (p. 1605)		
			aws:RequestTag/ \${TagKey} (p. 1607)		
	Associate a trial component with a trial constraint	Write	experiment-trial* (p. 1607)		
			experiment-trial-component* (p. 1607)		
BatchGetMetrics [permission only]	Retrieve metrics associated with SageMaker Resources such as Training Jobs. This API is not publicly exposed at this point, however admins can control this action	Read	training-job* (p. 1605)		
BatchPutMetrics [permission only]	Publish metrics associated with a SageMaker Resource such as a Training Job. This API is not publicly exposed at this point, however admins can control this action	Write	training-job* (p. 1605)		
CreateAlgorithm	Create an algorithm.	Write	algorithm* (p. 1605)		
CreateApp	Grants permission to create an App for a SageMaker Studio UserProfile	Write	app* (p. 1605)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:RequestTag/\${TagKey} (p. 1607) aws:TagKeys (p. 1607) sagemaker:InstanceTypes (p. 1608)	
CreateAutoMLJob	Creates automl job.	Write	automl-job* (p. 1606)		iam:PassRole
				aws:RequestTag/\${TagKey} (p. 1607) aws:TagKeys (p. 1607) sagemaker:InterContainerTrafficEncryption (p. 1608) sagemaker:OutputKmsKey (p. 1608) sagemaker:VolumeKmsKey (p. 1608) sagemaker:VpcSecurityGroupIds (p. 1608) sagemaker:VpcSubnets (p. 1608)	
CreateCodeRepository	Create a code repository.	Write	code-repository* (p. 1605)		
CreateCompilationJob	Create a compilation job.	Write	compilation-job* (p. 1606)		iam:PassRole
CreateDomain	Grants permission to create a Domain for SageMaker Studio	Write	domain* (p. 1605)		iam:CreateServiceLinkedRole iam:PassRole

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:RequestTag/\${TagKey} (p. 1607) aws:TagKeys (p. 1607) sagemaker:AppNetworkAccess (p. 1607) sagemaker:InstanceTypes (p. 1608) sagemaker:VpcSecurityGroupIds (p. 1608) sagemaker:VpcSubnets (p. 1608) sagemaker:DomainSharingOutputKms (p. 1607) sagemaker:HomeEfsFileSystemKmsKey (p. 1608)	
CreateEndpoint	Creates an endpoint using the endpoint configuration specified in the request.	Write	endpoint* (p. 1606)		
				aws:RequestTag/\${TagKey} (p. 1607) aws:TagKeys (p. 1607)	
CreateEndpointConfig	Creates an endpoint configuration that can be deployed using Amazon SageMaker hosting services.	Write	endpoint-config* (p. 1606)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:RequestTag/\${TagKey} (p. 1607) aws:TagKeys (p. 1607) sagemaker:AcceleratorTypes (p. 1607) sagemaker:InstanceTypes (p. 1608) sagemaker:ModelArn (p. 1608) sagemaker:VolumeKmsKey (p. 1608)	
CreateExperiment	Create an experiment.	Write	experiment* (p. 1606)		
				aws:RequestTag/\${TagKey} (p. 1607) aws:TagKeys (p. 1607)	
CreateFlowDefinition	Creates a flow definition, which defines settings for a human workflow.	Write	flow-definition* (p. 1604)		iam:PassRole
				sagemaker:WorkteamArn (p. 1608) sagemaker:WorkteamType (p. 1608) aws:RequestTag/\${TagKey} (p. 1607) aws:TagKeys (p. 1607)	
CreateHumanTaskUi	Defines the settings you will use for the human review workflow user interface.	Write	human-task-ui* (p. 1604)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:RequestTag/\${TagKey} (p. 1607) aws:TagKeys (p. 1607)	
CreateHyperParameterJob (p. 1607)	Creates hyper parameter tuning job that can be deployed using Amazon SageMaker.	Write	hyper-parameter-tuning-job* (p. 1606)	aws:RequestTag/\${TagKey} (p. 1607) aws:TagKeys (p. 1607) sagemaker:FileSystemAccessMode (p. 1607) sagemaker:FileSystemDirectoryPath (p. 1607) sagemaker:FileSystemId (p. 1607) sagemaker:FileSystemType (p. 1608) sagemaker:InstanceTypes (p. 1608) sagemaker:InterContainerTrafficEncryption (p. 1608) sagemaker:MaxRuntimeInSeconds (p. 1608) sagemaker:NetworkIsolation (p. 1608) sagemaker:OutputKmsKey (p. 1608) sagemaker:VolumeKmsKey (p. 1608) sagemaker:VpcSecurityGroupIds (p. 1608) sagemaker:VpcSubnets (p. 1608)	iam:PassRole

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateLabelingJob	Starts a labeling job. A labeling job takes unlabeled data in and produces labeled data as output, which can be used for training SageMaker models.	Write	labeling-job* (p. 1604)		iam:PassRole
				sagemaker:WorkteamArn (p. 1608)	
				sagemaker:WorkteamType (p. 1608)	
				sagemaker:VolumeKmsKey (p. 1608)	
				sagemaker:OutputKmsKey (p. 1608)	
				aws:RequestTag/\${TagKey} (p. 1607)	
				aws:TagKeys (p. 1607)	
CreateModel	Creates a model in Amazon SageMaker. In the request, you specify a name for the model and describe one or more containers.	Write	model* (p. 1606)		iam:PassRole
				aws:RequestTag/\${TagKey} (p. 1607)	
				aws:TagKeys (p. 1607)	
				sagemaker:NetworkIsolation (p. 1608)	
				sagemaker:VpcSecurityGroupIds (p. 1608)	
				sagemaker:VpcSubnets (p. 1608)	
CreateModelPackage	Create a model package.	Write	model-package* (p. 1606)		
CreateMonitoringSchedule	Creates a monitoring schedule.	Write	monitoring-schedule* (p. 1606)		iam:PassRole

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:RequestTag/\${TagKey} (p. 1607) aws:TagKeys (p. 1607) sagemaker:InstanceTypes (p. 1608) sagemaker:MaxRuntimeInSeconds (p. 1608) sagemaker:NetworkIsolation (p. 1608) sagemaker:OutputKmsKey (p. 1608) sagemaker:VolumeKmsKey (p. 1608) sagemaker:VpcSecurityGroupIds (p. 1608) sagemaker:VpcSubnets (p. 1608)	
CreateNotebookInstance	Creates an Amazon SageMaker notebook instance. A notebook instance is an Amazon EC2 instance running on a Jupyter Notebook.	Write	notebook-instance* (p. 1605)		iam:PassRole

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:RequestTag/\${TagKey} (p. 1607) aws:TagKeys (p. 1607) sagemaker:AcceleratorTypes (p. 1607) sagemaker:DirectInternetAccess (p. 1607) sagemaker:InstanceTypes (p. 1608) sagemaker:RootAccess (p. 1608) sagemaker:VolumeKmsKey (p. 1608) sagemaker:VpcSecurityGroupIds (p. 1608) sagemaker:VpcSubnets (p. 1608)	
CreateNotebookInstanceLifecycleConfig	Creates an notebook instance lifecycle configuration that can be deployed using Amazon SageMaker.	Write	notebook-instance-lifecycle-config* (p. 1605)		
CreatePresignedDomainUrl	Grants permission to return a URL that you can use from your browser to connect to the Domain as a specified UserProfile when AuthMode is 'IAM'	Write	user-profile* (p. 1605)		
CreatePresignedNotebookInstanceUrl	Returns a URL that you can use from your browser to connect to the Notebook Instance.	Write	notebook-instance* (p. 1605)		
CreateProcessingJob	Starts a processing job. After processing completes, Amazon SageMaker saves the resulting artifacts and other optional output to an Amazon S3 location that you specify.	Write	processing-job* (p. 1605)		iam:PassRole

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:RequestTag/\${TagKey} (p. 1607) aws:TagKeys (p. 1607) sagemaker:InstanceTypes (p. 1608) sagemaker:MaxRuntimeInSeconds (p. 1608) sagemaker:NetworkIsolation (p. 1608) sagemaker:OutputKmsKey (p. 1608) sagemaker:VolumeKmsKey (p. 1608) sagemaker:VpcSecurityGroupIds (p. 1608) sagemaker:VpcSubnets (p. 1608)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateTrainingJob	<p>Starts a model training job. After training completes, Amazon SageMaker saves the resulting model artifacts and other optional output to an Amazon S3 location that you specify.</p>	Write	training-job* (p. 1605)		iam:PassRole
CreateTransformJob	<p>Starts a transform job. After the results are obtained, Amazon SageMaker saves them to an Amazon S3 location that you specify.</p>	Write	transform-job* (p. 1606)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:RequestTag/\${TagKey} (p. 1607) aws:TagKeys (p. 1607) sagemaker:InstanceTypes (p. 1608) sagemaker:ModelArn (p. 1608) sagemaker:OutputKmsKey (p. 1608) sagemaker:VolumeKmsKey (p. 1608)	
CreateTrial	Create a trial.	Write	experiment-trial* (p. 1607)		
				aws:RequestTag/\${TagKey} (p. 1607) aws:TagKeys (p. 1607)	
CreateTrialComponent	Create a trial component.	Write	experiment-trial-component* (p. 1607)		
				aws:RequestTag/\${TagKey} (p. 1607) aws:TagKeys (p. 1607)	
CreateUserProfile	Grants permission to create a <code>UserProfile</code> for a SageMaker Studio Domain	Write	user-profile* (p. 1605)		iam:PassRole

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:RequestTag/\${TagKey} (p. 1607) aws:TagKeys (p. 1607) sagemaker:VpcSecurityGroupIds (p. 1608) sagemaker:InstanceTypes (p. 1608) sagemaker:DomainSharingOutputKms (p. 1607)	
CreateWorkteam	Create a workteam.	Write	workteam* (p. 1605)		
				aws:RequestTag/\${TagKey} (p. 1607) aws:TagKeys (p. 1607)	
DeleteAlgorithm	Deletes an algorithm.	Write	algorithm* (p. 1605)		
DeleteApp	Grants permission to delete an App	Write	app* (p. 1605)		
DeleteCodeRepository	Deletes a code repository.	Write	code-repository* (p. 1605)		
DeleteDomain	Grants permission to delete a Domain	Write	domain* (p. 1605)		
DeleteEndpoint	Deletes an endpoint. Amazon SageMaker frees up all the resources that were deployed when the endpoint was created.	Write	endpoint* (p. 1606)		
DeleteEndpointConfig	Deletes the endpoint configuration created using the CreateEndpointConfig API. The DeleteEndpointConfig API deletes only the specified configuration. It does not delete any endpoints created using the configuration.	Write	endpoint-config* (p. 1606)		
DeleteExperiment	Deletes an experiment.	Write	experiment* (p. 1606)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteFlowDefinition	Deletes the specified flow definition.	Write	flow-definition* (p. 1604)		
DeleteHumanLoop	Deletes the specified human loop.	Write	human-loop* (p. 1604)		
DeleteModel	Deletes a model created using the CreateModel API. The DeleteModel API deletes only the model entry in Amazon SageMaker that you created by calling the CreateModel API. It does not delete model artifacts, inference code, or the IAM role that you specified when creating the model.	Write	model* (p. 1606)		
DeleteModelPackage	Deletes a model package.	Write	model-package* (p. 1606)		
DeleteMonitoringSchedule	Deletes a monitoring schedule. Amazon SageMaker will no longer run the scheduled monitoring.	Write	monitoring-schedule* (p. 1606)		
DeleteNotebookInstance	Deletes an Amazon SageMaker notebook instance. Before you can delete a notebook instance, you must call the StopNotebookInstance API.	Write	notebook-instance* (p. 1605)		
DeleteNotebookLifecycleConfiguration	Deletes an notebook instance lifecycle configuration that can be deployed using Amazon SageMaker.	Write	notebook-instance-lifecycle-config* (p. 1605)		
DeleteTags	Deletes the specified set of tags from an Amazon SageMaker resource.	Tagging	app (p. 1605)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			endpoint (p. 1606)		
			endpoint-config (p. 1606)		
			experiment (p. 1606)		
			experiment-trial (p. 1607)		
			experiment-trial-component (p. 1607)		
			flow-definition (p. 1604)		
			human-task-ui (p. 1604)		
			hyper-parameter-tuning-job (p. 1606)		
			labeling-job (p. 1604)		
			model (p. 1606)		
			monitoring-schedule (p. 1606)		
			notebook-instance (p. 1605)		
			processing-job (p. 1605)		
			training-job (p. 1605)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			transform-job (p. 1606)		
			user-profile (p. 1605)		
			workteam (p. 1605)		
				aws:TagKeys (p. 1607)	
DeleteTrial	Deletes a trial.	Write	experiment-trial* (p. 1607)		
DeleteTrialComponent	Deletes a trial component.	Write	experiment-trial-component* (p. 1607)		
DeleteUserProfile	Grants permission to delete a UserProfile	Write	user-profile* (p. 1605)		
DeleteWorkteam	Deletes a workteam.	Write	workteam* (p. 1605)		
DescribeAlgorithm	Returns information about an algorithm.	Read	algorithm* (p. 1605)		
DescribeApp	Grants permission to describe an App	Read	app* (p. 1605)		
DescribeAutoMLJob	Describes an automl job that was created via CreateAutoMLJob API.	Read	automl-job* (p. 1606)		
DescribeCodeRepository	Returns information about a repository.	Read	code-repository* (p. 1605)		
DescribeCompilationJob	Returns information about a compilation job.	Read	compilation-job* (p. 1606)		
DescribeDomain	Grants permission to describe a Domain	Read	domain* (p. 1605)		
DescribeEndpoint	Returns the description of an endpoint.	Read	endpoint* (p. 1606)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeEndpointConfig	Returns the description of an endpoint configuration, which was created using the CreateEndpointConfig API.	Read	endpoint-config* (p. 1606)		
DescribeExperiment	Returns information about an experiment.	Read	experiment* (p. 1606)		
DescribeFlowDefinition	Returns detailed information about the specified flow definition.	Read	flow-definition* (p. 1604)		
DescribeHumanLoop	Returns detailed information about the specified human loop.	Read	human-loop* (p. 1604)		
DescribeHumanTaskUi	Returns detailed information about the specified human review workflow user interface.	Read	human-task-ui* (p. 1604)		
DescribeHyperParameterTuningJob	Describes a hyper parameter tuning job that was created via CreateHyperParameterTuningJob API.	Read	hyper-parameter-tuning-job* (p. 1606)		
DescribeLabelingJob	Returns information about a labeling job.	Read	labeling-job* (p. 1604)		
DescribeModel	Describes a model that you created using the CreateModel API.	Read	model* (p. 1606)		
DescribeModelPackage	Returns information about a model package.	Read	model-package* (p. 1606)		
DescribeMonitoringSchedule	Returns information about a monitoring schedule.	Read	monitoring-schedule* (p. 1606)		
DescribeNotebookInstance	Returns information about a notebook instance.	Read	notebook-instance* (p. 1605)		
DescribeNotebookInstanceLifecycleConfig	Describes an notebook instance lifecycle configuration that was created via CreateNotebookInstanceLifecycleConfig API.	Read	notebook-instance-lifecycle-config* (p. 1605)		
DescribeProcessingJob	Returns information about a processing job.	Read	processing-job* (p. 1605)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeSubscribedWorkteams	Returns information about a workteam.	Read	workteam* (p. 1605)		
DescribeTrainingJobs	Returns information about a training job.	Read	training-job* (p. 1605)		
DescribeTransformJobs	Returns information about a transform job.	Read	transform-job* (p. 1606)		
DescribeTrial	Returns information about a trial.	Read	experiment-trial* (p. 1607)		
DescribeTrialComponents	Returns information about a trial component.	Read	experiment-trial-component* (p. 1607)		
DescribeUserProfile	Grants permission to describe a user profile.	Read	user-profile* (p. 1605)		
DescribeWorkteam	Returns information about a workteam.	Read	workteam* (p. 1605)		
DisassociateTrialComponents	Disassociate a trial component with a trial.	Write	experiment-trial* (p. 1607)		
			experiment-trial-component* (p. 1607)		
			processing-job* (p. 1605)		
GetSearchSuggestions	Get search suggestions when provided with keyword.	Read	training-job* (p. 1605)		
InvokeEndpoint	After you deploy a model into production using Amazon SageMaker hosting services, your client applications use this API to get inferences from the model hosted at the specified endpoint.	Read	endpoint* (p. 1606)		
ListAlgorithms	Lists algorithms.	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListApps	Grants permission to list the Apps in your account	List			
ListAutoMLJobs	Lists automl jobs created via the CreateAutoMLJob.	List			
ListCandidatesForAutoMLJob	Lists candidates for AutoMLJob created via the CreateAutoMLJob.	List			
ListCodeRepositories	Lists code repositories.	List			
ListCompilationJobs	Lists compilation jobs.	List			
ListDomains	Grants permission to list the Domains in your account	List			
ListEndpointConfigs	Lists endpoint configurations.	List			
ListEndpoints	Lists endpoints.	List			
ListExperiments	Lists experiments.	List			
ListFlowDefinitions	Returns summary information about flow definitions, given the specified parameters.	List			
ListHumanLoops	Returns summary information about human loops, given the specified parameters.	List			
ListHumanTaskUi	Returns summary information about human review workflow user interfaces, given the specified parameters.	List			
ListHyperParameterJobs	Lists hyper parameter tuning jobs that was created using Amazon SageMaker.	List			
ListLabelingJobs	Lists labeling jobs.	List			
ListLabelingJobsForWorkteam	Lists labeling jobs for workteam.	List	workteam*	(p. 1605)	
ListModelPackages	Lists model packages.	List			
ListModels	Lists the models created with the CreateModel API.	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListMonitoringExecutions	Lists monitoring executions.	List			
ListMonitoringSchedules	Lists monitoring schedules.	List			
ListNotebookInstances	Lists notebook instance lifecycle configurations that can be deployed using Amazon SageMaker.	List			
ListNotebookInstances	Returns a list of the Amazon SageMaker notebook instances in the requester's account in an AWS Region.	List			
ListProcessingJobs	Lists processing jobs.	List			
ListSubscribedWorkteams	Lists subscribed workteams.	List			
ListTags	Returns the tag set associated with the specified resource.	List	app (p. 1605)		
			automl-job (p. 1606)		
			domain (p. 1605)		
			endpoint (p. 1606)		
			endpoint-config (p. 1606)		
			experiment (p. 1606)		
			experiment-trial (p. 1607)		
			experiment-trial-component (p. 1607)		
			flow-definition (p. 1604)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			human-task-ui (p. 1604)		
			hyper-parameter-tuning-job (p. 1606)		
			labeling-job (p. 1604)		
			model (p. 1606)		
			monitoring-schedule (p. 1606)		
			notebook-instance (p. 1605)		
			training-job (p. 1605)		
			transform-job (p. 1606)		
			user-profile (p. 1605)		
			workteam (p. 1605)		
ListTrainingJobs	Lists training jobs.	List			
ListTrainingJobsForHyperParameterTuningJob	Lists training jobs for a hyper parameter tuning job that was created using Amazon SageMaker.	List	hyper-parameter-tuning-job* (p. 1606)		
ListTransformJobs	Lists transform jobs.	List			
ListTrialComponents	Lists trial components.	List			
ListTrials	Lists trials.	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListUserProfiles	Grants permission to list the UserProfiles in your account	List			
ListWorkteams	Lists workteams.	List			
RenderUiTemplate	Render a UI template used for a human annotation task.	Read			iam:PassRole
Search	Search for a training job.	Read	training-job* (p. 1605)		
StartHumanLoop	Starts a human loop.	Write	flow-definition* (p. 1604)		
StartMonitoringSchedule	Starts a monitoring schedule.	Write	monitoring-schedule* (p. 1606)		
StartNotebookInstance	Launches an EC2 instance with the latest version of the libraries and attaches your EBS volume.	Write	notebook-instance* (p. 1605)		
StopAutoMLJob	Stops a running automl job created via the CreateAutoMLJob.	Write	automl-job* (p. 1606)		
StopCompilationJob	Stops a compilation job.	Write	compilation-job* (p. 1606)		
StopHumanLoop	Stops the specified human loop.	Write	human-loop* (p. 1604)		
StopHyperParameterTuningJob	Stops a running hyper parameter tuning job create via the CreateHyperParameterTuningJob.	Write	hyper-parameter-tuning-job* (p. 1606)		
StopLabelingJob	Stops a labeling job. Any labels already generated will be exported before stopping.	Write	labeling-job* (p. 1604)		
StopMonitoringSchedule	Stops a monitoring schedule.	Write	monitoring-schedule* (p. 1606)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
StopNotebookInstances	Terminates the EC2 instance. Before terminating the instance, Amazon SageMaker disconnects the EBS volume from it. Amazon SageMaker preserves the EBS volume.	Write	notebook-instance* (p. 1605)		
StopProcessingJob	Stops a processing job. To stop a job, Amazon SageMaker sends the algorithm the SIGTERM signal, which delays job termination for 120 seconds.	Write	processing-job* (p. 1605)		
StopTrainingJob	Stops a training job. To stop a job, Amazon SageMaker sends the algorithm the SIGTERM signal, which delays job termination for 120 seconds.	Write	training-job* (p. 1605)		
StopTransformJob	Stops a transform job. When Amazon SageMaker receives a StopTransformJob request, the status of the job changes to Stopping. After Amazon SageMaker stops the job, the status is set to Stopped	Write	transform-job* (p. 1606)		
UpdateCodeRepository	Updates a code repository.	Write	code-repository* (p. 1605)		
UpdateDomain	Grants permission to update a Domain	Write	domain* (p. 1605)		
				sagemaker:VpcSecurityGroupIds (p. 1608)	
				sagemaker:InstanceTypes (p. 1608)	
UpdateEndpoint	Updates an endpoint to use the endpoint configuration specified in the request.	Write	endpoint* (p. 1606)		
UpdateEndpointWeightsAndCapacities	Updates variant weight, capacity, or both of one or more variants associated with an endpoint.	Write	endpoint* (p. 1606)		
UpdateExperiment	Updates an experiment.	Write	experiment* (p. 1606)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UpdateMonitoringSchedule	Updates a monitoring schedule.	Write	monitoring-schedule* (p. 1606)		iam:PassRole
				aws:RequestTag/\${TagKey} (p. 1607) aws:TagKeys (p. 1607) sagemaker:InstanceTypes (p. 1608) sagemaker:MaxRuntimeInSeconds (p. 1608) sagemaker:NetworkIsolation (p. 1608) sagemaker:OutputKmsKey (p. 1608) sagemaker:VolumeKmsKey (p. 1608) sagemaker:VpcSecurityGroupIds (p. 1608) sagemaker:VpcSubnets (p. 1608)	
UpdateNotebook	Updates a notebook instance.	Write	notebook-instance* (p. 1605)		
	Notebook instance updates include upgrading or downgrading the EC2 instance used for your notebook instance to accommodate changes in your workload requirements. You can also update the VPC security groups.			sagemaker:AcceleratorTypes (p. 1607) sagemaker:InstanceTypes (p. 1608) sagemaker:RootAccess (p. 1608)	
UpdateNotebookInstanceLifecycleConfig	Updates a notebook instance lifecycle configuration created with the CreateNotebookInstanceLifecycleConfig API.	Write	notebook-instance-lifecycle-config* (p. 1605)		
UpdateTrial	Updates a trial.	Write	experiment-trial* (p. 1607)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UpdateTrialComponent	Updates a trial component.	Write	experiment-trial-component* (p. 1607)		
UpdateUserProfile	Grants permission to update a <code>UserProfile</code>	Write	user-profile* (p. 1605)	sagemaker:InstanceTypes (p. 1608) sagemaker:VpcSecurityGroupIds (p. 1608) sagemaker:InstanceTypes (p. 1608) sagemaker:DomainSharingOutputKms (p. 1607)	
UpdateWorkteam	Updates a workteam.	Write	workteam* (p. 1605)		

Resource Types Defined by Amazon SageMaker

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1579\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
human-loop	arn:\${Partition}:sagemaker:\${Region}: \${Account}:human-loop/\${HumanLoopName}	
flow-definition	arn:\${Partition}:sagemaker: \${Region}: \${Account}:flow-definition/ \${FlowDefinitionName}	aws:ResourceTag/ \${TagKey} (p. 1607) sagemaker:ResourceTag/ \${TagKey} (p. 1608)
human-task-ui	arn:\${Partition}:sagemaker:\${Region}: \${Account}:human-task-ui/\${HumanTaskUiName}	aws:ResourceTag/ \${TagKey} (p. 1607) sagemaker:ResourceTag/ \${TagKey} (p. 1608)
labeling-job	arn:\${Partition}:sagemaker:\${Region}: \${Account}:labeling-job/\${LabelingJobName}	aws:ResourceTag/ \${TagKey} (p. 1607)

Resource Types	ARN	Condition Keys
		sagemaker:ResourceTag/\${TagKey} (p. 1608)
workteam	<code>arn:\${Partition}:sagemaker:\${Region}:\${Account}:workteam/\${WorkteamName}</code>	aws:ResourceTag/\${TagKey} (p. 1607) sagemaker:ResourceTag/\${TagKey} (p. 1608)
domain	<code>arn:\${Partition}:sagemaker:\${Region}:\${Account}:domain/\${DomainId}</code>	aws:ResourceTag/\${TagKey} (p. 1607) sagemaker:ResourceTag/\${TagKey} (p. 1608)
user-profile	<code>arn:\${Partition}:sagemaker:\${Region}:\${Account}:user-profile/\${DomainId}/\${UserProfileName}</code>	aws:ResourceTag/\${TagKey} (p. 1607) sagemaker:ResourceTag/\${TagKey} (p. 1608)
app	<code>arn:\${Partition}:sagemaker:\${Region}:\${Account}:app/\${DomainId}/\${UserProfileName}/\${AppType}/\${AppName}</code>	aws:ResourceTag/\${TagKey} (p. 1607) sagemaker:ResourceTag/\${TagKey} (p. 1608)
notebook-instance	<code>arn:\${Partition}:sagemaker:\${Region}:\${Account}:notebook-instance/\${NotebookInstanceName}</code>	aws:ResourceTag/\${TagKey} (p. 1607) sagemaker:ResourceTag/\${TagKey} (p. 1608)
notebook-instance-lifecycle-config	<code>arn:\${Partition}:sagemaker:\${Region}:\${Account}:notebook-instance-lifecycle-config/\${NotebookInstanceLifecycleConfigName}</code>	
code-repository	<code>arn:\${Partition}:sagemaker:\${Region}:\${Account}:code-repository/\${CodeRepositoryName}</code>	
algorithm	<code>arn:\${Partition}:sagemaker:\${Region}:\${Account}:algorithm/\${AlgorithmName}</code>	
training-job	<code>arn:\${Partition}:sagemaker:\${Region}:\${Account}:training-job/\${TrainingJobName}</code>	aws:ResourceTag/\${TagKey} (p. 1607) sagemaker:ResourceTag/\${TagKey} (p. 1608)
processing-job	<code>arn:\${Partition}:sagemaker:\${Region}:\${Account}:processing-job/\${ProcessingJobName}</code>	aws:ResourceTag/\${TagKey} (p. 1607) sagemaker:ResourceTag/\${TagKey} (p. 1608)

Resource Types	ARN	Condition Keys
hyper-parameter-tuning-job	arn:\${Partition}:sagemaker:\${Region}: \${Account}:hyper-parameter-tuning-job/ \${HyperParameterTuningJobName}	aws:ResourceTag/\${TagKey} (p. 1607) sagemaker:ResourceTag/\${TagKey} (p. 1608)
model-package	arn:\${Partition}:sagemaker:\${Region}: \${Account}:model-package/\${ModelPackageName}	
model	arn:\${Partition}:sagemaker:\${Region}: \${Account}:model/\${ModelName}	aws:ResourceTag/\${TagKey} (p. 1607) sagemaker:ResourceTag/\${TagKey} (p. 1608)
endpoint-config	arn:\${Partition}:sagemaker: \${Region}: \${Account}:endpoint-config/ \${EndpointConfigName}	aws:ResourceTag/\${TagKey} (p. 1607) sagemaker:ResourceTag/\${TagKey} (p. 1608)
endpoint	arn:\${Partition}:sagemaker:\${Region}: \${Account}:endpoint/\${EndpointName}	aws:ResourceTag/\${TagKey} (p. 1607) sagemaker:ResourceTag/\${TagKey} (p. 1608)
transform-job	arn:\${Partition}:sagemaker:\${Region}: \${Account}:transform-job/\${TransformJobName}	aws:ResourceTag/\${TagKey} (p. 1607) sagemaker:ResourceTag/\${TagKey} (p. 1608)
compilation-job	arn:\${Partition}:sagemaker: \${Region}: \${Account}:compilation-job/ \${CompilationJobName}	
automl-job	arn:\${Partition}:sagemaker:\${Region}: \${Account}:automl-job/\${AutoMLJobJobName}	aws:ResourceTag/\${TagKey} (p. 1607) sagemaker:ResourceTag/\${TagKey} (p. 1608)
monitoring-schedule	arn:\${Partition}:sagemaker:\${Region}: \${Account}:monitoring-schedule/ \${MonitoringScheduleName}	aws:ResourceTag/\${TagKey} (p. 1607) sagemaker:ResourceTag/\${TagKey} (p. 1608)
experiment	arn:\${Partition}:sagemaker:\${Region}: \${Account}:experiment/\${ExperimentName}	aws:ResourceTag/\${TagKey} (p. 1607) sagemaker:ResourceTag/\${TagKey} (p. 1608)

Resource Types	ARN	Condition Keys
experiment-trial	arn:\${Partition}:sagemaker:\${Region}:\${Account}:experiment-trial/\${TrialName}	aws:ResourceTag/\${TagKey} (p. 1607) sagemaker:ResourceTag/\${TagKey} (p. 1608)
experiment-trial-component	arn:\${Partition}:sagemaker:\${Region}:\${Account}:experiment-trial-component/\${TrialComponentName}	aws:ResourceTag/\${TagKey} (p. 1607) sagemaker:ResourceTag/\${TagKey} (p. 1608)

Condition Keys for Amazon SageMaker

Amazon SageMaker defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	A key that is present in the request the user makes to the SageMaker service.	String
aws:ResourceTag/\${TagKey}	A tag key and value pair.	String
aws:TagKeys	The list of all the tag key names associated with the resource in the request.	String
sagemaker:AcceleratorType	The list of all accelerator types associated with the resource in the request.	ArrayOfString
sagemaker:AppNetworkInterfaces	App network access associated with the resource in the request.	String
sagemaker:DirectInternetAccess	The direct internet access associated with the resource in the request.	String
sagemaker:DomainSharingOutputKmsKey	The Domain sharing output KMS key associated with the resource in the request.	ARN
sagemaker:FileSystemRequestMode	File system access mode associated with the resource in the request.	String
sagemaker:FileSystemRequestPath	File system directory path associated with the resource in the request.	String
sagemaker:FileSystemId	A file system ID associated with the resource in the request.	String

Condition Keys	Description	Type
sagemaker:FileSystemType	File system type associated with the resource in the request.	String
sagemaker:HomeEfsHomeDirectories	The KMS Key Id of the EFS File System used for UserProfile Home Directories , which is associated with the resource in the request.	ARN
sagemaker:InstanceTypes	The list of all instance types associated with the resource in the request.	ArrayOfString
sagemaker:InterContainerTrafficEncryption	The inter container traffic encryption associated with the resource in the request.	Bool
sagemaker:MaxRuntime	The max runtime in seconds associated with the resource in the request.	Numeric
sagemaker:ModelArn	The model arn associated with the resource in the request.	ARN
sagemaker:NetworkIsolation	The network isolation associated with the resource in the request.	Bool
sagemaker:OutputKmsKey	The output kms key associated with the resource in the request.	ARN
sagemaker:ResourceTag	The preface string for a tag key and value pair attached to a resource.	String
sagemaker:ResourceTag/\${TagKey}	A tag key and value pair.	String
sagemaker:RootAccess	The root access associated with the resource in the request.	String
sagemaker:VolumeKmsKey	The volume kms key associated with the resource in the request.	ARN
sagemaker:VpcSecuritySource	The list of all vpc security group ids associated with the resource in the request.	ArrayOfString
sagemaker:VpcSubnets	The list of all vpc subnets associated with the resource in the request.	ArrayOfString
sagemaker:WorkteamArn	The workteam arn associated to the request.	ARN
sagemaker:WorkteamPublic	The workteam type associated to the request. This can be public-crowd, private-crowd or vendor-crowd.	String

Actions, Resources, and Condition Keys for AWS Savings Plans

AWS Savings Plans (service prefix: `savingsplans`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Savings Plans \(p. 1609\)](#)
- [Resource Types Defined by AWS Savings Plans \(p. 1610\)](#)
- [Condition Keys for AWS Savings Plans \(p. 1610\)](#)

Actions Defined by AWS Savings Plans

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateSavingsPlan	Grants permission to create a savings plan	Write		aws:RequestTag/\${TagKey} (p. 1610) aws:TagKeys (p. 1611)	
DescribeSavingsPlan	Grants permission to describe the Rates associated with customers savings plan	Read	savingsplan* (p. 1610)		
				aws:ResourceTag/\${TagKey} (p. 1611)	
DescribeSavingsPlans	Grants permission to describe the savings plans associated with customers account	Read	savingsplan* (p. 1610)		
				aws:ResourceTag/\${TagKey} (p. 1611)	
DescribeOfferingRate	Grants permission to describe the Offering Rate associated with savings plans offerings	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeSavingsPlansOfferings	Grants permission to describe the AWS Savings Plans offerings that customer is eligible to purchase	Read			
ListTagsForResource	Grants permission to list tags for a savings plan	List	savingsplan* (p. 1610)		
TagResource	Grants permission to tag a savings plan	Tagging	savingsplan* (p. 1610)		
				aws:TagKeys (p. 1611) aws:RequestTag/\${TagKey} (p. 1610)	
UntagResource	Grants permission to untag a savings plan	Tagging	savingsplan* (p. 1610)		
				aws:TagKeys (p. 1611)	

Resource Types Defined by AWS Savings Plans

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1609\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
savingsplan	<code>arn:\${Partition}:savingsplans::\${Account}:savingsplan/\${ResourceId}</code>	aws:ResourceTag/\${TagKey} (p. 1611)

Condition Keys for AWS Savings Plans

AWS Savings Plans defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Filters actions based on the allowed set of values for each of the tags	String

Condition Keys	Description	Type
aws:ResourceTag/\${TagKey}	Filters actions based on tag-value associated with the resource	String
aws:TagKeys	Filters actions based on the presence of mandatory tags in the request	String

Actions, Resources, and Condition Keys for AWS Secrets Manager

AWS Secrets Manager (service prefix: `secretsmanager`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Secrets Manager \(p. 1611\)](#)
- [Resource Types Defined by AWS Secrets Manager \(p. 1617\)](#)
- [Condition Keys for AWS Secrets Manager \(p. 1617\)](#)

Actions Defined by AWS Secrets Manager

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CancelRotateSecret	Enables the user to cancel an in-progress secret rotation.	Write	Secret* (p. 1617)		secretsmanager:SecretId (p. 1618)

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				secretsmanager:resource/AllowRotationLambdaArn (p. 1618)	secretsmanager:ResourceTag/tag-key (p. 1618)
CreateSecret	Enables the user to create a secret that stores encrypted data that can be queried and rotated.	Tagging		secretsmanager:Name (p. 1618)	secretsmanager:Description (p. 1618)
DeleteResourcePolicy	Enables the user to delete the resource policy attached to a secret.	Permissions management	Secret* Secret (p. 1617)	secretsmanager:SecretId (p. 1618)	secretsmanager:resource/AllowRotationLambdaArn (p. 1618)
DeleteSecret	Enables the user to delete a secret.	Write	Secret* Secret (p. 1617)		secretsmanager:ResourceTag/tag-key (p. 1618)

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				secretsmanager:SecretId (p. 1618) secretsmanager:resource/AllowRotationLambdaArn (p. 1618) secretsmanager:RecoveryWindowInDays (p. 1618) secretsmanager:ForceDeleteWithoutRecovery (p. 1618) secretsmanager:ResourceTag/tag-key (p. 1618)	
DescribeSecret	Enables the user to retrieve the metadata about a secret, but not the encrypted data.	Read	Secret* (p. 1617)		
				secretsmanager:SecretId (p. 1618) secretsmanager:resource/AllowRotationLambdaArn (p. 1618) secretsmanager:ResourceTag/tag-key (p. 1618)	
GetRandomPassword	Enables the user to generate a random string for use in password creation.	Read			
GetResourcePolicy	Enables the user to get the resource policy attached to a secret.	Read	Secret* (p. 1617)		
				secretsmanager:SecretId (p. 1618) secretsmanager:resource/AllowRotationLambdaArn (p. 1618) secretsmanager:ResourceTag/tag-key (p. 1618)	
GetSecretValue	Enables the user to retrieve and decrypt the encrypted data.	Read	Secret* (p. 1617)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				secretsmanager:SecretId (p. 1618) secretsmanager:VersionId (p. 1618) secretsmanager:VersionStage (p. 1618) secretsmanager:resource/AllowRotationLambdaArn (p. 1618) secretsmanager:ResourceTag/tag-key (p. 1618)	
ListSecretVersionIds	Enables the user to list the available versions of a secret.	Read	Secret* (p. 1617)		
				secretsmanager:SecretId (p. 1618) secretsmanager:resource/AllowRotationLambdaArn (p. 1618) secretsmanager:ResourceTag/tag-key (p. 1618)	
ListSecrets	Enables the user to list the available secrets.	List			
PutResourcePolicy	Enables the user to attach a resource policy to a secret.	Permissions management	Secret* (p. 1617)		
				secretsmanager:SecretId (p. 1618) secretsmanager:resource/AllowRotationLambdaArn (p. 1618) secretsmanager:ResourceTag/tag-key (p. 1618)	
PutSecretValue	Enables the user to create a new version of the secret with new encrypted data.	Write	Secret* (p. 1617)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				secretsmanager:SecretId (p. 1618) secretsmanager:resource/AllowRotationLambdaArn (p. 1618) secretsmanager:ResourceTag/tag-key (p. 1618)	
RestoreSecret	Enables the user to cancel deletion of a secret.	Write	Secret* (p. 1617)		
				secretsmanager:SecretId (p. 1618) secretsmanager:resource/AllowRotationLambdaArn (p. 1618) secretsmanager:ResourceTag/tag-key (p. 1618)	
RotateSecret	Enables the user to start rotation of a secret.	Write	Secret* (p. 1617)		
				secretsmanager:SecretId (p. 1618) secretsmanager:RotationLambdaARN (p. 1618) secretsmanager:resource/AllowRotationLambdaArn (p. 1618) secretsmanager:ResourceTag/tag-key (p. 1618)	
TagResource	Enables the user to add tags to a secret.	Tagging	Secret* (p. 1617)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				secretsmanager:SecretId (p. 1618) aws:RequestTag/tag-key (p. 1618) aws:TagKeys (p. 1618) secretsmanager:resource/AllowRotationLambdaArn (p. 1618) secretsmanager:ResourceTag/tag-key (p. 1618)	
UntagResource	Enables the user to remove tags from a secret.	Tagging	Secret* (p. 1617)		
				secretsmanager:SecretId (p. 1618) aws:TagKeys (p. 1618) secretsmanager:resource/AllowRotationLambdaArn (p. 1618) secretsmanager:ResourceTag/tag-key (p. 1618)	
UpdateSecret	Enables the user to update a secret with new metadata or with a new version of the encrypted data.	Write	Secret* (p. 1617)		
				secretsmanager:SecretId (p. 1618) secretsmanager:Description (p. 1618) secretsmanager:KmsKeyId (p. 1618) secretsmanager:resource/AllowRotationLambdaArn (p. 1618) secretsmanager:ResourceTag/tag-key (p. 1618)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UpdateSecretVersionStage	Enables the user to move a stage from one secret to another.	Write	Secret* (p. 1617)	secretsmanager:SecretId (p. 1618) secretsmanager:VersionStage (p. 1618) secretsmanager:resource/AllowRotationLambdaArn (p. 1618) secretsmanager:ResourceTag/tag-key (p. 1618)	

Resource Types Defined by AWS Secrets Manager

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1611\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
Secret	<code>arn:\${Partition}:secretsmanager:\${Region}:\${Account}:secret:\${SecretId}</code>	aws:RequestTag/tag-key (p. 1618) aws:TagKeys (p. 1618) secretsmanager:ResourceTag/tag-key (p. 1618) secretsmanager:resource/AllowRotationLambdaArn (p. 1618)

Condition Keys for AWS Secrets Manager

AWS Secrets Manager defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/tag-key	Filters access by a key that is present in the request the user makes to the Secrets Manager service.	String
aws:TagKeys	Filters access by the list of all the tag key names present in the request the user makes to the Secrets Manager service.	String
secretsmanager:Description	Filters access by the description text in the request.	String
secretsmanager:ForceDeleteWithoutRecovery	Filters access by whether the secret is to be deleted immediately without any recovery window.	Boolean
secretsmanager:KmsKeyId	Filters access by the ARN of the KMS key in the request.	String
secretsmanager:Name	Filters access by the friendly name of the secret in the request.	String
secretsmanager:RecoveryWindowBeforeDelete	Filters access by the number of days that Secrets Manager waits before it can delete the secret.	Long
secretsmanager:ResourceTag/tag-key	Filters access by a tag key and value pair.	String
secretsmanager:RotationLambdaArn	Filters access by the ARN of the rotation Lambda function in the request.	ARN
secretsmanager:SecretId	Filters access by the SecretID value in the request.	ARN
secretsmanager:VersionId	Filters access by the unique identifier of the version of the secret in the request.	String
secretsmanager:VersionStage	Filters access by the list of version stages in the request.	String
secretsmanager:resource/AllowRotationLambdaArn	Filters access by the ARN of the rotation Lambda function associated with the secret.	ARN

Actions, Resources, and Condition Keys for AWS Security Hub

AWS Security Hub (service prefix: `securityhub`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Security Hub \(p. 1619\)](#)
- [Resource Types Defined by AWS Security Hub \(p. 1622\)](#)

- [Condition Keys for AWS Security Hub \(p. 1622\)](#)

Actions Defined by AWS Security Hub

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AcceptInvitation	Grants permission to accept Security Hub invitations to become a member account.	Write			
BatchDisableStandards	Grants permission to disable standards in Security Hub.	Write	standards-subscription* (p. 1622)		
BatchEnableStandards	Grants permission to enable standards in Security Hub.	Write	standard* (p. 1622)		
BatchImportFindings	Grants permission to import findings into Security Hub from an integrated product.	Write			securityhub:TargetAccount (p. 1623)
CreateActionTargets	Grants permission to create custom actions in Security Hub.	Write			
CreateInsight	Grants permission to create insights in Security Hub. Insights are collections of related findings.	Write			
CreateMembers	Grants permission to create member accounts in Security Hub.	Write			
DeclineInvitations	Grants permission to decline Security Hub invitations to become a member account.	Write			
DeleteActionTargets	Grants permission to delete custom actions in Security Hub.	Write	action-target* (p. 1622)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteInsight	Grants permission to delete insights from Security Hub.	Write	insight* (p. 1622)		
DeleteInvitations	Grants permission to delete Security Hub invitations to become a member account.	Write			
DeleteMembers	Grants permission to delete Security Hub member accounts.	Write			
DescribeActionTargets	Grants permission to retrieve a list of custom actions using the API.	Read			
DescribeHub	Grants permission to retrieve information about the hub resource in your account.	Read			
DescribeProducts	Grants permission to retrieve information about the available Security Hub product integrations.	Read			
DescribeStandardInformation	Grants permission to retrieve information about Security Hub standards controls.	Read	hub* (p. 1622)		
DisableImportFindingsImporting	Grants permission to disable the findings import for a Security Hub integrated product.	Write	product* (p. 1622)		
DisableSecurityHub	Grants permission to disable Security Hub.	Write			
DisassociateFromHubMembers	Grants permission to a Security Hub member account to disassociate from the associated master account.	Write			
DisassociateMembers	Grants permission to disassociate Security Hub member accounts from the associated master account.	Write			
EnableImportFindingsImporting	Grants permission to enable the findings import for a Security Hub integrated product.	Write	product* (p. 1622)		
EnableSecurityHub	Grants permission to enable Security Hub.	Write		aws:RequestTag/ \${TagKey} (p. 1623)	aws:TagKeys (p. 1623)

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetEnabledStandards	Grants permission to retrieve a list of the standards that are enabled in Security Hub.	List			
GetFindings	Grants permission to retrieve a list of findings from Security Hub.	Read			
GetInsightResults	Grants permission to retrieve insight results from Security Hub.	Read	insight* (p. 1622)		
GetInsights	Grants permission to retrieve Security Hub insights.	List	insight* (p. 1622)		
GetInvitationsCount	Grants permission to retrieve the count of Security Hub membership invitations sent to the account.	Read			
GetMasterAccount	Grants permission to retrieve details about the Security Hub master account.	Read			
GetMembers	Grants permission to retrieve the details of Security Hub member accounts.	Read			
InviteMembers	Grants permission to invite other AWS accounts to become Security Hub member accounts.	Write			
ListEnabledProducts	Grants permission to retrieve the Security Hub integrated products that are currently enabled.	List			
ListInvitations	Grants permission to retrieve the Security Hub invitations sent to the account.	List			
ListMembers	Grants permission to retrieve details about Security Hub member accounts associated with the master account.	List			
ListTagsForResource	Grants permission to list of tags associated with a resource.	List	hub* (p. 1622)		
TagResource	Grants permission to add tags to a Security Hub resource.	Write	hub* (p. 1622)		
UntagResource	Grants permission to remove tags from a Security Hub resource.	Write	hub* (p. 1622)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UpdateActionTargets	Grants permission to update custom actions in Security Hub.	Write	action-target* (p. 1622)		
UpdateFindings	Grants permission to update Security Hub findings.	Write			
UpdateInsight	Grants permission to update insights in Security Hub.	Write	insight* (p. 1622)		
UpdateStandards	Grants permission to update Security Hub standards controls.	Write	hub* (p. 1622)		

Resource Types Defined by AWS Security Hub

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 1619\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
insight	arn:\${Partition}:securityhub:\${Region}: \${Account}:insight/\${CompanyId}/ \${ProductId}/\${UniqueId}	
standard	arn:\${Partition}:securityhub:::ruleset/ \${StandardsName}/v/\${StandardsVersion}	
standards-subscription	arn:\${Partition}:securityhub:\${Region}: \${Account}:subscription/\${StandardsName}/v/ \${StandardsVersion}	
product-subscription	arn:\${Partition}:securityhub:\${Region}: \${Account}:product-subscription/\${Company}/ \${ProductId}	
product	arn:\${Partition}:securityhub:\${Region}: \${Account}:product/\${Company}/\${ProductId}	
hub	arn:\${Partition}:securityhub:\${Region}: \${Account}:hub/default	aws:ResourceTag/\${TagKey} (p. 1623)
action-target	arn:\${Partition}:securityhub:\${Region}: \${Account}:action/custom/\${Id}	

Condition Keys for AWS Security Hub

AWS Security Hub defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys in the IAM Policy Reference](#).

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Filters actions based on the presence of tag key-value pairs in the request	String
aws:ResourceTag/\${TagKey}	Filters actions based on tag key-value pairs attached to the resource	String
aws:TagKeys	Filters actions based on the presence of tag keys in the request	String
securityhub:TargetAccountId	The ID of the AWS account into which you want to import findings. In the AWS Security Finding format, this field is called AwsAccountId	String

Actions, Resources, and Condition Keys for AWS Security Token Service

AWS Security Token Service (service prefix: sts) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Security Token Service \(p. 1623\)](#)
- [Resource Types Defined by AWS Security Token Service \(p. 1628\)](#)
- [Condition Keys for AWS Security Token Service \(p. 1629\)](#)

Actions Defined by AWS Security Token Service

You can specify the following actions in the Action element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the Resource element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AssumeRole	Returns a set of temporary security credentials that you can use to access AWS resources that you might not normally have access to	Write	role* (p. 1629)	aws:TagKeys (p. 1629) aws:PrincipalTag/ {\$TagKey} (p. 1629) aws:RequestTag/ {\$TagKey} (p. 1629) sts:TransitiveTagKeys (p. 1631)	
AssumeRoleWithSAML	Returns a set of temporary security credentials for users who have been authenticated via a SAML authentication response	Write	role* (p. 1629)	saml:namequalifier (p. 1631) saml:sub (p. 1631) saml:sub_type (p. 1631) saml:aud (p. 1630) saml:iss (p. 1631) saml:doc (p. 1630) saml:cn (p. 1630) saml:commonName (p. 1630) saml:eduorghomepageuri (p. 1630) saml:eduorgidentityauthnpolicyuri (p. 1630) saml:eduorglegalname (p. 1630) saml:eduorgsuperioruri (p. 1630)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
					<p>saml:eduorgwhitepagesuri (p. 1630)</p> <p>saml:edupersonaffiliation (p. 1630)</p> <p>saml:edupersonassurance (p. 1630)</p> <p>saml:edupersonentitlement (p. 1630)</p> <p>saml:edupersonnickname (p. 1630)</p> <p>saml:edupersonorgdn (p. 1630)</p> <p>saml:edupersonorgunitdn (p. 1630)</p> <p>saml:edupersonprimaryaffiliation (p. 1630)</p> <p>saml:edupersonprimaryorgunitdn (p. 1630)</p> <p>saml:edupersonprincipalname (p. 1630)</p> <p>saml:edupersonscopedaffiliation (p. 1631)</p> <p>saml:edupersontargetedid (p. 1631)</p> <p>saml:givenName (p. 1631)</p> <p>saml:mail (p. 1631)</p> <p>saml:name (p. 1631)</p> <p>saml:organizationStatus (p. 1631)</p> <p>saml:primaryGroupSID (p. 1631)</p> <p>saml:surname (p. 1631)</p> <p>saml:uid (p. 1631)</p>

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				saml:x500UniqueIdentifier (p. 1631)	aws:TagKeys (p. 1629)
	AssumeRoleWithWebIdentity <small>Security</small>	Write	role* (p. 1629)	aws:PrincipalTag/ <code> \${TagKey}</code> (p. 1629)	aws:RequestTag/ <code> \${TagKey}</code> (p. 1629)
				sts:TransitiveTagKeys (p. 1631)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				cognito-identity.amazonaws.com:amr (p. 1629) cognito-identity.amazonaws.com:aud (p. 1629) cognito-identity.amazonaws.com:sub (p. 1629) www.amazon.com:app_id (p. 1631) www.amazon.com:user_id (p. 1631) graph.facebook.com:app_id (p. 1630) graph.facebook.com:id (p. 1630) accounts.google.com:aud (p. 1629) accounts.google.com:oauth2 (p. 1629) accounts.google.com:sub (p. 1629) aws:TagKeys (p. 1629) aws:PrincipalTag/\${TagKey} (p. 1629) aws:RequestTag/\${TagKey} (p. 1629) sts:TransitiveTagKeys (p. 1631)	
DecodeAuthorizationMessage	Decodes additional information about the authorization status of a request from an encoded message returned in response to an AWS request	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetAccessKeyInfo	Returns details about the access key id passed as a parameter to the request.	Read			
GetCallerIdentity	Returns details about the IAM identity whose credentials are used to call the API	Read			
GetFederationToken	Returns a set of temporary security credentials (consisting of an access key ID, a secret access key, and a security token) for a federated user	Read	user (p. 1629)	aws:TagKeys (p. 1629) aws:PrincipalTag/\${TagKey} (p. 1629) aws:RequestTag/\${TagKey} (p. 1629)	
GetSessionToken	Returns a set of temporary security credentials (consisting of an access key ID, a secret access key, and a security token) for an AWS account or IAM user	Read			
TagSession [permission only]	Grants permission to add tags to a STS session	Tagging	role (p. 1629)		
			user (p. 1629)		
				aws:TagKeys (p. 1629) aws:PrincipalTag/\${TagKey} (p. 1629) aws:RequestTag/\${TagKey} (p. 1629) sts:TransitiveTagKeys (p. 1631)	

Resource Types Defined by AWS Security Token Service

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1623\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you

can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
role	arn:\${Partition}:iam::\${Account}:role/\${RoleNameWithPath}	aws:ResourceTag/\${TagKey} (p. 1629)
user	arn:\${Partition}:iam::\${Account}:user/\${UserNameWithPath}	

Condition Keys for AWS Security Token Service

AWS Security Token Service defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
accounts.google.com:aud	Filters actions based on the Google application ID	String
accounts.google.com:oaud	Filters actions based on the Google audience	String
accounts.google.com:user	Filters actions based on the subject of the claim (the Google user ID)	String
aws:FederatedProvider	Filters actions based on the IdP that was used to authenticate the user	String
aws:PrincipalTag/\${TagKey}	Filters actions based on the tag associated with the principal that is making the request	String
aws:RequestTag/\${TagKey}	Filters actions based on the tags that are passed in the request	String
aws:ResourceTag/\${TagKey}	Filters actions based on the tags associated with the resource	String
aws:TagKeys	Filters actions based on the tag keys that are passed in the request	String
cognito-identity.amazonaws.com:Cognito	Filters actions based on the login information for Amazon Cognito	String
cognito-identity.amazonaws.com:aud	Filters actions based on the Amazon Cognito identity pool ID	String
cognito-identity.amazonaws.com:Cognito	Filters actions based on the subject of the claim (the Amazon Cognito user ID)	String

Condition Keys	Description	Type
graph.facebook.com:app_id	Filters actions based on the Facebook application ID	String
graph.facebook.com:id	Filters actions based on the Facebook user ID	String
saml:aud	Filters actions based on the endpoint URL to which SAML assertions are presented	String
saml:cn	Filters actions based on the eduOrg attribute	String
saml:commonName	Filters actions based on the commonName attribute	String
saml:doc	Filters actions based on the principal that was used to assume the role	String
saml:eduorghomepageuri	Filters actions based on the eduOrg attribute	String
saml:eduorgidentityauthnpolicyuri	Filters actions based on the eduOrg attribute	String
saml:eduorglegalname	Filters actions based on the eduOrg attribute	String
saml:eduorgsuperioruri	Filters actions based on the eduOrg attribute	String
saml:eduorgwhitepagesuri	Filters actions based on the eduOrg attribute	String
saml:edupersonaffiliation	Filters actions based on the eduPerson attribute	String
saml:edupersonassurance	Filters actions based on the eduPerson attribute	String
saml:edupersonentitlement	Filters actions based on the eduPerson attribute	String
saml:edupersonnickname	Filters actions based on the eduPerson attribute	String
saml:edupersonorgdn	Filters actions based on the eduPerson attribute	String
saml:edupersonorgunitdn	Filters actions based on the eduPerson attribute	String
saml:edupersonprimaryaffiliation	Filters actions based on the eduPerson attribute	String
saml:edupersonprimaryorgunitdn	Filters actions based on the eduPerson attribute	String
saml:edupersonprincipalname	Filters actions based on the eduPerson attribute	String

Condition Keys	Description	Type
saml:edupersonscopedaffiliation	Filters actions based on the eduPerson attribute	String
saml:edupersontargetedid	Filters actions based on the eduPerson attribute	String
saml:givenName	Filters actions based on the givenName attribute	String
saml:iss	Filters actions based on the issuer, which is represented by a URN	String
saml:mail	Filters actions based on the mail attribute	String
saml:name	Filters actions based on the name attribute	String
saml:namequalifier	Filters actions based on the hash value of the issuer, account ID, and friendly name	String
saml:organizationStatus	Filters actions based on the organizationStatus attribute	String
saml:primaryGroupSID	Filters actions based on the primaryGroupSID attribute	String
saml:sub	Filters actions based on the subject of the claim (the SAML user ID)	String
saml:sub_type	Filters actions based on the value persistent, transient, or the full Format URI	String
saml:surname	Filters actions based on the surname attribute	String
saml:uid	Filters actions based on the uid attribute	String
saml:x500UniqueIdentifier	Filters actions based on the uid attribute	String
sts:ExternalId	Filters actions based on the unique identifier required when you assume a role in another account	String
sts:TransitiveTagKeys	Filters actions based on the transitive tag keys that are passed in the request	String
www.amazon.com:appId	Filters actions based on the Login with Amazon application ID	String
www.amazon.com:user_id	Filters actions based on the Login with Amazon user ID	String

Actions, Resources, and Condition Keys for AWS Server Migration Service

AWS Server Migration Service (service prefix: `sms`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Server Migration Service \(p. 1632\)](#)
- [Resource Types Defined by AWS Server Migration Service \(p. 1634\)](#)
- [Condition Keys for AWS Server Migration Service \(p. 1634\)](#)

Actions Defined by AWS Server Migration Service

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateApp	Create an application configuration to migrate on-premise application onto AWS..	Write			
CreateReplicationJob	Create a job to migrate on-premise server onto AWS.	Write			
DeleteApp	Delete an existing application configuration.	Write			
DeleteAppLaunchConfiguration	Delete launch configuration for an existing application.	Write			
DeleteAppReplicationConfiguration	Delete replication configuration for an existing application..	Write			
DeleteReplicationJob	Delete an existing job to migrate on-premise server onto AWS.	Write			
DeleteServerCatalog	Delete the complete list of on-premise servers gathered into AWS.	Write			
DisassociateConnector	Disassociate a connector that has been associated.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GenerateChangeSet	Generate a changeSet for the CloudFormation stack of an application.	Write			
GenerateTemplate	Generate a CloudFormation template for an existing application.	Write			
GetApp	Get the configuration and statuses for an existing application.	Read			
GetAppLaunchConfig	Get launch configuration for an existing application.	Read			
GetAppReplication	Get replication configuration for an existing application.	Read			
GetConnectors	Get all connectors that have been associated.	Read			
GetMessages [permission only]	Gets messages from AWS Server Migration Service to Server Migration Connector.	Read			
GetReplicationJobs	Get all existing jobs to migrate on-premise servers onto AWS.	Read			
GetReplicationRuns	Get all runs for an existing job.	Read			
GetServers	Get all servers that have been imported.	Read			
ImportServerCatalog	Gathers a complete list of on-premise servers.	Write			
LaunchApp	Create and launch a CloudFormation stack for an existing application.	Write			
ListApps	Get a list of summaries for existing applications.	List			
PutAppLaunchConfiguration	Create or update launch configuration for an existing application.	Write			
PutAppReplicationConfiguration	Create or update replication configuration for an existing application.	Write			
SendMessage [permission only]	Send message from Server Migration Connector to AWS Server Migration Service.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
StartAppReplication	Create and start replication jobs for an existing application.	Write			
StartOnDemandReplication	Start a replication run for an existing replication job.	Write			
StopAppReplication	Stop and delete replication jobs for an existing application.	Write			
TerminateApp	Terminate the CloudFormation stack for an existing application.	Write			
UpdateApp	Update an existing application configuration	Write			
UpdateReplicationJob	Update an existing job to migrate on-premise server onto AWS.	Write			

Resource Types Defined by AWS Server Migration Service

AWS Server Migration Service does not support specifying a resource ARN in the `Resource` element of an IAM policy statement. To allow access to AWS Server Migration Service, specify “`Resource`”: “`*`” in your policy.

Condition Keys for AWS Server Migration Service

`ServerMigrationService` has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for AWS Serverless Application Repository

AWS Serverless Application Repository (service prefix: `serverlessrepo`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

Topics

- [Actions Defined by AWS Serverless Application Repository \(p. 1634\)](#)
- [Resource Types Defined by AWS Serverless Application Repository \(p. 1636\)](#)
- [Condition Keys for AWS Serverless Application Repository \(p. 1636\)](#)

Actions Defined by AWS Serverless Application Repository

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources (“`*`”) in the `Resource` element of your

policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateApplication	Creates an application, optionally including an AWS SAM file to create the first application version in the same call.	Write			
CreateApplicationVersion	Creates an application version.	Write	applications* (p. 1636)		
CreateCloudFormationStack	Creates an AWS CloudFormation stack for the given application.	Write	applications* (p. 1636)		
CreateCloudFormationStackTemplate	Creates an AWS CloudFormation stack template	Write	applications* (p. 1636)		
DeleteApplication	Delete the specified Application	Write	applications* (p. 1636)		
GetApplication	Gets the specified application.	Read	applications* (p. 1636)		
GetApplicationPolicy	Gets the policy for the specified application.	Read	applications* (p. 1636)		
GetCloudFormationStackTemplate	Gets the specified AWS CloudFormation template	Read	applications* (p. 1636)		
ListApplications	Retrieves the list of applications containing application	List	applications* (p. 1636)		
ListApplicationVersions	Lists versions for the specified application owned by the requester.	List	applications* (p. 1636)		
ListApplications	Lists applications owned by the requester.	List			
PutApplicationPolicy	Puts the policy for the specified application.	Write	applications* (p. 1636)		
SearchApplications	Gets all applications authorized for this user	Read			
UpdateApplication	Updates meta-data of the application	Write	applications* (p. 1636)		

Resource Types Defined by AWS Serverless Application Repository

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 1634\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
applications	<code>arn:\${Partition}:serverlessrepo:\${Region}:\${Account}:applications/\${ResourceId}</code>	

Condition Keys for AWS Serverless Application Repository

Serverless Application Repository has no service-specific context keys that can be used in the condition element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for AWS Service Catalog

AWS Service Catalog (service prefix: `servicecatalog`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Service Catalog \(p. 1636\)](#)
- [Resource Types Defined by AWS Service Catalog \(p. 1643\)](#)
- [Condition Keys for AWS Service Catalog \(p. 1643\)](#)

Actions Defined by AWS Service Catalog

You can specify the following actions in the Action element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the Resource element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AcceptPortfolioShares	Accepts a portfolio that has been shared with you	Write	Portfolio* (p. 1643)		
AssociateBudgetWithResource	Associates a budget with a resource	Write			
AssociatePrincipalWithPortfolio	Associates an IAM principal with a portfolio, giving the specified principal access to any products associated with the specified portfolio	Write	Portfolio* (p. 1643)		
AssociateProductWithPortfolio	Associates a product with a portfolio	Write			
AssociateServiceActionWithProvisioningArtifact	Associates an action with a provisioning artifact	Write	Product* (p. 1643)		
AssociateTagOptionWithPortfolio	Associate the specified Tag Option with the specified portfolio or product	Write	Portfolio (p. 1643)		
			Product (p. 1643)		
BatchAssociateServiceActionsWithProvisioningArtifact	Associates multiple self-service actions with provisioning artifacts.	Write			
BatchDisassociateServiceActionsFromProvisioningArtifact	Disassociates a batch of self-service actions from the specified provisioning artifact.	Write			
CopyProduct	Copies the specified source product to the specified target product or a new product.	Write			
CreateConstraint	Creates a constraint on an associated product and portfolio	Write	Product* (p. 1643)		
CreatePortfolio	Creates a portfolio	Write	Portfolio* (p. 1643)		
				aws:RequestTag / \${TagKey} (p. 1644)	
				aws:TagKeys (p. 1644)	
CreatePortfolioShare	Shares a portfolio you own with another AWS account	Permissions management	Portfolio* (p. 1643)		
CreateProduct	Creates a product and that product's first provisioning artifact	Write	Product* (p. 1643)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:RequestTag/ \${TagKey} (p. 1644) aws:TagKeys (p. 1644)	
CreateProvisionedProductPlan	Adds a new provisioned product	Write			
CreateProvisioningArtifact	Adds a new provisioning artifact to an existing product	Write	Product* (p. 1643)		
CreateServiceAction	Creates a self-service action.	Write			
CreateTagOption	Creates a TagOption.	Write			
DeleteConstraint	Removes and deletes an existing constraint from an associated product and portfolio	Write			
DeletePortfolio	Deletes a portfolio if all associations and shares have been removed from the portfolio	Write	Portfolio* (p. 1643)		
DeletePortfolioShare	Unshares a portfolio you own from an AWS account you previously shared the portfolio with	Permissions management	Portfolio* (p. 1643)		
DeleteProduct	Deletes a product if all associations have been removed from the product	Write	Product* (p. 1643)		
DeleteProvisionedProductPlan	Deletes a provisioned product	Write			
DeleteProvisioningArtifact	Deletes a provisioning artifact from a product	Write	Product* (p. 1643)		
DeleteServiceAction	Deletes a self-service action.	Write			
DeleteTagOption	Deletes the specified TagOption.	Write			
DescribeConstraint	Describes a constraint	Read			
DescribeCopyProduct	Gets the status of the specified product operation.	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribePortfolio	Describes a portfolio	Read	Portfolio* (p. 1643)		
DescribePortfolioShare	Gets the status of the specified PortfolioShare operation.	Read			
DescribeProduct	Describes a product as an end-user	Read	Product* (p. 1643)		
DescribeProductAsAdmin	Describes a product as an admin	Read	Product* (p. 1643)		
DescribeProductUser	Describes a product as an end-user	Read			
DescribeProvisionedProduct	Describes a provisioned product	Read			
DescribeProductPlan	Describes a provisioned product plan	Read			
DescribeProvisioningArtifact	Describes a provisioning artifact	Read	Product* (p. 1643)		
DescribeProvisioningParameters	Describes the parameters that you need to specify to successfully provision a specified provisioning artifact	Read	Product* (p. 1643)		
DescribeRecord	Describes a record and lists any outputs	Read		servicecatalog:accountLevel (p. 1644) servicecatalog:roleLevel (p. 1644) servicecatalog:userLevel (p. 1644)	
DescribeServiceAction	Describes a self-service action.	Read			
DescribeServiceActionExecution	Gets the default parameters if you executed the specified Service Action on the specified Provisioned Product.	Read			
DescribeTagOption	Gets information about the specified TagOption.	Read			
DisableAWSOrganizationsSharing	Disable portfolio sharing through AWS Organizations feature.	Write			
DisassociateBudgetFromResource	Disassociates a budget from a resource.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DisassociatePrincipalFromPortfolio	Disassociates an IAM principal from a portfolio.	Write	Portfolio* (p. 1643)		
DisassociateProductFromPortfolio	Disassociates a product from a portfolio.	Write			
DisassociateServiceActionFromArtifact	Disassociates the specified service action from the specified provisioning artifact.	Write	Product* (p. 1643)		
DisassociateTagOptionFromResource	Disassociates the specified tag option from the specified resource.	Write	Portfolio (p. 1643)		
	Product (p. 1643)				
EnableAWSOrganizations	Enable portfolio sharing feature through AWS Organizations.	Write			
ExecuteProvisioningPlan	Executes a provisioned product plan.	Write			
ExecuteProvisioningProductServiceAction	Executes a provisioned product plan.	Write			
GetAWSOrganizationStatus	Get the Access Status for AWS Organization portfolio share feature.	Read			
ListAcceptedPortfolios	Lists the portfolios that have been shared with you and you have accepted.	List			
ListBudgetsForResource	Lists all the budgets associated with your resource.	List			
ListConstraintsForPortfolio	Lists constraints associated with a given portfolio.	List			
ListLaunchPaths	Lists the different ways to launch a given product as an end-user.	List	Product* (p. 1643)		
ListOrganizationPathsThatHaveAccess	Lists the organization nodes that have access to the specified portfolio.	List			
ListPortfolioAccess	Lists the AWS accounts you have shared a given portfolio with.	List	Portfolio* (p. 1643)		
ListPortfolios	Lists the portfolios in your account.	List			
ListPortfoliosForProduct	Lists the portfolios associated with a given product.	List	Product* (p. 1643)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListPrincipalsForPortfolio	Lists the IAM principals associated with a given portfolio	List	Portfolio* (p. 1643)		
ListProvisionedProductPlans	Lists the provisioned product plans	List			
ListProvisioningArtifacts	Lists the provisioning artifacts associated with a given product	List	Product* (p. 1643)		
ListProvisioningArtifactsForAction	Lists all provisioning artifacts for the specified self-service action.	List			
ListRecordHistory	Lists all the records in your account or all the records related to a given provisioned product	List		servicecatalog:accountLevel (p. 1644) servicecatalog:roleLevel (p. 1644) servicecatalog:userLevel (p. 1644)	
ListResourcesForTagOption	Lists the resources associated with the specified TagOption.	List			
ListServiceActions	Lists all self-service actions.	List			
ListServiceActionsForArtifact	Lists all the service actions associated with the specified provisioning artifact in your account	List	Product* (p. 1643)		
ListStackInstancesStatus	Lists account, region and status of each stack instances that are associated with a CFN_STACKSET type provisioned product	List			
ListTagOptions	Lists the specified TagOptions or all TagOptions.	List			
ProvisionProduct	Provisions a product with a specified provisioning artifact and launch parameters	Write	Product* (p. 1643)		
RejectPortfolioShare	Rejects a portfolio that has been shared with you that you previously accepted	Write	Portfolio* (p. 1643)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ScanProvisionedProducts	Lists all the provisioned products in your account	List		servicecatalog:accountLevel (p. 1644) servicecatalog:roleLevel (p. 1644) servicecatalog:userLevel (p. 1644)	
SearchProducts	Lists the products available to you as an end-user	List			
SearchProductsAsAdmin	Lists all the products in your account or all the products associated with a given portfolio	List			
SearchProvisionedProducts	Lists all the provisioned products in your account	List		servicecatalog:accountLevel (p. 1644) servicecatalog:roleLevel (p. 1644) servicecatalog:userLevel (p. 1644)	
TerminateProvisionedProduct	Terminates an existing provisioned product	Write		servicecatalog:accountLevel (p. 1644) servicecatalog:roleLevel (p. 1644) servicecatalog:userLevel (p. 1644)	
UpdateConstraint	Updates the metadata fields of an existing constraint	Write			
UpdatePortfolio	Updates the metadata fields and/or tags of an existing portfolio	Write	Portfolio* (p. 1643)		
				aws:RequestTag/\${TagKey} (p. 1644) aws:TagKeys (p. 1644)	
UpdateProduct	Updates the metadata fields and/or tags of an existing product	Write	Product* (p. 1643)		
				aws:RequestTag/\${TagKey} (p. 1644) aws:TagKeys (p. 1644)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UpdateProvisionedProduct	Updates an existing provisioned product	Write		servicecatalog:accountLevel (p. 1644) servicecatalog:roleLevel (p. 1644) servicecatalog:userLevel (p. 1644)	
UpdateProvisionedExistingProduct	Updates the properties of an existing provisioned product	Write			
UpdateProvisioningArtifact	Updates the metadata fields of an existing provisioning artifact	Write	Product* (p. 1643)		
UpdateServiceAction	Updates a self-service action.	Write			
UpdateTagOption	Updates the specified TagOption.	Write			

Resource Types Defined by AWS Service Catalog

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 1636\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
Portfolio	arn:\${Partition}:catalog:\${Region}: \${Account}:portfolio/\${PortfolioId}	aws:ResourceTag/\${TagKey} (p. 1644)
Product	arn:\${Partition}:catalog:\${Region}: \${Account}:product/\${ProductId}	aws:ResourceTag/\${TagKey} (p. 1644)

Condition Keys for AWS Service Catalog

AWS Service Catalog defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Note

For example policies that show how these condition keys can be used in an IAM policy, see [Example Access Policies for Provisioned Product Management](#) in the *AWS Service Catalog Administrator Guide*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Filters actions based on the presence of tag key-value pairs in the request	String
aws:ResourceTag/\${TagKey}	Filters actions based on tag key-value pairs attached to the resource	String
aws:TagKeys	Filters actions based on the presence of tag keys in the request	String
servicecatalog:accountByAny	Allows users to see and perform actions on resources created by anyone in the account.	String
servicecatalog:roleLevel	Allows users to see and perform actions on resources created either by them or by anyone federating into the same role as them.	String
servicecatalog:userLevel	Allows users to see and perform actions on only resources that they created.	String

Actions, Resources, and Condition Keys for Service Quotas

Service Quotas (service prefix: `servicequotas`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Service Quotas \(p. 1644\)](#)
- [Resource Types Defined by Service Quotas \(p. 1646\)](#)
- [Condition Keys for Service Quotas \(p. 1646\)](#)

Actions Defined by Service Quotas

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AssociateServiceQuotaTemplate	Grants permission to associate the Service Quotas template with your organization	Write			
DeleteServiceQuotaTemplate	Grants permission to remove the specified service quota from the service quota template	Write			
DisassociateServiceQuotaTemplate	Grants permission to disassociate the Service Quotas template from your organization	Write			
GetAWSDefaultServiceQuota	Grants permission to return the details for the specified service quota, including the AWS default value	Read			
GetAssociationForServiceQuotaTemplate	Grants permission to retrieve the Service Quota Template Association Status value, which tells you if the Service Quotas template is associated with an organization	Read			
GetRequestedServiceQuotaIncrease	Grants permission to retrieve the details for a particular service quota increase request	Read			
GetServiceQuota	Grants permission to return the details for the specified service quota, including the applied value	Read			
GetServiceQuotaIncreaseRequest	Grants permission to retrieve the details for a particular service quota increase request from the service quota template	Read			
ListAWSDefaultServiceQuotas	Grants permission to list all default service quotas for the specified AWS service	Read			
ListRequestedServiceQuotaIncreases	Grants permission to request a list of the changes to quotas for a service	Read			
ListRequestedServiceQuotas	Grants permission to request a list of the changes to specific quota service quotas	Read			
ListServiceQuotaIncreaseRequests	Grants permission to return a list of the service quota increase requests from the service quota template	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListServiceQuotas	Grants permission to list all service quotas for the specified AWS service, in that account, in that Region	Read			
ListServices	Grants permission to list the AWS services available in Service Quotas	Read			
PutServiceQuota	Grants permission to define and address quota to the Service quota template	Write	quota (p. 1646)		
				servicequotas:service (p. 1646)	
RequestServiceQuota	Grants permission to submit the request for a service quota increase	Write	quota (p. 1646)		
				servicequotas:service (p. 1646)	

Resource Types Defined by Service Quotas

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 1644\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
quota	arn:\${Partition}:servicequotas:\${Region}:\${Account}: \${ServiceCode}/\${QuotaCode}	

Condition Keys for Service Quotas

Service Quotas defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
servicequotas:service	Filters or restricts access to a specified AWS service	string

Actions, Resources, and Condition Keys for Amazon SES

Amazon SES (service prefix: ses) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon SES \(p. 1647\)](#)
- [Resource Types Defined by Amazon SES \(p. 1654\)](#)
- [Condition Keys for Amazon SES \(p. 1655\)](#)

Actions Defined by Amazon SES

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CloneReceiptRule	Creates a receipt rule set by cloning an existing one	Write			
CreateConfigurationSet	Creates a new configuration set	Write			
CreateConfigurationSetDestination	Creates a configuration set event destination	Write			
CreateCustomVerificationTemplate	Creates an association between a configuration set and a custom domain for open and click event tracking	Write			
CreateCustomVerificationTemplate	Creates a new custom verification template	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateReceiptFilter	Creates a new IP address filter	Write			
CreateReceiptRule	Creates a receipt rule	Write			
CreateReceiptRuleSet	Creates an empty receipt rule set	Write			
CreateTemplate	Creates an email template	Write			
DeleteConfigurationSet	Deletes the configuration set	Write			
DeleteConfigurationSetEventDestination	Deletes a configuration set event destination	Write			
DeleteConfigurationSetOption	Deletes an association between a configuration set and a custom domain for open and click event tracking	Write			
DeleteCustomVerificationEmailTemplate	Deletes an existing custom verification email template	Write			
DeleteIdentity	Deletes the specified identity (an email address or a domain) from the list of verified identities	Write			
DeleteIdentityPolicy	Deletes the specified identity (an email address or a domain) from the list of verified identities	Write			
DeleteReceiptFilter	Deletes the specified IP address filter	Write			
DeleteReceiptRule	Deletes the specified receipt rule	Write			
DeleteReceiptRuleSet	Deletes the specified receipt rule set and all of the receipt rules it contains	Write			
DeleteTemplate	Deletes an email template	Write			
DeleteVerifiedEmailAddress	Deletes the specified email address from the list of verified addresses	Write			
DescribeActiveReceiptRuleSets	Returns the metadata and receipt rules for the receipt rule set that is currently active	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeConfigurationSets	Returns the details of the specified configuration set	Read			
DescribeReceiptRule	Returns the details of the specified receipt rule	Read			
DescribeReceiptRuleSet	Returns the details of the specified receipt rule set	Read			
GetAccountSending	Returns the email sending status of the Amazon SES account for the current region	Read			
GetCustomVerificationTemplate	Returns the custom email verification template for the template name you specify	Read			
GetIdentityDkimAttributes	Returns the current status of Easy DKIM signing for an entity	Read			
GetIdentityMailFromAttributes	Returns the custom MAIL FROM attributes for a list of identities (email addresses and/or domains)	Read			
GetIdentityNotificationAttributes	Given a list of verified identities (email addresses and/or domains), returns a structure describing identity notification attributes	Read			
GetIdentityPolicies	Returns the requested sending authorization policies for the given identity (an email address or a domain)	Read			
GetIdentityVerificationAttributes	Given a list of identities (email addresses and/or domains), returns the verification status and (for domain identities) the verification token for each identity	Read			
GetSendQuota	Returns the user's current sending limits	Read			
GetSendStatistics	Returns the user's sending statistics. The result is a list of data points, representing the last two weeks of sending activity	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetTemplate	Returns the template object (which includes the Subject line, HTML part and text part) for the template you specify	Read			
ListConfigurationSets	Returns a list of the configuration sets associated with your Amazon SES account in the current AWS Region	List			
ListCustomVerificationTemplates	Lists the existing custom verification templates for your account in the current AWS Region	List			
ListIdentities	Returns a list containing all of the identities (email addresses and domains) for your AWS account, regardless of verification status	List			
ListIdentityPolicies	Returns a list of sending authorization policies that are attached to the given identity (an email address or a domain)	List			
ListReceiptFilters	Lists the IP address filters associated with your AWS account	List			
ListReceiptRuleSets	Lists the receipt rule sets that exist under your AWS account	List			
ListTemplates	Lists the email templates present in your Amazon SES account in the current AWS Region	List			
ListVerifiedEmailAddresses	Returns a list containing all of the email addresses that have been verified	List			
PutIdentityPolicy	Adds or updates a sending authorization policy for the specified identity (an email address or a domain)	Write			
ReorderReceiptRuleSet	Reorders the receipt rules within a receipt rule set	Write			
SendBounce	Generates and sends a bounce message to the sender of an email you received through Amazon SES	Write		ses:FromAddress (p. 1655)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
SendBulkTemplateMultiple	Composes an email message to multiple destinations	Write	identity* (p. 1654)		
				ses:FeedbackAddress (p. 1655) ses:FromAddress (p. 1655) ses:FromDisplayName (p. 1655) ses:Recipients (p. 1655)	
SendCustomVerificationEmail	Adds an email address to the list of identities for your Amazon SES account in the current AWS Region and attempts to verify it	Write	identity* (p. 1654)		
				ses:FeedbackAddress (p. 1655) ses:FromAddress (p. 1655) ses:FromDisplayName (p. 1655) ses:Recipients (p. 1655)	
SendEmail	Composes an email message based on input data, and then immediately queues the message for sending	Write	identity* (p. 1654)		
				ses:FeedbackAddress (p. 1655) ses:FromAddress (p. 1655) ses:FromDisplayName (p. 1655) ses:Recipients (p. 1655)	
SendRawEmail	Sends an email message, with header and content specified by the client	Write	identity* (p. 1654)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				ses:FeedbackAddress (p. 1655) ses:FromAddress (p. 1655) ses:FromDisplayName (p. 1655) ses:Recipients (p. 1655)	
SendTemplatedEmail	Composes an email message using an email template and immediately queues it for sending	Write	identity* (p. 1654)		
			ses:FeedbackAddress (p. 1655) ses:FromAddress (p. 1655) ses:FromDisplayName (p. 1655) ses:Recipients (p. 1655)		
SetActiveReceiptRule	Sets the specified receipt rule set as the active receipt rule set	Write			
SetIdentityDkimEnabled	Enables or disables Easy DKIM signing of email sent from an identity	Write			
SetIdentityFeedbackForwardingDomain	Given an identity (an email address or a domain), enables or disables whether Amazon SES forwards bounce and complaint notifications as email	Write			
SetIdentityHeaderNotificationEnabled	Given an identity (an email address or a domain), sets whether Amazon SES includes the original email headers in the Amazon Simple Notification Service (Amazon SNS) notifications of a specified type	Write			
SetIdentityMailFromDomain	Enables or disables the custom MAILFROM domain setup for a verified identity (an email address or a domain)	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
SetIdentityNotificationTopic	Given an identity (an email address or a domain), sets the Amazon Simple Notification Service (Amazon SNS) topic to which Amazon SES will publish bounce, complaint, and/or delivery notifications for emails sent with that identity as the Source	Write			
SetReceiptRulePosition	Sets the position of the specified receipt rule in the receipt rule set	Write			
TestRenderTemplate	Creates a preview of the MIME content of an email when provided with a template and a set of replacement data	Write			
UpdateAccountSendingEnabled	Enables or disables email sending for your entire Amazon SES account in the current AWS Region	Write			
UpdateConfigurationSetEventDestination	Updates the event destination of a configuration set	Write			
UpdateConfigurationSetPublishingMetrics	Enables or disables the publishing of reputation metrics for emails sent using a specific configuration set in a given AWS Region	Write			
UpdateConfigurationSetSendingEnabled	Enables or disables email sending for messages sent using a specific configuration set in a given AWS Region	Write			
UpdateConfigurationSetTrackingEnabled	Modifies an association between a configuration set and a custom domain for open and click event tracking	Write			
UpdateCustomVerificationEmailTemplate	Updates an existing custom verification email template	Write			
UpdateReceiptRule	Updates a receipt rule	Write			
UpdateTemplate	Updates an email template	Write			
VerifyDomainDkim	Returns a set of DKIM tokens for a domain	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
VerifyDomainIdentity	Verifies a domain	Read			
VerifyEmailAddress	Verifies an email address. This action causes a confirmation email message to be sent to the specified address. This action is throttled at one request per second	Read			
VerifyEmailIdentity	Verifies an email address. This action causes a confirmation email message to be sent to the specified address. This action is throttled at one request per second	Read			

Resource Types Defined by Amazon SES

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 1647\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
configuration-set	arn:\${Partition}:ses:\${Region}: \${Account}:configuration-set/ \${ConfigurationSetName}	
custom-verification-email-template	arn:\${Partition}:ses: \${Region}: \${Account}:custom-verification-email-template/ \${CustomVerificationEmailTemplateName}	
event-destination	arn:\${Partition}:ses:\${Region}: \${Account}:configuration-set/ \${ConfigurationSetName}:event-destination/ \${EventDestinationName}	
identity	arn:\${Partition}:ses:\${Region}: \${Account}:identity/\${IdentityName}	
receipt-filter	arn:\${Partition}:ses:\${Region}: \${Account}:receipt-filter/ \${ReceiptFilterName}	
receipt-rule	arn:\${Partition}:ses:\${Region}: \${Account}:receipt-rule-set/ \${ReceiptRuleSetName}:receipt-rule/ \${ReceiptRuleName}	

Resource Types	ARN	Condition Keys
receipt-rule-set	arn:\${Partition}:ses:\${Region}: \${Account}:receipt-rule-set/ \${ReceiptRuleSetName}	
template	arn:\${Partition}:ses:\${Region}: \${Account}:template/\${TemplateName}	

Condition Keys for Amazon SES

Amazon SES defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
ses:FeedbackAddress	The "Return-Path" address, which specifies where bounces and complaints are sent by email feedback forwarding.	String
ses:FromAddress	The "From" address of a message.	String
ses:FromDisplayName	The "From" address that is used as the display name of a message.	String
ses:Recipients	The recipient addresses of a message, which include the "To", "CC", and "BCC" addresses.	String

Actions, Resources, and Condition Keys for Amazon Session Manager Message Gateway Service

Amazon Session Manager Message Gateway Service (service prefix: `ssmmessages`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Session Manager Message Gateway Service \(p. 1655\)](#)
- [Resource Types Defined by Amazon Session Manager Message Gateway Service \(p. 1656\)](#)
- [Condition Keys for Amazon Session Manager Message Gateway Service \(p. 1656\)](#)

Actions Defined by Amazon Session Manager Message Gateway Service

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually

allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the **Resource** element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateControlChannel	Registers a control channel for an instance to send control messages to Systems Manager service.	Write			
CreateDataChannel	Registers a data channel for an instance to send data messages to Systems Manager service.	Write			
OpenControlChannel	Opens a websocket connection for a registered control channel stream from an instance to Systems Manager service.	Write			
OpenDataChannel	Opens a websocket connection for a registered data channel stream from an instance to Systems Manager service.	Write			

Resource Types Defined by Amazon Session Manager Message Gateway Service

Amazon Session Manager Message Gateway Service does not support specifying a resource ARN in the **Resource** element of an IAM policy statement. To allow access to Amazon Session Manager Message Gateway Service, specify “**Resource**”: “*” in your policy.

Condition Keys for Amazon Session Manager Message Gateway Service

SSM Messages has no service-specific context keys that can be used in the **Condition** element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for AWS Shield

AWS Shield (service prefix: `shield`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Shield \(p. 1657\)](#)
- [Resource Types Defined by AWS Shield \(p. 1658\)](#)
- [Condition Keys for AWS Shield \(p. 1659\)](#)

Actions Defined by AWS Shield

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AssociateDRTLogBucket	Authorizes the DDoS Response team to access the specified Amazon S3 bucket containing your flow logs	Write			s3:GetBucketPolicy s3:PutBucketPolicy
AssociateDRTRole	Authorizes the DDoS Response team using the specified role, to access your AWS account to assist with DDoS attack mitigation during potential attacks	Write			iam:GetRole iam>ListAttachedRolePoli iam:PassRole
CreateProtection	Activate DDoS protection service for a given resource ARN	Write	protection* (p. 1659)		
CreateSubscription	Activate subscription	Write			
DeleteProtection	Delete an existing protection	Write	protection* (p. 1659)		
DeleteSubscription	Deactivate subscription	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeAttack	Get attack details	Read	attack* (p. 1659)		
DescribeDRTAccess	Returns the current role and list of Amazon S3 log buckets used by the DDoS Response team to access your AWS account while assisting with attack mitigation	Read			
DescribeEmergencyContact	Lists the email addresses that the DRT can use to contact you during a suspected attack	Read			
DescribeProtection	Get protection details	Read	protection* (p. 1659)		
DescribeSubscription	Get subscription details, such as start time	Read			
DisassociateDRTAccess	Removes the DDoS Response team's access to the specified Amazon S3 bucket containing your flow logs	Write			s3:DeleteBucketPolicy s3:GetBucketPolicy s3:PutBucketPolicy
DisassociateDRTRole	Removes the DDoS Response team's access to your AWS account	Write			
GetSubscriptionState	Get subscription state	Read			
ListAttacks	List all existing attacks	List			
ListProtections	List all existing protections	List			
UpdateEmergencyContact	Updates the details of the list of email addresses that the DRT can use to contact you during a suspected attack	Write			
UpdateSubscription	Updates the details of an existing subscription	Write			

Resource Types Defined by AWS Shield

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1657\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
attack	arn:\${Partition}:shield::\${Account}:attack/\${Id}	
protection	arn:\${Partition}:shield::\${Account}:protection/\${Id}	

Condition Keys for AWS Shield

Shield has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Amazon Simple Workflow Service

Amazon Simple Workflow Service (service prefix: `swf`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Simple Workflow Service \(p. 1659\)](#)
- [Resource Types Defined by Amazon Simple Workflow Service \(p. 1666\)](#)
- [Condition Keys for Amazon Simple Workflow Service \(p. 1666\)](#)

Actions Defined by Amazon Simple Workflow Service

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CancelTimer	Description for CancelTimer	Write	domain* (p. 1666)		
CancelWorkflowExecution	Description for CancelWorkflowExecution	Write	domain* (p. 1666)		
CompleteWorkflowExecution	Description for CompleteWorkflowExecution	Write	domain* (p. 1666)		
ContinueAsNewWorkflowExecution	Description for ContinueAsNewWorkflowExecution	Write	domain* (p. 1666)		
CountClosedWorkflowExecutions	Returns the number of closed workflow executions within the given domain that meet the specified filtering criteria.	Read	domain* (p. 1666)		
				swf:tagFilter.tag (p. 1667)	
				swf:typeFilter.name (p. 1667)	
				swf:typeFilter.version (p. 1667)	
CountOpenWorkflowExecutions	Returns the number of open workflow executions within the given domain that meet the specified filtering criteria.	Read	domain* (p. 1666)		
				swf:tagFilter.tag (p. 1667)	
				swf:typeFilter.name (p. 1667)	
				swf:typeFilter.version (p. 1667)	
CountPendingActivityTasks	Returns the estimated number of activity tasks in the specified task list.	Read	domain* (p. 1666)		
				swf:taskList.name (p. 1667)	
CountPendingDecisionTasks	Returns the estimated number of decision tasks in the specified task list.	Read	domain* (p. 1666)		
				swf:taskList.name (p. 1667)	
DeprecateActivityType	Deprecates the specified activity type.	Write	domain* (p. 1666)		
				swf:activityType.name (p. 1666)	
				swf:activityType.version (p. 1667)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeprecateDomain	Deprecates the specified domain.	Write	domain* (p. 1666)		
DeprecateWorkflowType	Deprecates the specified workflow type.	Write	domain* (p. 1666)		
				swf:workflowType.name (p. 1667)	
					swf:workflowType.version (p. 1667)
DescribeActivityType	Returns information about the specified activity type.	Read	domain* (p. 1666)		
				swf:activityType.name (p. 1666)	
					swf:activityType.version (p. 1667)
DescribeDomain	Returns information about the specified domain, including description and status.	Read	domain* (p. 1666)		
DescribeWorkflowExecution	Returns information about the specified workflow execution including its type and some statistics.	Read	domain* (p. 1666)		
DescribeWorkflowType	Returns information about the specified workflow type.	Read	domain* (p. 1666)		
				swf:workflowType.name (p. 1667)	
					swf:workflowType.version (p. 1667)
FailWorkflowExecution	Description for FailWorkflowExecution	Write	domain* (p. 1666)		
GetWorkflowExecutionHistory	Returns the history of the specified workflow execution.	Read	domain* (p. 1666)		
ListActivityTypes	Returns information about all activities registered in the specified domain that match the specified name and registration status.	List	domain* (p. 1666)		
ListClosedWorkflowExecutions	Returns a list of closed workflow executions in the specified domain that meet the filtering criteria.	List	domain* (p. 1666)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				swf:tagFilter.tag (p. 1667) swf:typeFilter.name (p. 1667) swf:typeFilter.version (p. 1667)	
ListDomains	Returns the list of domains registered in the account.	List			
ListOpenWorkflowExecutions	Returns a list of open workflow executions in the specified domain that meet the filtering criteria.	List	domain* (p. 1666)		
				swf:tagFilter.tag (p. 1667) swf:typeFilter.name (p. 1667) swf:typeFilter.version (p. 1667)	
ListTagsForResource	This action lists tags for an AWS SWF resource.	List	domain (p. 1666)		
ListWorkflowType	Returns information about workflow types in the specified domain.	List	domain* (p. 1666)		
PollForActivityTask	Used by workers to get an ActivityTask from the specified activity taskList.	Write	domain* (p. 1666)		
				swf:taskList.name (p. 1667)	
PollForDecisionTask	Used by deciders to get a DecisionTask from the specified decision taskList.	Write	domain* (p. 1666)		
				swf:taskList.name (p. 1667)	
RecordActivityTaskHeartbeat	Used by activity workers to report to the service that the ActivityTask represented by the specified taskToken is still making progress.	Write	domain* (p. 1666)		
RecordMarker	Description for RecordMarker	Write	domain* (p. 1666)		
RegisterActivityType	Registers a new activity type along with its configuration settings in the specified domain.	Write	domain* (p. 1666)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				swf:defaultTaskList.name (p. 1667) swf:name (p. 1667) swf:version (p. 1667)	
RegisterDomain	Registers a new domain.	Write		aws:TagKeys (p. 1666) aws:RequestTag/\${TagKey} (p. 1666)	
RegisterWorkflowType	Registers a new workflow type and its configuration settings in the specified domain.	Write	domain* (p. 1666)	swf:defaultTaskList.name (p. 1667) swf:name (p. 1667) swf:version (p. 1667)	
RequestCancelActivityTask	Description for RequestCancelActivityTask	Write	domain* (p. 1666)		
RequestCancelExternalWorkflowExecution	Description for RequestCancelExternalWorkflowExecution	Write	domain* (p. 1666)		
RequestCancelWorkflowExecution	Records a WorkflowExecutionCancelRequested event in the currently running workflow execution identified by the given domain, workflowId, and runId.	Write	domain* (p. 1666)		
RespondActivityTaskCanceled	Used by workers to tell the service that the ActivityTask identified by the taskToken was successfully canceled.	Write	domain* (p. 1666)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
RespondActivityTaskResult	Used by workers to tell the service that the ActivityTask identified by the taskToken completed successfully with a result (if provided).	Write	domain* (p. 1666)		swf:activityType.name (p. 1666) swf:activityType.version (p. 1667) swf:tagList.member.0 (p. 1667) swf:tagList.member.1 (p. 1667) swf:tagList.member.2 (p. 1667) swf:tagList.member.3 (p. 1667) swf:tagList.member.4 (p. 1667) swf:taskList.name (p. 1667) swf:workflowType.name (p. 1667) swf:workflowType.version (p. 1667)
RespondActivityTaskFailure	Used by workers to tell the service that the ActivityTask identified by the taskToken has failed with reason (if specified).	Write	domain* (p. 1666)		
RespondDecisionTaskSuccess	Used by deciders to tell the service that the DecisionTask identified by the taskToken has successfully completed.	Write	domain* (p. 1666)		
ScheduleActivityTask	Description for ScheduleActivityTask	Write	domain* (p. 1666)		
SignalExternalWorkflowExecution	Description for SignalExternalWorkflowExecution	Write	domain* (p. 1666)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
SignalWorkflowExecution	Records a <code>WorkflowExecutionSignaled</code> event in the workflow execution history and creates a decision task for the workflow execution identified by the given domain, workflowId and runId.	Write	domain* (p. 1666)		
StartChildWorkflowExecution	Description for <code>StartChildWorkflowExecution</code>	Write	domain* (p. 1666)		
StartTimer	Description for StartTimer	Write	domain* (p. 1666)		
StartWorkflowExecution	Starts an execution of the workflow type in the specified domain using the provided workflowId and input data.	Write	domain* (p. 1666)		
swf:tagList.member.0 (p. 1667)					
swf:tagList.member.1 (p. 1667)					
swf:tagList.member.2 (p. 1667)					
swf:tagList.member.3 (p. 1667)					
swf:tagList.member.4 (p. 1667)					
swf:taskList.name (p. 1667)					
swf:workflowType.name (p. 1667)					
TagResource	This action tags an AWS SWF resource.	Tagging	domain (p. 1666)		
aws:TagKeys (p. 1666) aws:RequestTag/\${TagKey} (p. 1666)					

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
TerminateWorkflowExecution	Records a WorkflowExecutionTerminated event and forces closure of the workflow execution identified by the given domain, runId, and workflowId.	Write	domain* (p. 1666)		
UntagResource	This action removes a tag from an AWS SWF resource.	Tagging	domain (p. 1666)		
				aws:TagKeys (p. 1666)	

Resource Types Defined by Amazon Simple Workflow Service

The following resource types are defined by this service and can be used in the **Resource** element of IAM permission policy statements. Each action in the [Actions table \(p. 1659\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
domain	arn:\${Partition}:swf::\${Account}:domain/\${DomainName}	aws:ResourceTag/\${TagKey} (p. 1666)

Condition Keys for Amazon Simple Workflow Service

Amazon Simple Workflow Service defines the following condition keys that can be used in the **Condition** element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Tag for request.	String
aws:ResourceTag/\${TagKey}	Tag for resource.	String
aws:TagKeys	Tag for key.	String
swf:activityType.name	Constrains the policy statement to only an activity type of the specified name.	String

Condition Keys	Description	Type
swf:activityType.version	Constrains the policy statement to only an activity type of the specified version.	String
swf:defaultTaskList.name	Constrains the policy statement to only requests that specify a matching defaultTaskList name.	String
swf:name	Constrains the policy statement to only activities or workflows with the specified name.	String
swf:tagFilter.tag	Constrains the policy statement to only requests that specify a matching tagFilter.tag value.	String
swf:tagList.member.0	Constrains the policy statement to only requests that contain the specified tag.	String
swf:tagList.member.1	Constrains the policy statement to only requests that contain the specified tag.	String
swf:tagList.member.2	Constrains the policy statement to only requests that contain the specified tag.	String
swf:tagList.member.3	Constrains the policy statement to only requests that contain the specified tag.	String
swf:tagList.member.4	Constrains the policy statement to only requests that contain the specified tag.	String
swf:taskList.name	Constrains the policy statement to only requests that specify a tasklist with the specified name.	String
swf:typeFilter.name	Constrains the policy statement to only requests that specify a type filter with the specified name.	String
swf:typeFilter.version	Constrains the policy statement to only requests that specify a type filter with the specified version.	String
swf:version	Constrains the policy statement to only activities or workflows with the specified version.	String
swf:workflowType.name	Constrains the policy statement to only a workflow of the specified type.	String
swf:workflowType.name	Constrains the policy statement to only requests that specify a workflow type of the specified name.	String
swf:workflowType.version	Constrains the policy statement to only requests that specify a workflow type of the specified version.	String

Actions, Resources, and Condition Keys for Amazon SimpleDB

Amazon SimpleDB (service prefix: sdb) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).

- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon SimpleDB \(p. 1668\)](#)
- [Resource Types Defined by Amazon SimpleDB \(p. 1669\)](#)
- [Condition Keys for Amazon SimpleDB \(p. 1669\)](#)

Actions Defined by Amazon SimpleDB

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
BatchDeleteAttributes	Performs multiple <code>DeleteAttributes</code> operations in a single call, which reduces round trips and latencies.	Write	domain* (p. 1669)		
BatchPutAttributes	With the <code>BatchPutAttributes</code> operation, you can perform multiple <code>PutAttribute</code> operations in a single call. With the <code>BatchPutAttributes</code> operation, you can perform multiple <code>PutAttribute</code> operations in a single call.	Write	domain* (p. 1669)		
CreateDomain	The <code>CreateDomain</code> operation creates a new domain.	Write	domain* (p. 1669)		
DeleteAttributes	Deletes one or more attributes associated with the item.	Write	domain* (p. 1669)		
DeleteDomain	The <code>DeleteDomain</code> operation deletes a domain.	Write	domain* (p. 1669)		
DomainMetadata	Returns information about the domain, including when the domain was created, the number	Read	domain* (p. 1669)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
	of items and attributes, and the size of attribute names and values.				
GetAttributes	Returns all of the attributes associated with the item.	Read	domain* (p. 1669)		
ListDomains	Description for ListDomains	List			
PutAttributes	The PutAttributes operation creates or replaces attributes in an item.	Write	domain* (p. 1669)		
Select	Description for Select	Read	domain* (p. 1669)		

Resource Types Defined by Amazon SimpleDB

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1668\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
domain	<code>arn:\${Partition}:sdb:\${Region}:\${Account}:domain/\${DomainName}</code>	

Condition Keys for Amazon SimpleDB

SimpleDB has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for AWS Snowball

AWS Snowball (service prefix: `snowball`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).

Topics

- [Actions Defined by AWS Snowball \(p. 1670\)](#)
- [Resource Types Defined by AWS Snowball \(p. 1671\)](#)

- [Condition Keys for AWS Snowball \(p. 1671\)](#)

Actions Defined by AWS Snowball

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CancelCluster	Cancels a cluster job.	Write			
CancelJob	Cancels the specified job.	Write			
CreateAddress	Creates an address for a Snowball to be shipped to.	Write			
CreateCluster	Creates an empty cluster.	Write			
CreateJob	Creates a job to import or export data between Amazon S3 and your on-premises data center.	Write			
DescribeAddress	Takes an AddressId and returns specific details about that address in the form of an Address object.	Read			
DescribeAddressees	Returns a specified number of ADDRESS objects.	List			
DescribeCluster	Returns information about a specific cluster including shipping information, cluster status, and other important metadata.	Read			
DescribeJob	Returns information about a specific job including shipping information, job status, and other important metadata.	Read			
GetJobManifest	Returns a link to an Amazon S3 presigned URL for the manifest	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
	file associated with the specified JobId value.				
GetJobUnlockCode	Returns the UnlockCode code value for the specified job.	Read			
GetSnowballUsage	Returns information about the Snowball service limit for your account, and also the number of Snowballs your account has in use.	Read			
ListClusterJobs	Returns an array of JobListEntry objects of the specified length.	List			
ListClusters	Returns an array of ClusterListEntry objects of the specified length.	List			
ListJobs	Returns an array of JobListEntry objects of the specified length.	List			
UpdateCluster	While a cluster's ClusterState value is in the AwaitingQuorum state, you can update some of the information associated with a cluster.	Write			
UpdateJob	While a job's JobState value is New, you can update some of the information associated with a job.	Write			

Resource Types Defined by AWS Snowball

AWS Snowball does not support specifying a resource ARN in the `Resource` element of an IAM policy statement. To allow access to AWS Snowball, specify “`Resource`”: “`*`” in your policy.

Condition Keys for AWS Snowball

Snowball has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Amazon SNS

Amazon SNS (service prefix: `sns`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon SNS \(p. 1672\)](#)
- [Resource Types Defined by Amazon SNS \(p. 1675\)](#)
- [Condition Keys for Amazon SNS \(p. 1675\)](#)

Actions Defined by Amazon SNS

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AddPermission	Adds a statement to a topic's access control policy, granting access for the specified AWS accounts to the specified actions.	Permissions management	topic* (p. 1675)		
CheckIfPhoneNumberIsOptedOut	Accepts a phone number and indicates whether the phone holder has opted out of receiving SMS messages from your account.	Read			
ConfirmSubscription	Verifies an endpoint owner's intent to receive messages by validating the token sent to the endpoint by an earlier <code>Subscribe</code> action.	Write	topic* (p. 1675)		
CreatePlatformApplication	Creates a platform application object for one of the supported push notification services, such as APNS and GCM, to which devices and mobile apps may register.	Write			
CreatePlatformEndpoint	Creates an endpoint for a device endpoint on one of the supported push notification services, such as GCM and APNS.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateTopic	Creates a topic to which notifications can be published.	Write	topic* (p. 1675)		
DeleteEndpoint	Deletes the endpoint for a device and mobile app from Amazon SNS.	Write			
DeletePlatformApplication	Deletes a platform application object for one of the supported push notification services, such as APNS and GCM.	Write			
DeleteTopic	Deletes a topic and all its subscriptions.	Write	topic* (p. 1675)		
GetEndpointAttributes	Retrieves the endpoint attributes for a device on one of the supported push notification services, such as GCM and APNS.	Read			
GetPlatformApplication	Retrieves the attributes of the platform application object for the supported push notification services, such as APNS and GCM.	Read			
GetSMSAttributes	Returns the settings for sending SMS messages from your account.	Read			
GetSubscriptionAttributes	Returns all of the properties of a subscription.	Read			
GetTopicAttributes	Returns all of the properties of a topic. Topic properties returned might differ based on the authorization of the user.	Read	topic* (p. 1675)		
ListEndpointsByPlatform	Lists the endpoints and endpoint attributes for devices in a supported push notification service, such as GCM and APNS.	List			
ListPhoneNumbers	Returns a list of phone numbers that have Opted out, meaning you cannot send SMS messages to them.	Read			
ListPlatformApplications	Lists the platform application objects for the supported push notification services, such as APNS and GCM.	List			
ListSubscriptions	Returns a list of the requester's subscriptions.	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListSubscriptionsByTopic	Returns a list of the Subscriptions to a specific topic.	List	topic* (p. 1675)		
ListTagsForResource	List all tags added to the specified Amazon SNS topic.	Read	topic (p. 1675)		
ListTopics	Returns a list of the requester's topics. Each call returns a limited list of topics, up to 100.	List			
OptInPhoneNumber	Opt in a phone number that is currently opted out, which enables you to resume sending SMS messages to the number.	Write			
Publish	Sends a message to all of a topic's subscribed endpoints.	Write	topic* (p. 1675)		
RemovePermission	Removes a statement from a topic's access control policy.	Permissions management	topic* (p. 1675)		
SetEndpointAttributes	Sets the attributes for an endpoint for a device on one of the supported push notification services, such as GCM and APNS.	Write			
SetPlatformApplicationAttributes	Sets the attributes of the platform application object for the supported push notification services, such as APNS and GCM.	Write			
SetSubscriptionAttributes	Allows a subscription owner to set a SetAttribute attribute of the topic to a new value.	Write			
SetTopicAttribute	Allows a topic owner to set an SetAttribute attribute of the topic to a new value.	Write	topic* (p. 1675)		
Subscribe	Prepares to subscribe an endpoint by sending the endpoint a confirmation message.	Write	topic* (p. 1675)	sns:Endpoint (p. 1676)	sns:Protocol (p. 1676)
TagResource	Add tags to the specified Amazon SNS topic.	Tagging	topic (p. 1675)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:RequestTag/\${TagKey} (p. 1676) aws:TagKeys (p. 1676)	
Unsubscribe	Deletes a subscription. If the subscription requires authentication for deletion, only the owner of the subscription or the topic's owner can unsubscribe, and an AWS signature is required.	Write			
UntagResource	Remove tags from the specified Amazon SNS topic.	Tagging	topic (p. 1675)		
				aws:RequestTag/\${TagKey} (p. 1676) aws:TagKeys (p. 1676)	

Resource Types Defined by Amazon SNS

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1672\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
topic	<code>arn:\${Partition}:sns:\${Region}:\${Account}: \${TopicName}</code>	

Condition Keys for Amazon SNS

Amazon SNS defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Tags from request	String
aws:TagKeys	Tag keys from request	String
sns:Endpoint	The URL, email address, or ARN from a Subscribe request or a previously confirmed subscription.	String
sns:Protocol	The protocol value from a Subscribe request or a previously confirmed subscription.	String

Actions, Resources, and Condition Keys for Amazon SQS

Amazon SQS (service prefix: `sqs`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon SQS \(p. 1676\)](#)
- [Resource Types Defined by Amazon SQS \(p. 1678\)](#)
- [Condition Keys for Amazon SQS \(p. 1678\)](#)

Actions Defined by Amazon SQS

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AddPermission	Adds a permission to a queue for a specific principal.	Permissions management	queue* (p. 1678)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ChangeMessageVisibility	Changes the visibility timeout of a specified message in a queue to a new value.	Write	queue* (p. 1678)		
ChangeMessageVisibilityBatch	Changes the visibility timeout of multiple messages.	Write	queue* (p. 1678)		
CreateQueue	Creates a new queue, or returns the URL of an existing one.	Write	queue* (p. 1678)		
DeleteMessage	Deletes the specified message from the specified queue.	Write	queue* (p. 1678)		
DeleteMessageBatch	Deletes up to ten messages from the specified queue.	Write	queue* (p. 1678)		
DeleteQueue	Deletes the queue specified by the queue URL, regardless of whether the queue is empty.	Write	queue* (p. 1678)		
GetQueueAttributes	Gets attributes for the specified queue.	Read	queue* (p. 1678)		
GetQueueUrl	Returns the URL of an existing queue.	Read	queue* (p. 1678)		
ListDeadLetterSourcequeues	Returns a list of your queues that have the RedrivePolicy queue attribute configured with a dead letter queue.	Read	queue* (p. 1678)		
ListQueueTags	Lists tags added to an SQS queue.	Read	queue* (p. 1678)		
ListQueues	Returns a list of your queues.	List			
PurgeQueue	Deletes the messages in a queue specified by the queue URL.	Write	queue* (p. 1678)		
ReceiveMessage	Retrieves one or more messages, with a maximum limit of 10 messages, from the specified queue.	Read	queue* (p. 1678)		
RemovePermission	Revokes any permissions in the queue policy that matches the specified Label parameter.	Permissions management	queue* (p. 1678)		
SendMessage	Delivers a message to the specified queue.	Write	queue* (p. 1678)		
SendMessageBatch	Delivers up to ten messages to the specified queue.	Write	queue* (p. 1678)		
SetQueueAttributes	Sets the value of one or more queue attributes.	Write	queue* (p. 1678)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
TagQueue	Add tags to the specified SQS queue.	Tagging	queue* (p. 1678)		
UntagQueue	Remove tags from the specified SQS queue.	Tagging	queue* (p. 1678)		

Resource Types Defined by Amazon SQS

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 1676\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Note

The ARN of the queue is used only in IAM permission policies. In API and CLI calls, you use the queue's URL instead.

Resource Types	ARN	Condition Keys
queue	<code>arn:\${Partition}:sns:\${Region}:\${Account}: \${QueueName}</code>	

Condition Keys for Amazon SQS

SQS has no service-specific context keys that can be used in the Condition element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for AWS SSO

AWS SSO (service prefix: sso) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS SSO \(p. 1679\)](#)
- [Resource Types Defined by AWS SSO \(p. 1682\)](#)
- [Condition Keys for AWS SSO \(p. 1682\)](#)

Actions Defined by AWS SSO

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AssociateDirectory	Connect a directory to be used by AWS Single Sign-On	Write			
AssociateProfile	Create an association between a directory user or group and a profile	Write			
CreateApplication	Add an application instance to AWS Single Sign-On	Write			
CreateApplicationInstance	Add a new certificate for an application instance	Write			
CreateManagedApplicationInstance	Add a managed application instance to AWS Single Sign-On	Write			
CreatePermissionSet	Create a permission set	Write			
CreateProfile	Create a profile for an application instance	Write			
CreateTrust	Create a federation trust in a target account	Write			
DeleteApplicationInstance	Delete the application instance	Write			
DeleteApplicationInstanceCertificate	Delete an inactive or expired certificate from the application instance	Write			
DeleteManagedApplicationInstance	Delete the managed application instance	Write			
DeletePermissionSet	Delete a permission set	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeletePermissionSet	Delete the permission policy associated with a permission set	Write			
DeleteProfile	Delete the profile for an application instance	Write			
DescribePermissionSets	Retrieve all the permissions policies associated with a permission set	Read			
DisassociateDirectory	Disassociate a directory to be used by AWS Single Sign-On	Write			
DisassociateProfile	Disassociate a directory user or group from a profile	Write			
GetApplicationInstance	Retrieve details for an application instance	Read			
GetApplicationTemplateDetails	Retrieve application template details	Read			
GetManagedApplicationInstances	Retrieve details for an application instance	Read			
GetMfaDeviceManagementSettings	Retrieve Mfa Device Management Settings for the directory	Read			
GetPermissionSet	Retrieve details of a permission set	Read			
GetPermissionsPolicies	Retrieve all permission policies associated with a permission set	Read			sso:DescribePermissionsPolicy
GetProfile	Retrieve a profile for an application instance	Read			
GetSSOStatus	Check if AWS Single Sign-On is enabled	Read			
GetSharedSsoConfiguration	Retrieve shared configuration for the current SSO instance	Read			
GetSsoConfiguration	Retrieve configuration for the current SSO instance	Read			
GetTrust	Retrieve the federation trust in a target account	Read			
ImportApplication	Update the application instance by uploading an Application metadata SAML metadata file provided by the service provider	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListApplicationInstances	Retrieve all of the certificates for a given application instance	Read			
ListApplicationInstances	Retrieve all application instances	List			sso:GetApplicationInstances
ListApplicationTemplates	Retrieve all supported application templates	Read			sso:GetApplicationTemplates
ListApplications	Retrieve all supported applications	Read			
ListDirectoryAssociations	Retrieve details about the directory connected to AWS Single Sign-On	Read			
ListPermissionSets	Retrieve all permission sets	Read			
ListProfileAssociations	Retrieve the directory user or group associated with the profile	Read			
ListProfiles	Retrieve all profiles for an application instance	Read			sso:GetProfile
PutMfaDeviceManagementSettings	Put MFA Device Management settings for the directory	Write			
PutPermissionsPolicy	Add a policy to a permission set	Write			
StartSSO	Initialize AWS Single Sign-On	Write			
UpdateApplicationForThisApplicationInstance	Set a certificate as the active one for this application instance	Write			
UpdateApplicationInstanceData	Update display data of an application instance	Write			
UpdateApplicationConfigurationForTheApplication	Update federation response configuration for the application instance	Write			
UpdateApplicationSchemaConfigurationForTheConfiguration	Update federation response schema configuration for the configuration application instance	Write			
UpdateApplicationSecurityConfigurationForTheInstance	Update security details for the application instance configuration	Write			
UpdateApplicationServiceProviderConfigurationForTheApplication	Update service provider related configuration for the application instance	Write			
UpdateApplicationStatus	Update the status of an application instance	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UpdateDirectoryAttributeMappings	Update the user attribute mappings for your connected directory	Write			
UpdateManagedApplicationInstanceStatus	Update the status of a managed application instance	Write			
UpdatePermissionSet	Update the permission set.	Write			
UpdateProfile	Update the profile for an application instance	Write			
UpdateSSOConfiguration	Update the configuration for the current SSO instance	Write			
UpdateTrust	Update the federation trust in a target account	Write			

Resource Types Defined by AWS SSO

AWS SSO does not support specifying a resource ARN in the `Resource` element of an IAM policy statement. To allow access to AWS SSO, specify “Resource”: “*” in your policy.

Condition Keys for AWS SSO

SSO has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for AWS SSO Directory

AWS SSO Directory (service prefix: `sso-directory`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS SSO Directory \(p. 1682\)](#)
- [Resource Types Defined by AWS SSO Directory \(p. 1685\)](#)
- [Condition Keys for AWS SSO Directory \(p. 1685\)](#)

Actions Defined by AWS SSO Directory

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases,

a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the Resource element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AddMemberToGroup	Adds member to the group in the directory that AWS SSO provides by default	Write			
CompleteVirtualMfaVirtualMFADevice	Completes the creation process of a virtual MFA device	Write			
CreateAlias	Creates an alias for the directory that AWS SSO provides by default	Write			
CreateBearerToken	Creates a bearer token for a given provisioning tenant.	Write			
CreateExternalIdProviderConfiguration	Create an External Identity Provider configuration for the directory	Write			
CreateGroup	Creates a group in the directory that AWS SSO provides by default	Write			
CreateProvisioningTenant	Creates a provisioning tenant for a given directory.	Write			
CreateUser	Creates a user in the directory that AWS SSO provides by default	Write			
DeleteBearerToken	Deletes the bearer token.	Write			
DeleteExternalIdProviderConfiguration	Delete an External Identity Provider configuration associated with the directory	Write			
DeleteGroup	Deletes a group from the directory that AWS SSO provides by default	Write			
DeleteMfaDevice	Deletes a MFA device by device name for a given user	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteProvisioningTenant	Deletes the provisioning tenant.	Write			
DeleteUser	Deletes a user from the directory that AWS SSO provides by default	Write			
DescribeDirectory	Retrieve information about the directory that AWS SSO provides by default	Read			
DescribeGroups	Retrieves information about group from the directory that AWS SSO provides by default	List			
DescribeUsers	Retrieves information about user from the directory that AWS SSO provides by default	List			
DisableExternalIdPUsersWithAnExternalIdentity	Disable authentication of end users with an external identity provider	Write			
DisableUser	Deactivates user in the directory that AWS SSO provides by default	Write			
EnableExternalIdPUsersWithAnExternalIdentity	Enable authentication of end users with an external identity provider	Write			
EnableUser	Activates user in the directory that AWS SSO provides by default	Write			
GetAWSSPConfigProviderConfigurations	Retrieve the AWS SSO Service Provider Configuration for the directory	Read			
ListBearerTokens	Lists bearer tokens for a given provisioning tenant.	List			
ListExternalIdPConfigurations	List all the External Identity Provider configurations created for the directory	List			
ListGroupForUser	Lists groups for a user from the directory that AWS SSO provides by default	List			
ListMembersInGroup	Retrieves all members that are part of the group in the directory that AWS SSO provides by default	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListMfaDevicesForUser	Lists all active MFA devices and their MFA device metadata for a user	List			
ListProvisioningTenants	Lists provisioning tenants for a given directory.	List			
RemoveMemberFromGroup	Removes member that are part of the group in the directory that AWS SSO provides by default	Write			
SearchGroups	Search for groups within the associated directory	Read			
SearchUsers	Search for users within the associated directory	Read			
StartVirtualMfaDevice	Begins the creation process of a virtual MFA device	Write			
UpdateExternalIdProviderConfiguration	Update an External Identity Provider configuration for a directory associated with the directory	Write			
UpdateGroup	Updates information about a group in the directory that AWS SSO provides by default	Write			
UpdatePassword	Updates password by sending a password reset link via email or generating one time password for a user in the directory that AWS SSO provides by default	Write			
UpdateUser	Updates user information in the directory that AWS SSO provides by default	Write			
VerifyEmail	Verify email address of an User	Write			

Resource Types Defined by AWS SSO Directory

AWS SSO Directory does not support specifying a resource ARN in the Resource element of an IAM policy statement. To allow access to AWS SSO Directory, specify “Resource”: “*” in your policy.

Condition Keys for AWS SSO Directory

SSO Directory has no service-specific context keys that can be used in the Condition element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for AWS Step Functions

AWS Step Functions (service prefix: `states`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Step Functions \(p. 1686\)](#)
- [Resource Types Defined by AWS Step Functions \(p. 1688\)](#)
- [Condition Keys for AWS Step Functions \(p. 1689\)](#)

Actions Defined by AWS Step Functions

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateActivity	Creates an activity. Activities must poll Step Functions using the <code>GetActivityTask</code> and respond using <code>SendTask*</code> API calls.	Tagging	activity* (p. 1689)		
				aws:RequestTag/ {\$TagKey} (p. 1689)	
CreateStateMachine	Creates a state machine.	Tagging	statemachine* (p. 1689)		
				aws:RequestTag/ {\$TagKey} (p. 1689)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:TagKeys (p. 1689)	
DeleteActivity	Deletes an activity.	Write	activity* (p. 1689)		
DeleteStateMachine	Deletes a state machine.	Write	statemachine* (p. 1689)		
DescribeActivity	Describes an activity.	Read	activity* (p. 1689)		
DescribeExecution	Describes an execution.	Read	execution* (p. 1689)		
DescribeStateMachine	Describes a state machine.	Read	statemachine* (p. 1689)		
DescribeStateMachineForExecution	Describes state machine for an execution	Read	execution* (p. 1689)		
GetActivityTask	Used by workers to retrieve a task (with the specified activity ARN) which has been scheduled for execution by a running state machine.	Write	activity* (p. 1689)		
GetExecutionHistory	Returns the history of the specified execution as a list of events. By default, the results are returned in ascending order of the timeStamp of the events.	Read	execution* (p. 1689)		
ListActivities	Lists the existing activities. The results may be split into multiple pages.	List			
ListExecutions	Lists the executions of a state machine that meet the filtering criteria. The results may be split into multiple pages.	Read	statemachine* (p. 1689)		
ListStateMachines	Lists the existing state machines. The results may be split into multiple pages.	List			
ListTagsForResource	This action lists tags for an AWS Step Functions resource.	Read	activity (p. 1689) statemachine (p. 1689)		
SendTaskFailure	Used by workers to report that the task identified by the taskToken failed.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions	
SendTaskHeartbeat	Used by workers to report to the service that the task represented by the specified taskToken is still making progress.	Write				
SendTaskSuccess	Used by workers to report that the task identified by the taskToken completed successfully.	Write				
StartExecution	Starts a state machine execution.	Write	statemachine* (p. 1689)			
StopExecution	Stops an execution.	Write				
TagResource	This action tags an AWS Step Functions resource.	Tagging	activity (p. 1689)			
			statemachine (p. 1689)			
				aws:TagKeys (p. 1689)	aws:RequestTag/ {\$TagKey} (p. 1689)	
UntagResource	This action removes a tag from an AWS Step Functions resource.	Tagging	activity (p. 1689)			
			statemachine (p. 1689)			
				aws:TagKeys (p. 1689)		
UpdateStateMachine	Updates a state machine.	Write	statemachine* (p. 1689)			
				aws:RequestTag/ {\$TagKey} (p. 1689)		
				aws:TagKeys (p. 1689)		

Resource Types Defined by AWS Step Functions

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 1686\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
activity	arn:\${Partition}:states:\${Region}: \${Account}:activity:\${ActivityName}	aws:ResourceTag/\${TagKey} (p. 1689)
execution	arn:\${Partition}:states:\${Region}: \${Account}:execution:\${StateMachineName}: \${ExecutionId}	
statemachine	arn:\${Partition}:states:\${Region}: \${Account}:stateMachine:\${StateMachineName}	aws:ResourceTag/\${TagKey} (p. 1689)

Condition Keys for AWS Step Functions

AWS Step Functions defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Tag for request	String
aws:ResourceTag/\${TagKey}	Tag for resource	String
aws:TagKeys	Tag for key	String

Actions, Resources, and Condition Keys for Amazon Storage Gateway

Amazon Storage Gateway (service prefix: `storagegateway`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Storage Gateway \(p. 1690\)](#)
- [Resource Types Defined by Amazon Storage Gateway \(p. 1697\)](#)
- [Condition Keys for Amazon Storage Gateway \(p. 1698\)](#)

Actions Defined by Amazon Storage Gateway

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ActivateGateway	This operation activates the gateway you previously deployed on your host.	Write		aws:RequestTag/\${TagKey} (p. 1698) aws:TagKeys (p. 1698)	
AddCache	This operation configures one or more gateway local disks as cache for a cached-volume gateway.	Write	gateway* (p. 1697)		
AddTagsToResource	This operation adds one or more tags to the specified resource.	Tagging	gateway (p. 1697)		
			share (p. 1697)		
			tape (p. 1697)		
			volume (p. 1698)		
				aws:RequestTag/\${TagKey} (p. 1698) aws:TagKeys (p. 1698)	
AddUploadBuffer	This operation configures one or more gateway local disks as upload buffer for a specified gateway.	Write	gateway* (p. 1697)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AddWorkingStorage	This operation configures one or more gateway local disks as working storage for a gateway.	Write	gateway* (p. 1697)		
AttachVolume	This operation connects a volume to an iSCSI connection and then attaches the volume to the specified gateway.	Write	gateway* (p. 1697)		
			volume* (p. 1698)		
CancelArchival	Cancels archiving of a virtual tape to the virtual tape shelf (VTS) after the archiving process is initiated.	Write	gateway* (p. 1697)		
			tape* (p. 1697)		
CancelRetrieval	Cancels retrieval of a virtual tape from the virtual tape shelf (VTS) to a gateway after the retrieval process is initiated.	Write	gateway* (p. 1697)		
			tape* (p. 1697)		
CreateCachediSCSIVolume	This operation creates a cached volume on a specified cached gateway. This operation is supported only for the gateway-cached volume architecture.	Write	gateway* (p. 1697)		
			volume* (p. 1698)		
				aws:RequestTag/ \${TagKey} (p. 1698)	
CreateNFSFileShare	This operation creates a NFS file share on an existing file gateway.	Write	gateway* (p. 1697)		
				aws:RequestTag/ \${TagKey} (p. 1698)	
				aws:TagKeys (p. 1698)	
CreateSMBFileShare	This operation creates a SMB file share on an existing file gateway.	Write	gateway* (p. 1697)		
				aws:RequestTag/ \${TagKey} (p. 1698)	
				aws:TagKeys (p. 1698)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateSnapshot	This operation initiates a snapshot of a volume.	Write	volume* (p. 1698)		
CreateSnapshotFromGatewayRecoveryPoint	This operation initiates a snapshot of a gateway from a volume recovery point.	Write	volume* (p. 1698)		
CreateStorediSCSOnGateway	This operation creates a volume on a specified gateway .	Write	gateway* (p. 1697)		
				aws:RequestTag/\${TagKey} (p. 1698)	
				aws:TagKeys (p. 1698)	
CreateTapeWithBarcode	Creates a virtual tape by using your own barcode.	Write	gateway* (p. 1697)		
				aws:RequestTag/\${TagKey} (p. 1698)	
				aws:TagKeys (p. 1698)	
CreateTapes	Creates one or more virtual tapes. You write data to the virtual tapes and then archive the tapes.	Write	gateway* (p. 1697)		
				aws:RequestTag/\${TagKey} (p. 1698)	
				aws:TagKeys (p. 1698)	
DeleteBandwidthLimits	This operation deletes the bandwidth rate limits of a gateway.	Write	gateway* (p. 1697)		
DeleteChapCredentials	This operation deletes Challenge-Handshake Authentication Protocol (CHAP) credentials for a specified iSCSI target and initiator pair.	Write	target* (p. 1697)		
DeleteFileShare	This operation deletes a file share from a file gateway.	Write	share* (p. 1697)		
DeleteGateway	This operation deletes a gateway.	Write	gateway* (p. 1697)		
DeleteSnapshot	This operation deletes a snapshot of a volume.	Write	volume* (p. 1698)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteTape	Deletes the specified virtual tape.	Write	gateway* (p. 1697)		
			tape* (p. 1697)		
DeleteTapeArchive	Deletes the specified virtual tape from the virtual tape shelf (VTS).	Write			
DeleteVolume	This operation deletes the specified gateway volume that you previously created using the <code>CreateCachediSCSIVolume</code> or <code>CreateStorediSCSIVolume</code> API.	Write	volume* (p. 1698)		
DescribeBandwidthLimits	This operation returns the bandwidth limits of a gateway.	Read	gateway* (p. 1697)		
DescribeCache	This operation returns information about the cache of a gateway. This operation is supported only for the gateway-cached volume architecture.	Read	gateway* (p. 1697)		
DescribeCachediSCSIVolumes	This operation returns descriptions of the gateway volumes specified in the request. This operation is supported only for the gateway-cached volume architecture.	Read	volume* (p. 1698)		
DescribeChapCredentials	This operation returns an array of Challenge-Handshake Authentication Protocol (CHAP) credentials information for a specified iSCSI target, one for each target-initiator pair.	Read	target* (p. 1697)		
DescribeGatewayMetadata	This operation returns metadata about a gateway such as its name, network interfaces, configured time zone, and the state (whether the gateway is running or not).	Read	gateway* (p. 1697)		
DescribeMaintenanceWindowTime	This operation returns your gateway's weekly maintenance start time including the day and time of the week.	Read	gateway* (p. 1697)		
DescribeNFSFileShares	This operation gets a description for one or more file shares from a file gateway.	Read	share* (p. 1697)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeSMBFileShares	This operation gets a description for one or more file shares from a file gateway.	Read	share* (p. 1697)		
DescribeSMBSettings	This operation gets a description of a Server Message Block (SMB) file share settings from a file gateway.	Read	gateway* (p. 1697)		
DescribeSnapshotSchedule	This operation describes the snapshot schedule for the specified gateway volume.	Read	volume* (p. 1698)		
DescribeStorediSCSIVolumes	This operation returns the description of the gateway volumes specified in the request.	Read	volume* (p. 1698)		
DescribeTapeArchives	Returns a description of specified virtual tapes in the virtual tape shelf (VTS).	Read			
DescribeTapeRecoveryPoints	Returns a list of virtual tape recovery points that are available for the specified gateway-VTL.	Read	gateway* (p. 1697)		
DescribeTapes	Returns a description of the specified Amazon Resource Name (ARN) of virtual tapes.	Read	gateway* (p. 1697)		
DescribeUploadBuffer	This operation returns information about the upload buffer of a gateway.	Read	gateway* (p. 1697)		
DescribeVTLDevices	Returns a description of virtual tape library (VTL) devices for the specified gateway.	Read	gateway* (p. 1697)		
DescribeWorkingStorage	This operation returns information about the working storage of a gateway.	Read	gateway* (p. 1697)		
DetachVolume	This operation disconnects a volume from an iSCSI connection and then detaches the volume from the specified gateway.	Write	volume* (p. 1698)		
DisableGateway	Disables a gateway when the gateway is no longer functioning.	Write	gateway* (p. 1697)		
JoinDomain	This operation enables you to join an Active Directory Domain.	Write	gateway* (p. 1697)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListFileShares	This operation gets a list of the file shares for a specific file gateway, or the list of file shares that belong to the calling user account.	List	gateway* (p. 1697)		
ListGateways	This operation lists gateways owned by an AWS account in a region specified in the request. The returned list is ordered by gateway Amazon Resource Name (ARN).	List			
ListLocalDisks	This operation returns a list of the gateway's local disks.	List	gateway* (p. 1697)		
ListTagsForResource	This operation lists the tags that have been added to the specified resource.	Read	gateway (p. 1697)		
			share (p. 1697)		
			tape (p. 1697)		
			volume (p. 1698)		
ListTapes	Lists virtual tapes in your virtual tape library (VTL) and your virtual tape shelf (VTS).	Read	tape* (p. 1697)		
ListVolumeInitiators	This operation lists iSCSI initiators that are connected to a volume.	Read	volume* (p. 1698)		
ListVolumeRecoveryPoints	This operation lists the recovery points for a specified gateway.	List	gateway* (p. 1697)		
ListVolumes	This operation lists the iSCSI stored volumes of a gateway.	List	gateway* (p. 1697)		
NotifyWhenUploadsComplete	This action sends you a notification through CloudWatch Events when all files written to your NFS file share have been uploaded to Amazon S3.	Write	share* (p. 1697)		
RefreshCache	This operation refreshes the cache for the specified file share.	Write	share* (p. 1697)		
RemoveTagsFromResource	This operation removes one or more tags from the specified resource.	Tagging	gateway (p. 1697)		
			share (p. 1697)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			tape (p. 1697)		
			volume (p. 1698)		
				aws:TagKeys (p. 1698)	
ResetCache	This operation resets all cache disks that have encountered an error and makes the disks available for reconfiguration as cache storage.	Write	gateway* (p. 1697)		
RetrieveTapeArchived	Retrieves an archived virtual tape from the virtual tape shelf (VTS) to a gateway-VTL.	Write	gateway* (p. 1697)		
			tape* (p. 1697)		
RetrieveTapeRecoveryPoint	Retrieves the recovery point for the specified virtual tape.	Write	gateway* (p. 1697)		
			tape* (p. 1697)		
SetLocalConsolePassword	Sets the password for your VM local console.	Write	gateway* (p. 1697)		
SetSMBGuestPassword	Sets the password for SMB Guest user.	Write	gateway* (p. 1697)		
ShutdownGateway	This operation shuts down a gateway.	Write	gateway* (p. 1697)		
StartGateway	This operation starts a gateway that you previously shut down.	Write	gateway* (p. 1697)		
UpdateBandwidthLimit	This operation updates the bandwidth rate limits of a gateway.	Write	gateway* (p. 1697)		
UpdateChapCredentials	This operation updates the Challenge-Handshake Authentication Protocol (CHAP) credentials for a specified iSCSI target.	Write	target* (p. 1697)		
UpdateGatewayInfo	This operation updates a gateway's metadata, which includes the gateway's name and time zone.	Write	gateway* (p. 1697)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UpdateGatewaySoftware	This operation updates the gateway's virtual machine (VM) software.	Write	gateway* (p. 1697)		
UpdateMaintenanceTime	This operation updates a gateway's weekly maintenance start time information, including day and time of the week. The maintenance time is the time in your gateway's time zone.	Write	gateway* (p. 1697)		
UpdateNFSFileShare	This operation updates a NFS file share.	Write	share* (p. 1697)		
UpdateSMBFileShare	This operation updates a SMB file share.	Write	share* (p. 1697)		
UpdateSnapshotSchedule	This operation updates a snapshot schedule configured for a gateway volume.	Write	volume* (p. 1698)		
UpdateVTLDeviceType	This operation updates the type of medium changer in a gateway-VTL.	Write	device* (p. 1697)		

Resource Types Defined by Amazon Storage Gateway

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 1690\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
device	arn:\${Partition}:storagegateway:\${Region}: \${Account}:gateway/\${GatewayId}/device/ \${Vtldevice}	
gateway	arn:\${Partition}:storagegateway:\${Region}: \${Account}:gateway/\${GatewayId}	aws:ResourceTag/ \${TagKey} (p. 1698)
share	arn:\${Partition}:storagegateway:\${Region}: \${Account}:share/\${ShareId}	aws:ResourceTag/ \${TagKey} (p. 1698)
tape	arn:\${Partition}:storagegateway:\${Region}: \${Account}:gateway/\${TapeBarcode}	aws:ResourceTag/ \${TagKey} (p. 1698)
target	arn:\${Partition}:storagegateway:\${Region}: \${Account}:gateway/\${GatewayId}/target/ \${IscsiTarget}	

Resource Types	ARN	Condition Keys
volume	arn:\${Partition}:storagegateway:\${Region}:\${Account}:gateway/\${GatewayId}/volume/\${VolumeId}	aws:ResourceTag/\${TagKey} (p. 1698)

Condition Keys for Amazon Storage Gateway

Amazon Storage Gateway defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Filters create requests based on the allowed set of values for each of the tags.	String
aws:ResourceTag/\${TagKey}	Filters actions based on tag-value associated with the resource.	String
aws:TagKeys	Filters create requests based on the presence of mandatory tags in the request.	String

Actions, Resources, and Condition Keys for Amazon Sumerian

Amazon Sumerian (service prefix: `sumerian`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).

Topics

- [Actions Defined by Amazon Sumerian \(p. 1698\)](#)
- [Resource Types Defined by Amazon Sumerian \(p. 1699\)](#)
- [Condition Keys for Amazon Sumerian \(p. 1699\)](#)

Actions Defined by Amazon Sumerian

You can specify the following actions in the Action element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the Resource element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in

a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
Login	Grant login access to the Sumerian console.	Write			
ViewRelease	Grant access to view a project release.	Read	project* (p. 1699)		

Resource Types Defined by Amazon Sumerian

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1698\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
project	<code>arn:\${Partition}:sumerian:\${Region}: \${Account}:project:\${ProjectName}</code>	

Condition Keys for Amazon Sumerian

Sumerian has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the [IAM Policy Reference](#).

Actions, Resources, and Condition Keys for AWS Support

AWS Support (service prefix: `support`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Support \(p. 1700\)](#)
- [Resource Types Defined by AWS Support \(p. 1702\)](#)
- [Condition Keys for AWS Support \(p. 1702\)](#)

Actions Defined by AWS Support

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Note

AWS Support provides the ability to access, modify and resolve cases, as well as use Trusted Advisor actions. When you use Support API to call Trusted Advisor-related actions, none of the "trustedadvisor:*" actions restrict your access. The "trustedadvisor:*" actions apply only to Trusted Advisor in the AWS Console.

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AddAttachmentsToCase	Adds one or more attachments to an AWS Support case.	Write			
AddCommunicationToCase	Adds a customer communication to an AWS Support case.	Write			
CreateCase	Creates a new AWS Support case.	Write			
DescribeAttachment	Returns the description for an attachment.	Read			
DescribeCaseAttributes	This is an internally managed attribute which allows secondary services to read AWS Support case attributes.	Read			
DescribeCases	Returns a list of AWS Support cases that matches the given inputs.	Read			
DescribeCommunications	Returns the communications and attachments for one or more AWS Support cases.	Read			
DescribeIssueTypes	Returns issue types for AWS Support cases.	Read			
DescribeServices	Returns the current list of AWS services and categories that applies to each service.	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeSeverityLevels	Returns the list of severity levels that can be assigned to an AWS Support case.	Read			
DescribeSupportLevel	Returns the support level for an AWS Account identifier.	Read			
DescribeTrustedAdvisorCheckRefreshes	Returns the status of a Trusted Advisor check based on a list of check identifiers.	Read			
DescribeTrustedAdvisorCheck	Returns the results of the Trusted Advisor check that has the specified check identifier.	Read			
DescribeTrustedAdvisorCheckSummaries	Returns the summaries of the results of the Trusted Advisor checks that have the specified check identifiers.	Read			
DescribeTrustedAdvisorChecks	Returns a list of all available Trusted Advisor checks, including name, identifier, category and description.	Read			
InitiateCallForCase	This is an internally managed function to initiate a call on AWS Support Center.	Write			
InitiateChatForCase	This is an internally managed function to initiate a chat on AWS Support Center.	Write			
PutCaseAttribute	This is an internally managed function which allows secondary services to attach attributes to AWS Support cases.	Write			
RateCaseCommunication	Rate an AWS Support case communication.	Write			
RefreshTrustedAdvisorCheck	Requests a refresh of the Trusted Advisor check that has the specified check identifier.	Write			
ResolveCase	Resolves an AWS Support case.	Write			
SearchForCases	Returns a list of AWS Support cases that matches the given inputs.	Read			

Resource Types Defined by AWS Support

AWS Support does not support specifying a resource ARN in the `Resource` element of an IAM policy statement. To allow access to AWS Support, specify “`Resource`”: “`*`” in your policy.

Condition Keys for AWS Support

Support has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for AWS Systems Manager

AWS Systems Manager (service prefix: `ssm`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Systems Manager \(p. 1702\)](#)
- [Resource Types Defined by AWS Systems Manager \(p. 1714\)](#)
- [Condition Keys for AWS Systems Manager \(p. 1716\)](#)

Actions Defined by AWS Systems Manager

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources (“`*`”) in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AddTagsToResource	Grants permission to add or overwrite one or more tags for a specified AWS resource	Tagging	document (p. 1715)		
			maintenancewindow (p. 1715)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			managed-instance (p. 1715)		
			parameter (p. 1715)		
			patchbaseline (p. 1715)		
CancelCommand	Grants permission to cancel a specified Run Command command	Write			
CancelMaintenanceWindow	Grants permission to cancel an in progress maintenance window execution	Write			
CreateActivation	Grants permission to create an activation that is used to register on-premises servers and virtual machines (VMs) with Systems Manager	Write			
CreateAssociation	Grants permission to associate a specified Systems Manager document with specified instances or other targets	Write	document* (p. 1715)		
CreateAssociationBatch	Grants permission to combine entries for multiple CreateAssociation operations in a single command	Write	document* (p. 1715)		
CreateDocument	Grants permission to create a Systems Manager SSM document	Write		aws:RequestTag/ \${TagKey} (p. 1716)	
CreateMaintenanceWindow	Grants permission to create a in progress maintenance window	Write		aws:RequestTag/ \${TagKey} (p. 1716)	
CreateOpsItem	Grants permission to create an OpsItem in OpsCenter	Write		aws:TagKeys (p. 1716)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreatePatchBaseline	Grants permission to create a patch baseline	Write		aws:RequestTag/ \${TagKey} (p. 1716) aws:TagKeys (p. 1716)	
CreateResourceDataSync	Grants permission to create a resource data sync configuration, which regularly collects inventory data from managed instances and updates the data in an Amazon S3 bucket	Write	resourcedatasync* (p. 1715)		
			ssm:SyncType (p. 1716)		
DeleteActivation	Grants permission to delete a specified activation for managed instances	Write			
DeleteAssociation	Grants permission to disassociate a specified SSM document from a specified instance	Write	document* (p. 1715)		
DeleteDocument	Grants permission to delete a specified SSM document and its instance associations	Write	document* (p. 1715)		
DeleteInventory	Grants permission to delete a specified custom inventory type, or the data associated with a custom inventory type	Write			
DeleteMaintenanceWindow	Grants permission to delete a specified maintenance window	Write	maintenancewindow* (p. 1715)		
DeleteParameter	Grants permission to delete a specified SSM parameter	Write	parameter* (p. 1715)		
DeleteParameters	Grants permission to delete multiple specified SSM parameters	Write	parameter* (p. 1715)		
DeletePatchBaseline	Grants permission to delete a specified patch baseline	Write	patchbaseline* (p. 1715)		
DeleteResourceDataSync	Grants permission to delete a specified resource data sync	Write	resourcedatasync* (p. 1715)		
			ssm:SyncType (p. 1716)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeregisterManagedInstances	Grants permission to deregister specified on-premises server or virtual machine (VM) from Systems Manager	Write	managed-instance* (p. 1715)		
DeregisterPatchBaseline	Grants permission to deregister specified patch baseline from being the default patch baseline for a specified patch group	Write	patchbaseline* (p. 1715)		
DeregisterTargetFromMaintenanceWindow	Grants permission to deregister specified target from a maintenance window	Write	maintenancewindow* (p. 1715)		
DeregisterTaskFromMaintenanceWindow	Grants permission to deregister specified task from a maintenance window	Write	maintenancewindow* (p. 1715)		
DescribeActivationDetails	Grants permission to view details about a specified managed instance activation, such as when it was created and the number of instances registered using the activation	Read			
DescribeAssociationDetails	Grants permission to view details about the specified association for a specified instance or target	Read	document* (p. 1715)		
DescribeAssociationInformationForTargets	Grants permission to view information about a specified association execution	Read			
DescribeAssociationAllExecutions	Grants permission to view all executions for a specified association	Read			
DescribeAutomationDetails	Grants permission to view details about all active and terminated Automation executions	Read			
DescribeAutomationInformationAboutStepExecutions	Grants permission to view information about all active and terminated step executions in an Automation workflow	Read			
DescribeAvailablePatches	Grants permission to view all patches eligible to include in a patch baseline	Read			
DescribeDocumentDetails	Grants permission to view details about a specified SSM document	Read	document* (p. 1715)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeDocumentInformation	Grants permission to display information about SSM document parameters in the Systems Manager console (internal Systems Manager action)	Read	document* (p. 1715)		
DescribeDocumentPermissions	Grants permission to view the permissions for a specified SSM document	Read	document* (p. 1715)		
DescribeEffectiveInstanceAssociations	Grants permission to view all current associations for a specified instance	Read			
DescribeEffectivePatchBaselineDetails	Grants permission to view details about the patches currently associated with the specified patch baseline (Windows only)	Read	patchbaseline* (p. 1715)		
DescribeInstanceAssociationStatus	Grants permission to view the status of the associations for a specified instance	Read			
DescribeInstanceDetails	Grants permission to view details about a specified instance	Read			
DescribeInstancePatchDetails	Grants permission to view status details about patches on a specified instance	Read			
DescribeInstancePatchGroupLevelPatchState	Grants permission to describe the high-level patch state for the instances in the specified patch group	Read			
DescribeInstancePatchGeneral	Grants permission to view general details about the patches on a specified instance	Read			
DescribeInstanceAmazonEC2ConsoleRenderedNodes	Grants permission to user's Amazon EC2 console to render managed instances' nodes	Read			
DescribeInventoryDeletionDetails	Grants permission to view details about a specified inventory deletion	Read			
DescribeMaintenanceWindowTaskInvocations	Grants permission to view details of a specified task invocation execution for a maintenance window	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeMaintenanceWindowExecutionDetails	Grants permission to view details about the tasks that ran during a specified maintenance window execution	List			
DescribeMaintenanceWindowUpcomingExecutions	Grants permission to view the executions of a specified maintenance window	List	maintenancewindow* (p. 1715)		
DescribeMaintenanceWindowTargets	Grants permission to view details about upcoming executions of a specified maintenance window	List			
DescribeMaintenanceWindowTasks	Grants permission to view a list of the targets associated with a specified maintenance window	List	maintenancewindow* (p. 1715)		
DescribeMaintenanceWindows	Grants permission to view a list of the tasks associated with a specified maintenance window	List	maintenancewindow* (p. 1715)		
DescribeMaintenanceWindowTargetInformation	Grants permission to view information about all or specified maintenance windows	List			
DescribeMaintenanceWindowTaskInformation	Grants permission to view information about the maintenance window targets and tasks associated with a specified instance	List			
DescribeOpsItems	Grants permission to view details about specified OpsItems	Read			
DescribeParameters	Grants permission to view details about a specified SSM parameter	List			
DescribePatchBaselines	Grants permission to view information about patch baselines that meet the specified criteria	List			
DescribePatchGroups	Grants permission to view aggregated status details for patches for a specified patch group	Read			
DescribePatchGroupInformation	Grants permission to view information about the patch baseline for a specified patch group	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribePatchProperties	Grants permission to view details of available patches for a specified operating system and patch property	List			
DescribeSessions	Grants permission to view a list of recent Session Manager sessions that meet the specified search criteria	List			
GetAutomationExecutionDetails	Grants permission to view details of a specified Automation execution	Read			
GetCommandInvocationDetails	Grants permission to view details about the command execution of a specified invocation or plugin	Read			
GetConnectionStatus	Grants permission to view the Session Manager connection status for a specified managed instance	Read			
GetDefaultPatchBaseline	Grants permission to view the current default patch baseline for a specified operating system type	Read	patchbaseline* (p. 1715)		
GetDeployablePatchBaseline	Grants permission to retrieve the current patch baseline snapshot for a specified instance	Read			
GetDocument	Grants permission to view the contents of a specified SSM document	Read	document* (p. 1715)		
GetInventory	Grants permission to view instance inventory details per the specified criteria	Read			
GetInventorySchemas	Grants permission to view a list of inventory types or attribute names for a specified inventory item type	Read			
GetMaintenanceWindowDetails	Grants permission to view details about a specified maintenance window	Read	maintenancewindow* (p. 1715)		
GetMaintenanceWindowExecutionDetails	Grants permission to view details about a specified maintenance window execution	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetMaintenanceWindowTaskDetails	Grants permission to view details about a specified maintenance window execution task	Read			
GetMaintenanceWindowTaskInvocationDetails	Grants permission to view details about a specific invocation maintenance window task running on a specific target	Read			
GetMaintenanceWindowTaskDetailsForWindow	Grants permission to view details about tasks registered with a specified maintenance window	Read	maintenancewindow*	(p. 1715)	
GetManifest	Used by Systems Manager and SSM Agent to determine package installation requirements for an instance (internal Systems Manager call)	Read			
GetOpsItem	Grants permission to view information about a specified OpsItem	Read			
GetOpsSummary	Grants permission to view summary information about OpsItems based on specified filters and aggregators	Read	resourcedatasync*	(p. 1715)	
GetParameter	Grants permission to view information about a specified parameter	Read	parameter*	(p. 1715)	
GetParameterHistory	Grants permission to view details and changes for a specified parameter	Read	parameter*	(p. 1715)	
GetParameters	Grants permission to view information about multiple specified parameters	Read	parameter*	(p. 1715)	
GetParametersByPath	Grants permission to view information about parameters in a specified hierarchy	Read	parameter*	(p. 1715)	
GetPatchBaseline	Grants permission to view information about a specified patch baseline	Read	patchbaseline*	(p. 1715)	
GetPatchBaselineForPatchGroup	Grants permission to view the ID of the current patch baseline for a specified patch group	Read	patchbaseline*	(p. 1715)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetServiceSetting	Grants permission to view the account-level setting for an AWS service	Read	servicesetting* (p. 1715)		
LabelParameterVersion	Grants permission to apply an identifying label to a specified version of a parameter	Write	parameter* (p. 1715)		
ListAssociationVersions	Grants permission to list versions of the specified association	List			
ListAssociations	Grants permission to list the associations for a specified SSM document or managed instance	List			
ListCommandInvocations	Grants permission to list information about command invocations sent to a specified instance	Read			
ListCommands	Grants permission to list the commands sent to a specified instance	Read			
ListComplianceEntries	Grants permission to list compliance status for specified resource types on a specified resource	List			
ListComplianceSummaries	Grants permission to list a summary count of compliant and noncompliant resources for a specified compliance type	List			
ListDocumentVersions	Grants permission to list all versions of a specified document	List	document* (p. 1715)		
ListDocuments	Grants permission to view information about a specified SSM document	List			
ListInstanceStateTransitions	Used by SSM Agent to check for new State Manager associations (internal Systems Manager call)	List			
ListInventoryEntries	Grants permission to view a list of specified inventory types for a specified instance	List			
ListResourceComplianceLevelSummaries	Grants permission to list resource-level summary count	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListResourceDataSync	Grants permission to list information about resource data sync configurations in an account	List		ssm:SyncType (p. 1716)	
ListTagsForResource	Grants permission to view a list of resource tags for a specified resource	Read	document (p. 1715)		
			maintenancewindow (p. 1715)		
			managed-instance (p. 1715)		
			parameter (p. 1715)		
			patchbaseline (p. 1715)		
ModifyDocument	Grants permission to share a custom SSM document publicly or privately with specified AWS accounts	Write	document* (p. 1715)		
PutComplianceItem	Grants permission to register a compliance type and other compliance details on a specified resource	Write			
PutConfigurePackagerResult	Used by SSM Agent to generate the results of specific agent requests (internal Systems Manager call)	Read			
PutInventory	Grants permission to add or update inventory items on multiple specified managed instances	Write			
PutParameter	Grants permission to create an SSM parameter	Write	parameter* (p. 1715)		
aws:RequestTag / \${TagKey} (p. 1716)	aws:TagKeys (p. 1716)				
RegisterDefaultPatchBaseline	Grants permission to specify the default patch baseline for an operating system type	Write	patchbaseline* (p. 1715)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
RegisterPatchBaseline	Grants permission to specify the default patch baseline for a specified patch group	Write	patchbaseline* (p. 1715)		
RegisterTargetWithMaintenanceWindow	Grants permission to register a target with a specified maintenance window	Write	maintenancewindow* (p. 1715)		
RegisterTaskWithMaintenanceWindow	Grants permission to register a task with a specified maintenance window	Write	maintenancewindow* (p. 1715)		
RemoveTagsFromResource	Grants permission to remove a specified tag key from a specified resource	Tagging	document (p. 1715)		
			maintenancewindow (p. 1715)		
			managed-instance (p. 1715)		
			parameter (p. 1715)		
			patchbaseline (p. 1715)		
ResetServiceSetting	Grants permission to reset the service setting for an AWS account to the default value	Write	servicesetting* (p. 1715)		
ResumeSession	Grants permission to reconnect a Session Manager session to a managed instance	Write	session* (p. 1715)		
SendAutomationSignal	Grants permission to send a signal to change the current behavior or status of a specified Automation execution	Write			
SendCommand	Grants permission to run commands on one or more specified managed instances	Write	document* (p. 1715)		
			instance (p. 1715)		
			managed-instance (p. 1715)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
				aws:ResourceTag/ \${TagKey} (p. 1716) ssm:resourceTag/ tag-key (p. 1716)	
StartAssociations	Grants permission to run a specified association manually	Write			
StartAutomation	Grants permission to initiate the execution of an Automation document	Write	document* (p. 1715)		
StartSession	Grants permission to initiate a connection to a specified target for a Session Manager session	Write	instance* (p. 1715)		
			document (p. 1715)		
				ssm:SessionDocumentAccessCheck (p. 1716)	
StopAutomation	Grants permission to stop a specified Automation execution that is already in progress	Write			
TerminateSession	Grants permission to permanently end a Session Manager connection to an instance.	Write	session* (p. 1715)		
UpdateAssociation	Grants permission to update an association and immediately run the association on the specified targets	Write			
UpdateAssociationStatus	Grants permission to update the status of the SSM document associated with a specified instance	Write	document (p. 1715)		
UpdateDocument	Grants permission to update one or more values for an SSM document	Write			
UpdateDocumentDefaultVersion	Grants permission to change the default version of an SSM document	Write			
UpdateInstanceAssociationStatus	Used by SSM Agent to update the status of the association that it is currently running (internal Systems Manager call)	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UpdateInstanceInHeartbeat	Used by SSM Agent to send a heartbeat signal to the Systems Manager service in the cloud	Write			
UpdateMaintenanceWindow	Grants permission to update a specified maintenance window	Write	maintenancewindow* (p. 1715)		
UpdateMaintenanceWindowTarget	Grants permission to update a specified maintenance window target	Write	maintenancewindow* (p. 1715)		
UpdateMaintenanceWindowTask	Grants permission to update a specified maintenance window task	Write	maintenancewindow* (p. 1715)		
UpdateManagedInstanceRole	Grants permission to assign or change the IAM role assigned to a specified managed instance	Write	managed-instance* (p. 1715)		
UpdateOpsItem	Grants permission to edit or change an OpsItem	Write			
UpdatePatchBaseline	Grants permission to update a specified patch baseline	Write	patchbaseline* (p. 1715)		
UpdateResourceDataSync	Grants permission to update a resource data sync	Write	resourcedatasync* (p. 1715)		
UpdateServiceSetting	Grants permission to update the service setting for an AWS account	Write	servicesetting* (p. 1715)		

Resource Types Defined by AWS Systems Manager

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 1702\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
association	arn:\${Partition}:ssm:\${Region}: \${Account}:association/\${AssociationId}	
automation-execution	arn:\${Partition}:ssm:\${Region}: \${Account}:automation-execution/ \${AutomationExecutionId}	

Resource Types	ARN	Condition Keys
automation-definition	arn:\${Partition}:ssm:\${Region}: \${Account}:automation-definition/ \${AutomationDefinitionName}:VersionId}	
document	arn:\${Partition}:ssm:\${Region}: \${Account}:document/\${DocumentName}	aws:ResourceTag/ \${TagKey} (p. 1716) ssm:resourceTag/tag-key (p. 1716)
instance	arn:\${Partition}:ec2:\${Region}: \${Account}:instance/\${InstanceId}	aws:ResourceTag/ \${TagKey} (p. 1716) ssm:resourceTag/tag-key (p. 1716)
maintenancewindow	arn:\${Partition}:ssm:\${Region}: \${Account}:maintenancewindow/\${ResourceId}	aws:ResourceTag/ \${TagKey} (p. 1716) ssm:resourceTag/tag-key (p. 1716)
managed-instance	arn:\${Partition}:ssm:\${Region}: \${Account}:managed-instance/ \${ManagedInstanceName}	aws:ResourceTag/ \${TagKey} (p. 1716) ssm:resourceTag/tag-key (p. 1716)
managed-instance-inventory	arn:\${Partition}:ssm:\${Region}: \${Account}:managed-instance-inventory/ \${InstanceId}	
opsitem	arn:\${Partition}:ssm:\${Region}: \${Account}:opsitem/\${ResourceId}	
parameter	arn:\${Partition}:ssm: \${Region}: \${Account}:parameter/ \${FullyQualifiedParameterName}	aws:ResourceTag/ \${TagKey} (p. 1716) ssm:resourceTag/tag-key (p. 1716)
patchbaseline	arn:\${Partition}:ssm:\${Region}: \${Account}:patchbaseline/ \${PatchBaselineIdResourceId}	aws:ResourceTag/ \${TagKey} (p. 1716) ssm:resourceTag/tag-key (p. 1716)
session	arn:\${Partition}:ssm:\${Region}: \${Account}:session/\${SessionId}	
resourcedatasync	arn:\${Partition}:ssm:\${Region}: \${Account}:resource-data-sync/\${SyncName}	
servicesetting	arn:\${Partition}:ssm:\${Region}: \${Account}:servicesetting/\${ResourceId}	

Resource Types	ARN	Condition Keys
windowtarget	arn:\${Partition}:ssm:\${Region}: \${Account}:windowtarget/\${WindowTargetId}	
windowtask	arn:\${Partition}:ssm:\${Region}: \${Account}:windowtask/\${WindowTaskId}	

Condition Keys for AWS Systems Manager

AWS Systems Manager defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Filters 'Create' requests based on the allowed set of values for a specified tags	String
aws:ResourceTag/\${TagKey}	Filters access based on a tag key-value pair assigned to the AWS resource	String
aws:TagKeys	Filters 'Create' requests based on whether mandatory tags are included in the request	String
ssm:Overwrite	Filters access by controlling whether the values for specified resources can be overwritten.	String
ssm:Recursive	Filters access for resources created in a hierarchical structure.	String
ssm:SessionDocumentDefaultSession	Filters access by verifying that a user also has access to the Default Session Manager configuration document.	Boolean
ssm:SyncType	Filters access by verifying that a user also has access to the ResourceDataSync SyncType specified in the request	String
ssm:resourceTag/tag-key	Filters access based on a tag key-value pair assigned to the Systems Manager resource	String

Actions, Resources, and Condition Keys for Amazon Textract

Amazon Textract (service prefix: `textract`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Textract \(p. 1717\)](#)
- [Resource Types Defined by Amazon Textract \(p. 1717\)](#)
- [Condition Keys for Amazon Textract \(p. 1718\)](#)

Actions Defined by Amazon Textract

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AnalyzeDocument	Detects instances of real-world document entities within an image provided as input.	Read			s3:GetObject
DetectDocumentText	Detects text in document images.	Read			s3:GetObject
GetDocumentAnalysis	Returns information about a document analysis job.	Read			
GetDocumentText	Returns information about a document text detection job.	Read			
StartDocumentAnalysis	Starts an asynchronous job to detect instances of real-world document entities within an image or pdf provided as input.	Write			s3:GetObject
StartDocumentTextDetection	Starts an asynchronous job to detect text in document images or pdfs.	Write			s3:GetObject

Resource Types Defined by Amazon Textract

Amazon Textract does not support specifying a resource ARN in the `Resource` element of an IAM policy statement. To allow access to Amazon Textract, specify "Resource": "*" in your policy.

Condition Keys for Amazon Textract

Textract has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Amazon Transcribe

Amazon Transcribe (service prefix: `transcribe`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Transcribe \(p. 1718\)](#)
- [Resource Types Defined by Amazon Transcribe \(p. 1720\)](#)
- [Condition Keys for Amazon Transcribe \(p. 1720\)](#)

Actions Defined by Amazon Transcribe

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateVocabulary	Creates a new custom vocabulary that you can use to change the way Amazon Transcribe handles transcription of an audio file.	Write			s3:GetObject
CreateVocabularyFilter	Creates a new vocabulary filter that you can use to filter out words from the transcription of an audio file generated by Amazon Transcribe	Write			s3:GetObject

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteTranscriptJob	Deletes a previously submitted transcription job along with any other generated results such as the transcription, models, and so on.	Write			
DeleteVocabulary	Deletes a vocabulary from Amazon Transcribe.	Write			
DeleteVocabularyFilter	Deletes a vocabulary filter from Amazon Transcribe.	Write			
GetTranscriptionJob	Returns information about a transcription job.	Read			
GetVocabulary	Gets information about a vocabulary.	Read			
GetVocabularyFilter	Gets information about a vocabulary filter.	Read			
ListTranscriptionJobs	Lists transcription jobs with the specified status.	List			
ListVocabularies	Returns a list of vocabularies that match the specified criteria. If no criteria are specified, returns the entire list of vocabularies.	List			
ListVocabularyFilters	Returns a list of vocabulary filters that match the specified criteria. If no criteria are specified, returns the at most 5 vocabulary filters.	List			
StartStreamTranscription	Starts a bidirectional HTTP2 stream to transcribe speech to text in real time.	Write			
StartTranscriptionJob	Starts an asynchronous job to transcribe speech to text.	Write			s3:GetObject
UpdateVocabulary	Updates an existing vocabulary with new values. The UpdateVocabulary operation overwrites all of the existing information with the values that you provide in the request.	Write			s3:GetObject

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UpdateVocabularyFilter	Updates an existing vocabulary filter with new values. The UpdateVocabularyFilter operation overwrites all of the existing information with the values that you provide in the request.	Write			s3:GetObject

Resource Types Defined by Amazon Transcribe

Amazon Transcribe does not support specifying a resource ARN in the Resource element of an IAM policy statement. To allow access to Amazon Transcribe, specify “Resource”: “*” in your policy.

Condition Keys for Amazon Transcribe

Transcribe has no service-specific context keys that can be used in the Condition element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for AWS Transfer for SFTP

AWS Transfer for SFTP (service prefix: `transfer`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).

Topics

- [Actions Defined by AWS Transfer for SFTP \(p. 1720\)](#)
- [Resource Types Defined by AWS Transfer for SFTP \(p. 1722\)](#)
- [Condition Keys for AWS Transfer for SFTP \(p. 1722\)](#)

Actions Defined by AWS Transfer for SFTP

You can specify the following actions in the Action element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources (“*”) in the Resource element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateServer	Enables the caller to create a server.	Write		aws:TagKeys (p. 1723) aws:RequestTag/\${TagKey} (p. 1723)	
CreateUser	Enables the caller to add a user associated with a server.	Write	server* (p. 1722)		iam:PassRole
				aws:TagKeys (p. 1723) aws:RequestTag/\${TagKey} (p. 1723)	
DeleteServer	Enables the caller to delete a server.	Write	server* (p. 1722)		
DeleteSshPublicKey	Enables the caller to delete an SSH public key from a user.	Write	user* (p. 1722)		
DeleteUser	Enables the caller to delete a user associated with a server.	Write	user* (p. 1722)		
DescribeServer	Enables the caller to describe a server.	Read	server* (p. 1722)		
DescribeUser	Enables the caller to describe a user associated with a server.	Read	user* (p. 1722)		
ImportSshPublicKey	Enables the caller to add an SSH public key to a user.	Write	user* (p. 1722)		
ListServers	Enables the caller to list servers	List			
ListTagsForResource	Enables the caller to list tags for a server or a user.	Read	server (p. 1722)		
			user (p. 1722)		
ListUsers	Enables the caller to list users associated with a server.	List	user* (p. 1722)		
StartServer	Enables the caller to start a server.	Write	server* (p. 1722)		
StopServer	Enables the caller to stop a server.	Write	server* (p. 1722)		
TagResource	Enables the caller to tag a server or a user.	Tagging	server (p. 1722)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			user (p. 1722)		
				aws:TagKeys (p. 1723)	
				aws:RequestTag/ \${TagKey} (p. 1723)	
TestIdentityProvider	Enables the caller to test a server's custom identity provider.	Read	server* (p. 1722)		
UntagResource	Enables the caller to untag a server or a user.	Tagging	server (p. 1722)		
			user (p. 1722)		
				aws:TagKeys (p. 1723)	
UpdateServer	Enables the caller to update the configuration of a server	Write	server* (p. 1722)		
UpdateUser	Enables the caller to update the configuration of a user	Write	server* (p. 1722)		
			user* (p. 1722)		

Resource Types Defined by AWS Transfer for SFTP

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 1720\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
user	arn:\${Partition}:transfer:\${region}: \${account}:user/\${serverId}/\${username}	aws:ResourceTag/ \${TagKey} (p. 1723)
server	arn:\${Partition}:transfer:\${region}: \${account}:server/\${serverId}	aws:ResourceTag/ \${TagKey} (p. 1723)

Condition Keys for AWS Transfer for SFTP

AWS Transfer for SFTP defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the

policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/ \${TagKey}	A key that is present in the request the user makes.	String
aws:ResourceTag/ \${TagKey}	A key that is present on the resource the user makes.	String
aws:TagKeys	The list of all the tag key names associated with the resource in the request.	String

Actions, Resources, and Condition Keys for Amazon Translate

Amazon Translate (service prefix: `translate`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon Translate \(p. 1723\)](#)
- [Resource Types Defined by Amazon Translate \(p. 1724\)](#)
- [Condition Keys for Amazon Translate \(p. 1724\)](#)

Actions Defined by Amazon Translate

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteTerminology	A synchronous action that deletes a custom terminology.	Write			
DescribeTextTranslationJob	Gets the properties associated with a synchronous batch translation job including name, ID, status, source and target languages, input/output S3 buckets, and so on.	Read			
GetTerminology	Retrieves a custom terminology.	Read			
ImportTerminology	Creates or updates a custom terminology, depending on whether or not one already exists for the given terminology name.	Write			
ListTerminologies	Provides a list of custom terminologies associated with your account.	Read			
ListTextTranslationJobs	Gets a list of the batch translation jobs that you have submitted.	Read			
StartTextTranslationJob	Starts an asynchronous batch translation job. Batch translation jobs can be used to translate large volumes of text across multiple documents at once.	Write			
StopTextTranslationJob	Stops an asynchronous batch translation job that is in progress.	Write			
TranslateText	Translate text from a source language to a target language.	Read			

Resource Types Defined by Amazon Translate

Amazon Translate does not support specifying a resource ARN in the `Resource` element of an IAM policy statement. To allow access to Amazon Translate, specify “`Resource`”: “`*`” in your policy.

Condition Keys for Amazon Translate

Translate has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for AWS Trusted Advisor

AWS Trusted Advisor (service prefix: `trustedadvisor`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Trusted Advisor \(p. 1725\)](#)
- [Resource Types Defined by AWS Trusted Advisor \(p. 1726\)](#)
- [Condition Keys for AWS Trusted Advisor \(p. 1726\)](#)

Actions Defined by AWS Trusted Advisor

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
<code>DescribeAccount</code>	View support plan and various TA preferences.	Read			
<code>DescribeAccountAdvised</code>	Resolve whether Account has Trusted Advisor	Read			
<code>DescribeCheckItems</code>	View details for the check items	Read	checks* (p. 1726)		
<code>DescribeCheckRefreshStatuses</code>	Describe check refresh statuses	Read	checks* (p. 1726)		
<code>DescribeCheckSummaries</code>	Describes the check's summaries	Read	checks* (p. 1726)		
<code>DescribeChecks</code>	List valid Trusted Advisor checks and details.	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeNotificationPreferences	Describes the notification preferences for the account	Read			
ExcludeCheckItem	Exclude recommendations for checks for a given customer	Write	checks* (p. 1726)		
IncludeCheckItem	Include recommendations for checks for a given customer	Write	checks* (p. 1726)		
RefreshCheck	Enqueue a refresh for the specified check	Write	checks* (p. 1726)		
SetAccountAccess	Toggle whether TrustedAdvisor is enabled/disabled for the account	Write			
UpdateNotificationPreferences	Update notification preferences	Write			

Resource Types Defined by AWS Trusted Advisor

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 1725\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
checks	arn:\${Partition}:trustedadvisor:\${Region}:\${Account}:checks/\${CategoryCode}/\${CheckId}	

Condition Keys for AWS Trusted Advisor

Trusted Advisor has no service-specific context keys that can be used in the Condition element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for AWS WAF

AWS WAF (service prefix: waf) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS WAF \(p. 1727\)](#)
- [Resource Types Defined by AWS WAF \(p. 1733\)](#)
- [Condition Keys for AWS WAF \(p. 1734\)](#)

Actions Defined by AWS WAF

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateByteMatchSet	Creates a ByteMatchSet.	Write	bytematchset* (p. 1734)		
CreateGeoMatchSet	Creates a GeoMatchSet, which you use to specify which web requests you want to allow or block based on the country that the requests originate from.	Write	geomatchset* (p. 1734)		
CreateIPSet	Creates an IPSet, which you use to specify which web requests you want to allow or block based on the IP addresses that the requests originate from.	Write	ipset* (p. 1734)		
CreateRateBasedRule	Creates a RateBasedRule, which contains a RateLimit specifying the maximum number of requests that AWS WAF allows from a specified IP address in a five-minute period.	Write	ratebasedrule* (p. 1734)		
				aws:RequestTag/\${TagKey} (p. 1734) aws:TagKeys (p. 1734)	
CreateRegexMatchSet	Creates a RegexMatchSet, which you use to specify which web requests you want to allow or block based on the regex patterns you specified in a RegexPatternSet.	Write	regexmatchset* (p. 1734)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateRegexPatternSet	Creates a RegexPatternSet, which you use to specify the regular expression (regex) pattern that you want AWS WAF to search for.	Write	regexpatternset* (p. 1734)		
CreateRule	Creates a Rule, which contains the IPSet objects, ByteMatchSet objects, and other predicates that identify the requests that you want to block.	Write	rule* (p. 1734)		
				aws:RequestTag/\${TagKey} (p. 1734)	aws:TagKeys (p. 1734)
CreateRuleGroup	Creates a RuleGroup. A rule group is a collection of predefined rules that you add to a WebACL.	Write	rulegroup* (p. 1734)		
				aws:RequestTag/\${TagKey} (p. 1734)	aws:TagKeys (p. 1734)
CreateSizeConstraintSet	Creates a SizeConstraintSet, which you use to identify the part of a web request that you want to check for length.	Write	sizeconstraintset* (p. 1734)		
CreateSqlInjectionMatchSet	Creates a SqlInjectionMatchSet, which you use to allow, block, or count requests that contain snippets of SQL code in a specified part of web requests.	Write	sqlinjectionmatchset* (p. 1734)		
CreateWebACL	Creates a WebACL, which contains the Rules that identify the CloudFront web requests that you want to allow, block, or count.	Permissions management	webacl* (p. 1734)		
				aws:RequestTag/\${TagKey} (p. 1734)	aws:TagKeys (p. 1734)
CreateXssMatchSet	Creates an XssMatchSet, which you use to allow, block, or count requests that contain cross-site scripting attacks in the specified part of web requests.	Write	xssmatchset* (p. 1734)		
DeleteByteMatchSet	Permanently deletes a ByteMatchSet.	Write	bytematchset* (p. 1734)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteGeoMatchSet	Permanently deletes an GeoMatchSet.	Write	geomatchset* (p. 1734)		
DeleteIPSet	Permanently deletes an IPSet.	Write	ipset* (p. 1734)		
DeleteLoggingConfiguration	Permanently deletes the LoggingConfiguration from the specified web ACL.	Write	webacl* (p. 1734)		
DeletePermission	Permanently deletes an IAM Policy from the specified RuleGroup.	Permissions management	rulegroup* (p. 1734)		
DeleteRateBasedRule	Permanently deletes a RateBasedRule.	Write	ratebasedrule* (p. 1734)		
DeleteRegexMatchSet	Permanently deletes an RegexMatchSet.	Write	regexmatchset* (p. 1734)		
DeleteRegexPatternSet	Permanently deletes an RegexPatternSet.	Write	regexpatternset* (p. 1734)		
DeleteRule	Permanently deletes a Rule.	Write	rule* (p. 1734)		
DeleteRuleGroup	Permanently deletes a RuleGroup.	Write	rulegroup* (p. 1734)		
DeleteSizeConstraintSet	Permanently deletes a SizeConstraintSet.	Write	sizeconstraintset* (p. 1734)		
DeleteSqlInjectionMatchSet	Permanently deletes a SqlInjectionMatchSet.	Write	sqlinjectionmatchset* (p. 1734)		
DeleteWebACL	Permanently deletes a WebACL.	Permissions management	webacl* (p. 1734)		
DeleteXssMatchSet	Permanently deletes an XssMatchSet.	Write	xssmatchset* (p. 1734)		
GetByteMatchSet	Returns the ByteMatchSet specified by ByteMatchSetId.	Read	bytematchset* (p. 1734)		
GetChangeToken	When you want to create, update, or delete AWS WAF objects, get a change token and include the change token in the create, update, or delete request.	Read			
GetChangeToken	Returns the status of a ChangeToken that you got by calling GetChangeToken.	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetGeoMatchSet	Returns the GeoMatchSet specified by GeoMatchSetId.	Read	geomatchset* (p. 1734)		
GetIPSet	Returns the IPSet that is specified by IPSetId.	Read	ipset* (p. 1734)		
GetLoggingConfiguration	Returns the LoggingConfiguration for the specified web ACL.	Read	webacl* (p. 1734)		
GetPermissionPolicy	Returns the IAM policy attached to the RuleGroup.	Read	rulegroup* (p. 1734)		
GetRateBasedRule	Returns the RateBasedRule that is specified by the RuleId that you included in the GetRateBasedRule request.	Read	ratebasedrule* (p. 1734)		
GetRateBasedRulesCurrentlyBeingBlocked	Returns an array of IP addresses currently being blocked by the RateBasedRule that is specified by the RuleId.	Read	ratebasedrule* (p. 1734)		
GetRegexMatchSet	Returns the RegexMatchSet specified by RegexMatchSetId.	Read	regexmatchset* (p. 1734)		
GetRegexPatternSet	Returns the RegexPatternSet specified by RegexPatternSetId.	Read	regexpatternset* (p. 1734)		
GetRule	Returns the Rule that is specified by the RuleId that you included in the GetRule request.	Read	rule* (p. 1734)		
GetRuleGroup	Returns the RuleGroup that is specified by the RuleGroupId that you included in the GetRuleGroup request.	Read	rulegroup* (p. 1734)		
GetSampledRequests	Gets detailed information about a specified number of requests--a sample--that AWS WAF randomly selects from among the first 5,000 requests that your AWS resource received during a time range that you choose.	Read	rule (p. 1734)		
			webacl (p. 1734)		
GetSizeConstraintSet	Returns the SizeConstraintSet specified by SizeConstraintSetId.	Read	sizeconstraintset* (p. 1734)		
GetSqlInjectionMatchSet	Returns the SqlInjectionMatchSet that is specified by SqlInjectionMatchSetId.	Read	sqlinjectionmatchset* (p. 1734)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetWebACL	Returns the WebACL that is specified by WebACLId.	Read	webacl* (p. 1734)		
GetXssMatchSet	Returns the XssMatchSet that is specified by XssMatchSetId.	Read	xssmatchset* (p. 1734)		
ListActivatedRules	Returns an array of ActivatedRule objects.	List			
ListByteMatchSets	Returns an array of ByteMatchSetSummary objects.	List			
ListGeoMatchSets	Returns an array of GeoMatchSetSummary objects.	List			
ListIPSets	Returns an array of IPSetSummary objects in the response.	List			
ListLoggingConfigurations	Returns an array of LoggingConfiguration objects.	List			
ListRateBasedRules	Returns an array of RuleSummary objects.	List			
ListRegexMatchSets	Returns an array of RegexMatchSetSummary objects.	List			
ListRegexPatternSets	Returns an array of RegexPatternSetSummary objects.	List			
ListRuleGroups	Returns an array of RuleGroup objects.	List			
ListRules	Returns an array of RuleSummary objects.	List			
ListSizeConstraints	Returns an array of SizeConstraintSetSummary objects.	List			
ListSqlInjectionMatchSets	Returns an array of SqlInjectionMatchSet objects.	List			
ListSubscribedRuleGroups	Returns an array of RuleGroup objects that you are subscribed to.	List			
ListTagsForResource	Lists the Tags for a given resource.	Read	ratebasedrule (p. 1734) rule (p. 1734)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
			rulegroup (p. 1734)		
			webacl (p. 1734)		
ListWebACLS	Returns an array of WebACLSummary objects in the response.	List			
ListXssMatchSets	Returns an array of XSSMatchSet objects.	List			
PutLoggingConfig	Associates a LoggingConfiguration with a specified web ACL.	Write	webacl* (p. 1734)		iam:CreateServiceLinkedRole
PutPermissionPolicy	Attaches a IAM policy to the specified resource. The only supported use for this action is to share a RuleGroup across accounts.	Permissions management	rulegroup* (p. 1734)		
TagResource	Adds a Tag to a given resource.	Tagging	ratebasedrule (p. 1734)		
	rule (p. 1734)				
	rulegroup (p. 1734)				
	webacl (p. 1734)				
			aws:RequestTag/\${TagKey} (p. 1734)		
UntagResource	Removes a Tag from a given resource.	Tagging	ratebasedrule (p. 1734)		
	rule (p. 1734)				
	rulegroup (p. 1734)				
	webacl (p. 1734)				
			aws:TagKeys (p. 1734)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UpdateByteMatch	Inserts or deletes ByteMatchTuple objects (filters) in a ByteMatchSet.	Write	bytematchset* (p. 1734)		
UpdateGeoMatch	Inserts or deletes GeoMatchConstraint objects in a GeoMatchSet.	Write	geomatchset* (p. 1734)		
UpdateIPSet	Inserts or deletes IPSetDescriptor objects in an IPSet.	Write	ipset* (p. 1734)		
UpdateRateBasedObject	Inserts or deletes Predicate objects in a rule and updates the RateLimit in the rule.	Write	ratebasedrule* (p. 1734)		
UpdateRegexMatch	Inserts or deletes RegexMatchTuple objects (filters) in a RegexMatchSet.	Write	regexmatchset* (p. 1734)		
UpdateRegexPattern	Inserts or deletes RegexPatternStrings in a RegexPatternSet.	Write	regexpatternset* (p. 1734)		
UpdateRule	Inserts or deletes Predicate objects in a Rule.	Write	rule* (p. 1734)		
UpdateRuleGroup	Inserts or deletes ActivatedRule objects in a RuleGroup.	Write	rulegroup* (p. 1734)		
UpdateSizeConstraint	Inserts or deletes SizeConstraint objects (filters) in a SizeConstraintSet.	Write	sizeconstraintset* (p. 1734)		
UpdateSqlInjection	Inserts or deletes SqlInjectionMatchTuple objects (filters) in a SqlInjectionMatchSet.	Write	sqlinjectionmatchset* (p. 1734)		
UpdateWebACL	Inserts or deletes ActivatedRule objects in a WebACL.	Permissions management	webacl* (p. 1734)		
UpdateXssMatch	Inserts or deletes XSSMatchTuple objects (filters) in an XSSMatchSet.	Write	xssmatchset* (p. 1734)		

Resource Types Defined by AWS WAF

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 1727\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
bytematchset	arn:\${Partition}:waf::: \${Account}:bytematchset/\${Id}	
ipset	arn:\${Partition}:waf:::\${Account}:ipset/\${Id}	
ratebasedrule	arn:\${Partition}:waf::: \${Account}:ratebasedrule/\${Id}	aws:ResourceTag/ \${TagKey} (p. 1734)
rule	arn:\${Partition}:waf:::\${Account}:rule/\${Id}	aws:ResourceTag/ \${TagKey} (p. 1734)
sizeconstraintset	arn:\${Partition}:waf::: \${Account}:sizeconstraintset/\${Id}	
sqlinjectionmatchset	arn:\${Partition}:waf::: \${Account}:sqlinjectionmatchset/\${Id}	
webacl	arn:\${Partition}:waf:::\${Account}:webacl/ \${Id}	aws:ResourceTag/ \${TagKey} (p. 1734)
xssmatchset	arn:\${Partition}:waf::: \${Account}:xssmatchset/\${Id}	
regexmatchset	arn:\${Partition}:waf:::\${Account}:regexmatch/ \${Id}	
regexpatternset	arn:\${Partition}:waf::: \${Account}:regexpatternset/\${Id}	
geomatchset	arn:\${Partition}:waf::: \${Account}:geomatchset/\${Id}	
rulegroup	arn:\${Partition}:waf:::\${Account}:rulegroup/ \${Id}	aws:ResourceTag/ \${TagKey} (p. 1734)

Condition Keys for AWS WAF

AWS WAF defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/ \${TagKey}	Filters actions based on the allowed set of values for each of the tags	String
aws:ResourceTag/ \${TagKey}	Filters actions based on tag-value associated with the resource	String
aws:TagKeys	Filters actions based on the presence of mandatory tags in the request	String

Actions, Resources, and Condition Keys for AWS WAF Regional

AWS WAF Regional (service prefix: `waf-regional`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS WAF Regional \(p. 1735\)](#)
- [Resource Types Defined by AWS WAF Regional \(p. 1742\)](#)
- [Condition Keys for AWS WAF Regional \(p. 1743\)](#)

Actions Defined by AWS WAF Regional

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AssociateWebACL	Associates a WebACL with a <code>resource</code> .	Write	<code>loadbalancer/app/*</code> (p. 1742)		
			<code>webacl*</code> (p. 1743)		
CreateByteMatchSet	Creates a ByteMatchSet.	Write	<code>bytematchset*</code> (p. 1742)		
CreateGeoMatchSet	Creates a GeoMatchSet, which you use to specify which web requests you want to allow or block based on the country that the requests originate from.	Write	<code>geomatchset*</code> (p. 1743)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateIPSet	Creates an IPSet, which you use to specify which web requests you want to allow or block based on the IP addresses that the requests originate from.	Write	ipset* (p. 1742)		
CreateRateBasedRule	Creates a RateBasedRule, which contains a RateLimit specifying the maximum number of requests that AWS WAF allows from a specified IP address in a five-minute period.	Write	ratebasedrule* (p. 1742)		
				aws:RequestTag/\${TagKey} (p. 1743)	
CreateRegexMatchSet	Creates a RegexMatchSet, which you use to specify which web requests you want to allow or block based on the regex patterns you specified in a RegexPatternSet.	Write	regexmatchset* (p. 1743)		
CreateRegexPatternSet	Creates a RegexPatternSet, which you use to specify the regular expression (regex) pattern that you want AWS WAF to search for.	Write	regexpatternset* (p. 1743)		
CreateRule	Creates a Rule, which contains the IPSet objects, ByteMatchSet objects, and other predicates that identify the requests that you want to lock.	Write	rule* (p. 1743)		
				aws:RequestTag/\${TagKey} (p. 1743)	
CreateRuleGroup	Creates a RuleGroup. A rule group is a collection of predefined rules that you add to a WebACL.	Write	rulegroup* (p. 1743)		
				aws:RequestTag/\${TagKey} (p. 1743)	
CreateSizeConstraintSet	Creates a SizeConstraintSet, which you use to identify the part of a web request that you want to check for length.	Write	sizeconstraintset* (p. 1743)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateSqlInjectionMatchSet	Creates a SqlInjectionMatchSet, which you use to allow, block, or count requests that contain snippets of SQL code in a specified part of web requests.	Write	sqlinjectionmatchset* (p. 1743)		
CreateWebACL	Creates a WebACL, which contains the Rules that identify the CloudFront web requests that you want to allow, block, or count.	Permissions management	webacl* (p. 1743)		
			aws:RequestTag/\${TagKey} (p. 1743)	aws:TagKeys (p. 1743)	
CreateXssMatchSet	Creates an XssMatchSet, which you use to allow, block, or count requests that contain cross-site scripting attacks in the specified part of web requests.	Write	xssmatchset* (p. 1743)		
DeleteByteMatchSet	Permanently deletes a ByteMatchSet.	Write	bytematchset* (p. 1742)		
DeleteGeoMatchSet	Permanently deletes an GeoMatchSet.	Write	geomatchset* (p. 1743)		
DeleteIPSet	Permanently deletes an IPSet.	Write	ipset* (p. 1742)		
DeleteLoggingConfiguration	Permanently deletes the LoggingConfiguration from the specified web ACL.	Write	webacl* (p. 1743)		
DeletePermission	Permanently deletes an IAM Policy from the specified RuleGroup.	Permissions management	rulegroup* (p. 1743)		
DeleteRateBasedRule	Permanently deletes a RateBasedRule.		ratebasedrule* (p. 1742)		
DeleteRegexMatchSet	Permanently deletes an RegexMatchSet.	Write	regexmatchset* (p. 1743)		
DeleteRegexPatternSet	Permanently deletes an RegexPatternSet.	Write	regexpatternset* (p. 1743)		
DeleteRule	Permanently deletes a Rule.	Write	rule* (p. 1743)		
DeleteRuleGroup	Permanently deletes a RuleGroup.	Write	rulegroup* (p. 1743)		
DeleteSizeConstraintSet	Permanently deletes a SizeConstraintSet.	Write	sizeconstraintset* (p. 1743)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteSqlInjectionMatchSet	Permanently deletes a SqlInjectionMatchSet.	Write	sqlinjectionmatchset* (p. 1743)		
DeleteWebACL	Permanently deletes a WebACL.	Permissions management	webacl* (p. 1743)		
DeleteXssMatchSet	Permanently deletes an XSSMatchSet.	Write	xssmatchset* (p. 1743)		
DisassociateWebACL	Removes a WebACL from the specified resource.	Write	loadbalancer/app/* (p. 1742)		
GetByteMatchSet	Returns the ByteMatchSet specified by ByteMatchSetId.	Read	bytematchset* (p. 1742)		
GetChangeToken	When you want to create, update, or delete AWS WAF objects, get a change token and include the change token in the create, update, or delete request.	Read			
GetChangeToken	Returns the status of a ChangeToken that you got by calling GetChangeToken.	Read			
GetGeoMatchSet	Returns the GeoMatchSet specified by GeoMatchSetId.	Read	geomatchset* (p. 1743)		
GetIPSet	Returns the IPSet that is specified by IPSetId.	Read	ipset* (p. 1742)		
GetLoggingConfiguration	Returns the LoggingConfiguration for the specified web ACL.	Read	webacl* (p. 1743)		
GetPermissionPolicy	Returns the IAM policy attached to the RuleGroup.	Read	rulegroup* (p. 1743)		
GetRateBasedRule	Returns the RateBasedRule that is specified by the RuleId that you included in the GetRateBasedRule request.	Read	ratebasedrule* (p. 1742)		
GetRateBasedRulesCurrentlyBeingBlocked	Returns an array of IP addresses currently being blocked by the RateBasedRule that is specified by the RuleId.	Read	ratebasedrule* (p. 1742)		
GetRegexMatchSet	Returns the RegexMatchSet specified by RegexMatchSetId.	Read	regexmatchset* (p. 1743)		
GetRegexPatternSet	Returns the RegexPatternSet specified by RegexPatternSetId.	Read	regexpatternset* (p. 1743)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetRule	Returns the Rule that is specified by the RuleId that you included in the GetRule request.	Read	rule* (p. 1743)		
GetRuleGroup	Returns the RuleGroup that is specified by the RuleGroupId that you included in the GetRuleGroup request.	Read	rulegroup* (p. 1743)		
GetSampledRequests	Gets detailed information about a specified number of requests--a sample--that AWS WAF randomly selects from among the first 5,000 requests that your AWS resource received during a time range that you choose.	Read	rule (p. 1743)		
			webacl (p. 1743)		
GetSizeConstraintSet	Returns the SizeConstraintSet specified by SizeConstraintSetId.	Read	sizeconstraintset* (p. 1743)		
GetSqlInjectionMatchSet	Returns the SqlInjectionMatchSet that is specified by SqlInjectionMatchSetId.	Read	sqlinjectionmatchset* (p. 1743)		
GetWebACL	Returns the WebACL that is specified by WebACLIId.	Read	webacl* (p. 1743)		
GetWebACLForResource	Returns the WebACL for the specified resource.	Read	loadbalancer/app/* (p. 1742)		
GetXssMatchSet	Returns the XssMatchSet that is specified by XssMatchSetId.	Read	xssmatchset* (p. 1743)		
ListActivatedRules	Returns an array of ActivatedRule objects.	List			
ListByteMatchSets	Returns an array of ByteMatchSetSummary objects.	List			
ListGeoMatchSets	Returns an array of GeoMatchSetSummary objects.	List			
ListIPSets	Returns an array of IPSetSummary objects in the response.	List			
ListLoggingConfigurations	Returns an array of LoggingConfiguration objects.	List			
ListRateBasedRules	Returns an array of RuleSummary objects.	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListRegexMatchSets	Returns an array of RegexMatchSetSummary objects.	List			
ListRegexPatterns	Returns an array of RegexPatternSetSummary objects.	List			
ListResourcesForWebACL	Returns an array of resources associated with the specified WebACL.	List	webacl* (p. 1743)		
ListRuleGroups	Returns an array of RuleGroup objects.	List			
ListRules	Returns an array of RuleSummary objects.	List			
ListSizeConstraints	Returns an array of SizeConstraintSetSummary objects.	List			
ListSqlInjectionMatchSets	Returns an array of SqlInjectionMatchSet objects.	List			
ListSubscribedRuleGroups	Returns an array of RuleGroup objects that you are subscribed to.	List			
ListTagsForResource	Lists the Tags for a given resource.	Read	ratebasedrule (p. 1742)		
			rule (p. 1743)		
			rulegroup (p. 1743)		
			webacl (p. 1743)		
ListWebACLs	Returns an array of WebACLSummary objects in the response.	List			
ListXssMatchSets	Returns an array of XSSMatchSet objects.	List			
PutLoggingConfiguration	Associates a Logging Configuration with a specified web ACL.	Write	webacl* (p. 1743)		iam:CreateServiceLinkedRole

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
PutPermissionPolicy	Attaches a IAM policy to the specified resource. The only supported use for this action is to share a RuleGroup across accounts.	Permissions management	rulegroup* (p. 1743)		
TagResource	Adds a Tag to a given resource.	Tagging	ratebasedrule (p. 1742) rule (p. 1743) rulegroup (p. 1743) webacl (p. 1743)	aws:RequestTag / \${TagKey} (p. 1743) aws:TagKeys (p. 1743)	
UntagResource	Removes a Tag from a given resource.	Tagging	ratebasedrule (p. 1742) rule (p. 1743) rulegroup (p. 1743) webacl (p. 1743)	aws:TagKeys (p. 1743)	
UpdateByteMatchSet	Inserts or deletes ByteMatchTuple objects (filters) in a ByteMatchSet.	Write	bytematchset* (p. 1742)		
UpdateGeoMatchSet	Inserts or deletes GeoMatchConstraint objects in a GeoMatchSet.	Write	geomatchset* (p. 1743)		
UpdateIPSet	Inserts or deletes IPSetDescriptor objects in an IPSet.	Write	ipset* (p. 1742)		
UpdateRateBasedRule	Inserts or deletes Predicate objects in a rule and updates the RateLimit in the rule.	Write	ratebasedrule* (p. 1742)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UpdateRegexMatchSet	Inserts or deletes RegexMatchTuple objects (filters) in a RegexMatchSet.	Write	regexmatchset* (p. 1743)		
UpdateRegexPatternSet	Inserts or deletes RegexPatternStrings in a RegexPatternSet.	Write	regexpatternset* (p. 1743)		
UpdateRule	Inserts or deletes Predicate objects in a Rule.	Write	rule* (p. 1743)		
UpdateRuleGroup	Inserts or deletes ActivatedRule objects in a RuleGroup.	Write	rulegroup* (p. 1743)		
UpdateSizeConstraintSet	Inserts or deletes SizeConstraint objects (filters) in a SizeConstraintSet.	Write	sizeconstraintset* (p. 1743)		
UpdateSqlInjectionMatchSet	Inserts or deletes SqlInjectionMatchTuple objects (filters) in a SqlInjectionMatchSet.	Write	sqlinjectionmatchset* (p. 1743)		
UpdateWebACL	Inserts or deletes ActivatedRule objects in a WebACL.	Permissions management	webacl* (p. 1743)		
UpdateXssMatchSet	Inserts or deletes XssMatchTuple objects (filters) in an XssMatchSet.	Write	xssmatchset* (p. 1743)		

Resource Types Defined by AWS WAF Regional

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1735\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
bytematchset	arn:\${Partition}:waf-regional:\${Region}: \${Account}:bytematchset/\${Id}	
ipset	arn:\${Partition}:waf-regional:\${Region}: \${Account}:ipset/\${Id}	
loadbalancer/app/	arn:\${Partition}:elasticloadbalancing: \${Region}: \${Account}:loadbalancer/app/ \${LoadBalancerName}/\${LoadBalancerId}	
ratebasedrule	arn:\${Partition}:waf-regional:\${Region}: \${Account}:ratebasedrule/\${Id}	aws:ResourceTag/\${TagKey} (p. 1743)

Resource Types	ARN	Condition Keys
rule	arn:\${Partition}:waf-regional:\${Region}: \${Account}:rule/\${Id}	aws:ResourceTag/\${TagKey} (p. 1743)
sizeconstraintset	arn:\${Partition}:waf-regional:\${Region}: \${Account}:sizeconstraintset/\${Id}	
sqlinjectionmatchset	arn:\${Partition}:waf-regional:\${Region}: \${Account}:sqlinjectionmatchset/\${Id}	
webacl	arn:\${Partition}:waf-regional:\${Region}: \${Account}:webacl/\${Id}	aws:ResourceTag/\${TagKey} (p. 1743)
xssmatchset	arn:\${Partition}:waf-regional:\${Region}: \${Account}:xssmatchset/\${Id}	
regexmatchset	arn:\${Partition}:waf-regional:\${Region}: \${Account}:regexmatch/\${Id}	
regexpatternset	arn:\${Partition}:waf-regional:\${Region}: \${Account}:regexpatternset/\${Id}	
geomatchset	arn:\${Partition}:waf-regional:\${Region}: \${Account}:geomatchset/\${Id}	
rulegroup	arn:\${Partition}:waf-regional:\${Region}: \${Account}:rulegroup/\${Id}	aws:ResourceTag/\${TagKey} (p. 1743)

Condition Keys for AWS WAF Regional

AWS WAF Regional defines the following condition keys that can be used in the `Condition` element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Filters actions based on the allowed set of values for each of the tags	String
aws:ResourceTag/\${TagKey}	Filters actions based on tag-value associated with the resource	String
aws:TagKeys	Filters actions based on the presence of mandatory tags in the request	String

Actions, Resources, and Condition Keys for AWS WAF V2

AWS WAF V2 (service prefix: `wafv2`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS WAF V2 \(p. 1744\)](#)
- [Resource Types Defined by AWS WAF V2 \(p. 1749\)](#)
- [Condition Keys for AWS WAF V2 \(p. 1749\)](#)

Actions Defined by AWS WAF V2

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AssociateWebACL	Grants permission to associate a WebACL with a resource.	Write	webacl* (p. 1749)		
			apigateway (p. 1749)		
			loadbalancer/app/ (p. 1749)		
CheckCapacity	Grants permission to calculate web ACL capacity unit (WCU) requirements for a specified scope and set of rules.	Read			
CreateIPSet	Grants permission to create an IPSet.	Write	ipset* (p. 1749)		
				aws:RequestTag/\${TagKey} (p. 1750)	
				aws:TagKeys (p. 1750)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateRegexPatternSet	Grants permission to create a RegexPatternSet.	Write	regexpatternset* (p. 1749)		
			aws:RequestTag/ \${TagKey} (p. 1750)		
CreateRuleGroup	Grants permission to create a RuleGroup.	Write	rulegroup* (p. 1749)		
			aws:RequestTag/ \${TagKey} (p. 1750)		
CreateWebACL	Grants permission to create a WebACL.	Permissions management	webacl* (p. 1749)		
			aws:RequestTag/ \${TagKey} (p. 1750)		
DeleteIPSet	Grants permission to delete the specified IPSet.	Write	ipset* (p. 1749)		
DeleteLoggingConfiguration	Grants permission to delete the LoggingConfiguration from the specified WebACL.	Write	webacl* (p. 1749)		
DeleteRegexPatternSet	Grants permission to delete the specified RegexPatternSet.	Write	regexpatternset* (p. 1749)		
DeleteRuleGroup	Grants permission to delete the specified RuleGroup.	Write	rulegroup* (p. 1749)		
DeleteWebACL	Grants permission to delete the specified WebACL.	Permissions management	webacl* (p. 1749)		
DescribeManagedRuleGroup	Grants permission to view high-level information for a managed rule group.				
DisassociateWebACL	Grants permission disassociate a WebACL from an application resource.	Write	apigateway (p. 1749)		
	loadbalancer/app/ (p. 1749)				

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetIPSet	Grants permission to view details about the specified IPSet.	Read	ipset* (p. 1749)		
				aws:ResourceTag/ {\$TagKey} (p. 1750)	
GetLoggingConfig	Grants permission to view Logging Configuration about the specified WebACL.	Read	webacl* (p. 1749)		
				aws:ResourceTag/ {\$TagKey} (p. 1750)	
GetRateBasedStatistics	Grants permission to view keys that are currently blocked by a rate-based rule.	Read	webacl* (p. 1749)		
				aws:ResourceTag/ {\$TagKey} (p. 1750)	
GetRegexPatternSet	Grants permission to view details about the specified RegexPatternSet.	Read	regexpatternset* (p. 1749)		
				aws:ResourceTag/ {\$TagKey} (p. 1750)	
GetRuleGroup	Grants permission to view details about the specified RuleGroup.	Read	rulegroup* (p. 1749)		
				aws:ResourceTag/ {\$TagKey} (p. 1750)	
GetSampledRequests	Grants permission to view detailed information about a specified number of requests--a sample--that AWS WAF randomly selects from among the first 5,000 requests that your AWS resource received during a time range that you choose.	Read	webacl* (p. 1749)		
GetWebACL	Grants permission to view details about the specified GetWebACL.	Read	webacl* (p. 1749)		
				aws:ResourceTag/ {\$TagKey} (p. 1750)	
GetWebACLForResource	Grants permission to view the WebACL for the specified resource.	Read	apigateway (p. 1749)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions	
			loadbalancer/app/ (p. 1749)			
ListAvailableManagerRoleGroups	Grants permission to view an array of rule groups that are available for you to use.	List				
ListIPSets	Grants permission to view an array of IPSetSummary objects for the IP sets that you manage.	List				
ListLoggingConfigurations	Grants permission to view an array of LoggingConfiguration objects.	List				
ListRegexPatternSets	Grants permission to view an array of RegexPatternSetSummary objects for the regex pattern sets that you manage.	List				
ListResourcesForWebACL	Grants permission to view an array of the Amazon Resource Names (ARNs) for the resources that are associated with the specified web ACL.	List	webacl* (p. 1749)			
ListRuleGroups	Grants permission to view an array of RuleGroupSummary objects for the rule groups that you manage.	List				
ListTagsForResource	Grants permission to lists tag for the specified resource.	Read	ipset (p. 1749)			
			regexpatternset (p. 1749)			
			rulegroup (p. 1749)			
			webacl (p. 1749)			
			aws:ResourceTag/\$ {TagKey} (p. 1750)			
ListWebACLS	Grants permission to view an array of WebACLSummary objects for the web ACLs that you manage.	List				

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
PutLoggingConfig	Grants permission to enables the specified LoggingConfiguration, to start logging from a web ACL.	Write	webacl* (p. 1749)		iam:CreateServiceLinkedRole
TagResource	Grants permission to associates tags with the specified AWS resource.	Tagging	ipset (p. 1749)		
			regexpatternset (p. 1749)		
			rulegroup (p. 1749)		
			webacl (p. 1749)		
				aws:TagKeys (p. 1750)	
UntagResource	Grants permission to disassociates tags from an AWS resource.	Tagging	ipset (p. 1749)		
			regexpatternset (p. 1749)		
			rulegroup (p. 1749)		
			webacl (p. 1749)		
				aws:TagKeys (p. 1750)	
UpdateIPSet	Grants permission to update the specified IPSet.	Write	ipset* (p. 1749)		
				aws:ResourceTag /\${TagKey} (p. 1750)	
UpdateRegexPatternSet	Grants permission to update the specified RegexPatternSet.	Write	regexpatternset* (p. 1749)		
				aws:ResourceTag /\${TagKey} (p. 1750)	

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UpdateRuleGroup	Grants permission to update the specified RuleGroup.	Write	rulegroup* (p. 1749)		
				aws:ResourceTag/ {\$TagKey} (p. 1750)	
UpdateWebACL	Grants permission to update the specified WebACL.	Permissions management (p. 1749)	webacl*		
				aws:ResourceTag/ {\$TagKey} (p. 1750)	

Resource Types Defined by AWS WAF V2

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 1744\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
webacl	arn:\${Partition}:wafv2:\${Region}:\${Account}: \${Scope}/webacl/\${Name}/\${Id}	aws:ResourceTag/ {\$TagKey} (p. 1750)
ipset	arn:\${Partition}:wafv2:\${Region}:\${Account}: \${Scope}/ipset/\${Name}/\${Id}	aws:ResourceTag/ {\$TagKey} (p. 1750)
rulegroup	arn:\${Partition}:wafv2:\${Region}:\${Account}: \${Scope}/rulegroup/\${Name}/\${Id}	aws:ResourceTag/ {\$TagKey} (p. 1750)
regexpatternset	arn:\${Partition}:wafv2:\${Region}:\${Account}: \${Scope}/regexpatternset/\${Name}/\${Id}	aws:ResourceTag/ {\$TagKey} (p. 1750)
loadbalancer/app/	arn:\${Partition}:elasticloadbalancing: \${Region}: \${Account}:loadbalancer/app/ \${LoadBalancerName}/\${LoadBalancerId}	
apigateway	arn:\${Partition}:apigateway:\${Region}:::/ restapis/\${ApiId}/stages/prod	

Condition Keys for AWS WAF V2

AWS WAF V2 defines the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For details about the columns in the following table, see [The Condition Keys Table \(p. 687\)](#).

To view the global condition keys that are available to all services, see [Available Global Condition Keys](#) in the *IAM Policy Reference*.

Condition Keys	Description	Type
aws:RequestTag/\${TagKey}	Filters actions based on the allowed set of values for each of the tags	String
aws:ResourceTag/\${TagKey}	Filters actions based on tag-value associated with the resource	String
aws:TagKeys	Filters actions based on the presence of mandatory tags in the request	String

Actions, Resources, and Condition Keys for AWS Well-Architected Tool

AWS Well-Architected Tool (service prefix: wellarchitected) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS Well-Architected Tool \(p. 1750\)](#)
- [Resource Types Defined by AWS Well-Architected Tool \(p. 1751\)](#)
- [Condition Keys for AWS Well-Architected Tool \(p. 1751\)](#)

Actions Defined by AWS Well-Architected Tool

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateWorkload	Creates a new workload.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateWorkloadShare	Shares a workload with another account.	Write	workload* (p. 1751)		
DeleteWorkload	Deletes an existing workload.	Write	workload* (p. 1751)		
GetWorkload	Retrieves the specified workload.	Read	workload* (p. 1751)		
ListWorkloads	Lists the workloads in this account.	List			

Resource Types Defined by AWS Well-Architected Tool

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 1750\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
workload	arn:\${Partition}:wellarchitected:\${Region}:\${Account}:workload/\${ResourceId}	

Condition Keys for AWS Well-Architected Tool

Well-Architected Tool has no service-specific context keys that can be used in the Condition element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Amazon WorkDocs

Amazon WorkDocs (service prefix: `workdocs`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon WorkDocs \(p. 1752\)](#)
- [Resource Types Defined by Amazon WorkDocs \(p. 1756\)](#)
- [Condition Keys for Amazon WorkDocs \(p. 1756\)](#)

Actions Defined by Amazon WorkDocs

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AbortDocumentVersionUpload	Grants permission to abort the upload of the specified document version that was previously initiated by <code>InitiateDocumentVersionUpload</code> .	Write			
ActivateUser	Grants permission to activate the specified user. Only active users can access Amazon WorkDocs.	Write			
AddResourcePermissions	Grants permission to create a set of permissions for the specified folder or document.	Write			
AddUserToGroup [permission only]	Grants permission to add a user to a group.	Write			
CheckAlias [permission only]	Grants permission to check an alias.	Read			
CreateComment	Grants permission to add a new comment to the specified document version.	Write			
CreateCustomMetadata	Grants permission to add one or more custom properties to the specified resource.	Write			
CreateFolder	Grants permission to create a folder with the specified name and parent folder.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateInstance [permission only]	Grants permission to create an instance.	Write			
CreateLabels	Grants permission to add labels to the given resource.	Write			
CreateNotification	Grants permission to configure WorkDocs to use Amazon SNS notifications.	Write			
CreateUser	Grants permission to create a user in a Simple AD or Microsoft AD directory.	Write			
DeactivateUser	Grants permission to deactivate the specified user, which revokes the user's access to Amazon WorkDocs.	Write			
DeleteComment	Grants permission to delete the specified comment from the document version.	Write			
DeleteCustomMetadata	Grants permission to delete custom metadata from the specified resource.	Write			
DeleteDocument	Grants permission to permanently delete the specified document and its associated metadata.	Write			
DeleteFolder	Grants permission to permanently delete the specified folder and its contents.	Write			
DeleteFolderContents	Grants permission to delete the contents of the specified folder.	Write			
DeleteInstance [permission only]	Grants permission to delete an instance.	Write			
DeleteLabels	Grants permission to delete one or more labels from a resource.	Write			
DeleteNotificationSubscription	Grants permission to delete the specified subscription from the specified organization.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteUser	Grants permission to delete the specified user from a Simple AD or Microsoft AD directory.	Write			
DeregisterDirectory [permission only]	Grants permission to deregister a directory.	Write			
DescribeActivities	Grants permission to fetch user activities in a specified time period.	List			
DescribeAvailableDirectories [permission only]	Grants permission to describe available directories.	List			
DescribeComments	Grants permission to list all the comments for the specified document version.	List			
DescribeDocumentVersions	Grants permission to retrieve the document versions for the specified document.	List			
DescribeFolderContents	Grants permission to describe the contents of the specified folder, including its documents and sub-folders.	List			
DescribeGroups	Grants permission to describe the user groups.	List			
DescribeInstances [permission only]	Grants permission to describe instances.	List			
DescribeNotificationSubscriptions	Grants permission to list the specified notification subscriptions.	List			
DescribeResourcePermissions	Grants permission to view the description of a specified resource's permissions.	List			
DescribeRootFolders	Grants permission to describe the root folders.	List			
DescribeUsers	Grants permission to view a description of the specified users. You can describe all users or filter the results (for example, by status or organization).	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DownloadDocument [permission only]	Grants permission to download a specified document version.	Read			
GetCurrentUser	Grants permission to retrieve the details of the current user.	Read			
GetDocument	Grants permission to retrieve the specified document object.	Read			
GetDocumentPath	Grants permission to retrieve the path information (the hierarchy from the root folder) for the requested document.	Read			
GetDocumentVersion	Grants permission to retrieve version metadata for the specified document.	Read			
GetFolder	Grants permission to retrieve the metadata of the specified folder.	Read			
GetFolderPath	Grants permission to retrieve the path information (the hierarchy from the root folder) for the specified folder.	Read			
GetResources	Grants permission to get a collection of resources.	Read			
InitiateDocument	Grants permission to create a new document object and version object.	Write			
RegisterDirectory [permission only]	Grants permission to register a directory.	Write			
RemoveAllResources	Grants permission to remove all the permissions from the specified resource.	Write			
RemoveResourcePermissions	Grants permission to remove the permission for the specified principal from the specified resource.	Write			
UpdateDocument	Grants permission to update the specified attributes of the specified document.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UpdateDocumentStatus	Grants permission to change the status of the document version to ACTIVE.	Write			
UpdateFolder	Grants permission to update the specified attributes of the specified folder.	Write			
UpdateInstanceAlias [permission only]	Grants permission to update an instance alias .	Write			
UpdateUser	Grants permission to update the specified attributes of the specified user, and grants or revokes administrative privileges to the Amazon WorkDocs site.	Write			

Resource Types Defined by Amazon WorkDocs

Amazon WorkDocs does not support specifying a resource ARN in the Resource element of an IAM policy statement. To allow access to Amazon WorkDocs, specify "Resource": "*" in your policy.

Condition Keys for Amazon WorkDocs

WorkDocs has no service-specific context keys that can be used in the Condition element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Amazon WorkLink

Amazon WorkLink (service prefix: `worklink`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon WorkLink \(p. 1756\)](#)
- [Resource Types Defined by Amazon WorkLink \(p. 1759\)](#)
- [Condition Keys for Amazon WorkLink \(p. 1759\)](#)

Actions Defined by Amazon WorkLink

You can specify the following actions in the Action element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually

allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the Resource element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AssociateDomain	Grants permission to associate a domain with an Amazon WorkLink fleet	Write	fleet* (p. 1759)		
AssociateWebsiteAuthorization	Grants permission to associate a website authorization provider with an Amazon WorkLink fleet	Write	fleet* (p. 1759)		
AssociateWebsiteCertificate	Grants permission to associate a website certificate authority with an Amazon WorkLink fleet	Write	fleet* (p. 1759)		
CreateFleet	Grants permission to create an Amazon WorkLink fleet	Write			
DeleteFleet	Grants permission to delete an Amazon WorkLink fleet	Write	fleet* (p. 1759)		
DescribeAuditStream	Grants permission to describe the audit stream configuration for an Amazon WorkLink fleet	Read	fleet* (p. 1759)		
DescribeCompanyNetwork	Grants permission to describe the company network configuration for an Amazon WorkLink fleet	Read	fleet* (p. 1759)		
DescribeDevice	Grants permission to describe details of a device associated with an Amazon WorkLink fleet	Read	fleet* (p. 1759)		
DescribeDevicePolicy	Grants permission to describe the device policy configuration for an Amazon WorkLink fleet	Read	fleet* (p. 1759)		
DescribeDomain	Grants permission to describe details about a domain associated with an Amazon WorkLink fleet	Read	fleet* (p. 1759)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeFleetMetadata	Grants permission to describe metadata of an Amazon WorkLink fleet	Read	fleet* (p. 1759)		
DescribeIdentityProvider	Grants permission to describe the identity provider configuration for an Amazon WorkLink fleet	Read	fleet* (p. 1759)		
DescribeWebsiteAuthority	Grants permission to describe a website authority associated with an Amazon WorkLink fleet	Read	fleet* (p. 1759)		
DisassociateDomain	Grants permission to disassociate a domain from an Amazon WorkLink fleet	Write	fleet* (p. 1759)		
DisassociateWebsiteAuthorizationProvider	Grants permission to disassociate a website authorization provider from an Amazon WorkLink fleet	Write	fleet* (p. 1759)		
DisassociateWebsiteCertificateAuthority	Grants permission to disassociate a website certificate authority from an Amazon WorkLink fleet	Write	fleet* (p. 1759)		
ListDevices	Grants permission to list the devices associated with an Amazon WorkLink fleet	List	fleet* (p. 1759)		
ListDomains	Grants permission to list the associated domains for an Amazon WorkLink fleet	List	fleet* (p. 1759)		
ListFleets	Grants permission to list the Amazon WorkLink fleets associated with the account	List			
ListWebsiteAuthorities	Grants permission to list the website authorities for an Amazon WorkLink fleet	List	fleet* (p. 1759)		
ListWebsiteCertificates	Grants permission to list the website certificates associated with an Amazon WorkLink fleet	List	fleet* (p. 1759)		
RestoreDomainAccess	Grants permission to restore access to a domain associated with an Amazon WorkLink fleet	Write	fleet* (p. 1759)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
RevokeDomainAccess	Grants permission to revoke access to a domain associated with an Amazon WorkLink fleet	Write	fleet* (p. 1759)		
SignOutUser	Grants permission to sign out a user from an Amazon WorkLink fleet	Write	fleet* (p. 1759)		
UpdateAuditStreamConfiguration	Grants permission to update the audit stream configuration for an Amazon WorkLink fleet	Write	fleet* (p. 1759)		
UpdateCompanyNetworkConfiguration	Grants permission to update the company network configuration for an Amazon WorkLink fleet	Write	fleet* (p. 1759)		
UpdateDevicePolicy	Grants permission to update the device policy configuration for an Amazon WorkLink fleet	Write	fleet* (p. 1759)		
UpdateDomainMetadata	Grants permission to update the metadata for a domain associated with an Amazon WorkLink fleet	Write	fleet* (p. 1759)		
UpdateFleetMetadata	Grants permission to update the metadata of an Amazon WorkLink fleet	Write	fleet* (p. 1759)		
UpdateIdentityProviderConfiguration	Grants permission to update the identity provider configuration for an Amazon WorkLink fleet	Write	fleet* (p. 1759)		

Resource Types Defined by Amazon WorkLink

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1756\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
fleet	arn:\${Partition}:worklink::\${Account}:fleet/\${fleetName}	

Condition Keys for Amazon WorkLink

WorkLink has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Amazon WorkMail

Amazon WorkMail (service prefix: `workmail`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon WorkMail \(p. 1760\)](#)
- [Resource Types Defined by Amazon WorkMail \(p. 1767\)](#)
- [Condition Keys for Amazon WorkMail \(p. 1767\)](#)

Actions Defined by Amazon WorkMail

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AddMembersToGroup [permission only]	Adds a list of members (users or groups) to a group.	Write			
AssociateDelegateToGroup	Adds a member (user or group) to the resource's set of delegates.	Write			
AssociateMemberToGroup	Adds a member (user or group) to the group's set.	Write			
CreateAlias	Adds an alias to the set of a given member (user or group) of WorkMail.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
CreateGroup	Creates a group that can be used in WorkMail by calling the RegisterToWorkMail operation.	Write			
CreateInboundMailFlow [permission only]	Create an inbound email flow which will apply to all email sent to an organization	Write			
CreateMailDomain [permission only]	Creates a mail domain.	Write			
CreateMailUser [permission only]	Creates a user in the directory and the WorkMail storage but does not enable the user for mail.	Write			
CreateOrganization [permission only]	Creates an organization, either using an existing directory or creates a new directory on-the-fly. Also creates and enables the complementary mail domain. Optionally creates KMS key	Write			
CreateOutboundMailFlow [permission only]	Create an outbound email flow which will apply to all email sent from an organization	Write			
CreateResource	Creates a new WorkMail resource.	Write			
CreateSmtpGateway [permission only]	Register an SMTP device against a WorkMail organization	Write			
CreateUser	Creates a user who can be used in WorkMail by calling the RegisterToWorkMail operation.	Write			
DeleteAlias	Remove one or more specified aliases from a set of aliases for a given user.	Write			
DeleteGroup	Deletes a group from WorkMail.	Write			
DeleteInboundMailRule [permission only]	Remove an inbound email flow rule to no longer apply to emails sent to an organization	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DeleteMailDomain [permission only]	Removes an unused mail domain from an organization	Write			
DeleteMailboxPermissions	Deletes permissions granted to a member (user or group).	Write			
DeleteMobileDevice [permission only]	Removes a mobile device from a user	Write			
DeleteOrganization [permission only]	Removes an organization from an account, either removing the directory from directory services or leaving it available for re-use	Write			
DeleteOutboundMailFlowRule [permission only]	Remove an outbound email flow rule. The rule no longer apply to emails sent from an organization	Write			
DeleteResource	Deletes the specified resource.	Write			
DeleteSmtpGateway [permission only]	Remove an SMTP device from an organization	Write			
DeleteUser	Deletes a user from WorkMail and all subsequent systems. The action cannot be undone.	Write			
DeregisterFromWorkMail	Mark a user, group, or resource as no longer used in WorkMail.	Write			
DescribeDirectories [permission only]	Shows a list of directories available for use in creating an organization	List			
DescribeGroup	Returns the data available for the group.	List			
DescribeInboundMailFlowRule [permission only]	Returns the details of an inbound mail flow rule configured for an organization	Read			
DescribeKmsKeys [permission only]	Shows a list of KMS Keys available for use in creating an organization	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
DescribeMailDomains [permission only]	Shows the details of all mail domains associated with the organization	List			
DescribeMailGroups [permission only]	Shows the details of all groups associated with the organization	List			
DescribeMailUsers [permission only]	Shows the details of all users associated with the organization	List			
DescribeOrganization	Provides more information regarding a given organization based on its identifier.	List			
DescribeOrganizations [permission only]	Shows a summary of all organizations associated with the account	List			
DescribeOutboundMailFlowRules [permission only]	Returns the details of an Outbound Mail flow rule configured for an organization	Read			
DescribeResource	Returns the data available for the resource.	List			
DescribeSmtpGateways [permission only]	Returns the details of an SMTP device registered against an organization	Read			
DescribeUser	Provides information regarding the user.	List			
DisableMailGroup [permission only]	Disable a mail group when it is not being used and, to allow it to be deleted	Write			
DisableMailUsers [permission only]	Disable a user mailbox when it is no longer being used, and to allow it to be deleted	Write			
DisassociateDelegates [resource set of delegates]	Removes a member from the resource set of delegates.	Write			
DisassociateMembers [group]	Removes a member from a group	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
EnableMailDomain [permission only]	Enable a mail domain in the organization	Write			
EnableMailGroup [permission only]	Enable a mail group after it has been created to allow it to receive mail	Write			
EnableMailUser [permission only]	Enable a user's mailbox after it has been created to allow it to receive mail	Write			
GetJournalingRules [permission only]	Returns journaling and fallback email addresses configured for email journaling	Read			
GetMailDomainDetails [permission only]	Get the details of the mail domain	Read			
GetMailGroupDetails [permission only]	Get the details of the mail group	Read			
GetMailUserDetails [permission only]	Get the details of the user's mailbox and account	Read			
GetMailboxDetails	Returns the details of the user's mailbox.	Read			
GetMobileDeviceDetails [permission only]	Get the details of the mobile device	Read			
GetMobileDevicesAssociated [permission only]	Get a list of the mobile devices associated with the user	Read			
GetMobilePolicyDevice [permission only]	Get the details of the mobile device policy associated with the organization	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
ListAliases	Creates a paginated call to list the aliases associated with a given entity.	List			
ListGroupMembers	Returns an overview of the members of a group. Users and groups can be members of a group.	List			
ListGroups	Returns summaries of the organization's groups.	List			
ListInboundMailFlowRules [permission only]	Returns a list of inbound mail flow rules configured for an organization	List			
ListMailboxPermissions	Lists the mailbox permissions associated with a user, group, or resource mailbox.	List			
ListMembersInMailGroup [permission only]	Get a list of all the members in a mailgroup	Read			
ListOrganizations	Returns summaries of the customer's non-deleted organizations.	List			
ListOutboundMailFlowRules [permission only]	Returns a list of outbound mail flow rules configured for an organization	List			
ListResourceDelegates	Lists the delegates associated with a resource.	List			
ListResources	Returns summaries of the organization's resources.	List			
ListSmtpGateway [permission only]	Returns a list of SMTP devices registered against the organization	List			
ListUsers	Returns summaries of the organization's users.	List			
PutMailboxPermissions	Sets permissions for a user, group, or resource. This replaces any pre-existing permissions.	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
RegisterToWorkMail [permission only]	Registers an existing and disabled user, group, or resource for use by associating a mailbox and calendaring capabilities.	Write			
RemoveMembersFromGroup [permission only]	Remove members from a mail group	Write			
ResetPassword [permission only]	Allows the administrator to reset the password for a user.	Write			
ResetUserPassword [permission only]	Reset the password for a user's account	Write			
SearchMembers [permission only]	Prefix search to find a specific user in a mail group	Read			
SetAdmin [permission only]	Mark a user as being an administrator	Write			
SetDefaultMailDomain [permission only]	Set the default mail domain for the organization	Write			
SetJournalingRules [permission only]	Set journaling and fallback email addresses for email journaling	Write			
SetMailGroupDetails [permission only]	Set the details of the mail group which has just been created	Write			
SetMailUserDetails [permission only]	Set the details for the user account which has just been created	Write			
SetMobilePolicyDetails [permission only]	Set the details of a mobile policy associated with the organization	Write			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
TestInboundMailFlow [permission only]	Test what inbound rules will apply to an email with a given sender and recipient	Write			
TestOutboundMailFlow [permission only]	Test what outbound rules will apply to an email with a given sender and recipient	Write			
UpdateInboundMailFlow [permission only]	Update the details of an inbound mail flow rule which will apply to all email sent to an organization	Write			
UpdateMailboxQuota	Updates the maximum size (in MB) of the user's mailbox.	Write			
UpdateOutboundMailFlow [permission only]	Update the details of an outbound mail flow rule which will apply to all email sent from an organization	Write			
UpdatePrimaryEmail [user, group]	Updates the primary email for a user, group, or resource.	Write			
UpdateResource	Updates data for the resource. To retrieve the latest information, it must be preceded by a <code>DescribeResource</code> call.	Write			
UpdateSmtpGateway [permission only]	Update the details of an existing SMTP device registered against an organization	Write			
WipeMobileDevice [permission only]	Remotely wipe the mobile device associated with a user's account	Write			

Resource Types Defined by Amazon WorkMail

Amazon WorkMail does not support specifying a resource ARN in the `Resource` element of an IAM policy statement. To allow access to Amazon WorkMail, specify “`Resource`”: “`*`” in your policy.

Condition Keys for Amazon WorkMail

WorkMail has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Amazon WorkMail Message Flow

Amazon WorkMail Message Flow (service prefix: `workmailmessageflow`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon WorkMail Message Flow \(p. 1768\)](#)
- [Resource Types Defined by Amazon WorkMail Message Flow \(p. 1768\)](#)
- [Condition Keys for Amazon WorkMail Message Flow \(p. 1769\)](#)

Actions Defined by Amazon WorkMail Message Flow

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetRawMessageContent	Grants permission to read the <code>Content</code> of email messages with the specified message ID	Read	RawMessage* (p. 1769)		

Resource Types Defined by Amazon WorkMail Message Flow

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1768\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
RawMessage	arn:\${Partition}:workmailmessageflow: \${Region}:\${Account}:message/ \${OrganizationId}/\${Context}/\${MessageId}	

Condition Keys for Amazon WorkMail Message Flow

WorkMail Message Flow has no service-specific context keys that can be used in the Condition element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Amazon WorkSpaces

Amazon WorkSpaces (service prefix: `workspaces`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon WorkSpaces \(p. 1769\)](#)
- [Resource Types Defined by Amazon WorkSpaces \(p. 1772\)](#)
- [Condition Keys for Amazon WorkSpaces \(p. 1772\)](#)

Actions Defined by Amazon WorkSpaces

You can specify the following actions in the Action element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the Resource element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AssociateIpGroup	Associates the specified IP access control group with the specified directory.	Write			
AuthorizeIpRules	Adds one or more rules to the specified IP access control group.	Write	workspaceipgroup* (p. 1772)		
CreateIpGroup	Creates an IP access control group.	Write			
CreateTags	Creates tags for a WorkSpace.	Tagging			
CreateWorkspaces	Creates one or more WorkSpaces.	Write	directoryid* (p. 1772)		
			workspacebundle* (p. 1772)		
DeleteIpGroup	Deletes the specified IP access control group.	Write	workspaceipgroup* (p. 1772)		
DeleteTags	Deletes tags from a Workspace.	Write			
DeleteWorkspaceImage	Deletes the specified workspace image.	Write			
DescribeAccount	Retrieves a list that describes the configuration of bring your own license (BYOL) for the specified account.	List			
DescribeAccountModifications	Retrieves a list that describes modifications to the configuration of bring your own license (BYOL) for the specified account.	List			
DescribeClientProperties	Describes client properties about the specified resources.	List	directoryid* (p. 1772)		
DescribeIpGroups	Retrieves information about the IP access control groups of your account in the region.	List	workspaceipgroup* (p. 1772)		
DescribeTags	Describes tags for a WorkSpace.	List			
DescribeWorkspaces	Obtains information about the WorkSpace bundles that are available to your account in the specified region.	List	workspacebundle* (p. 1772)		
DescribeWorkspacesAWSDirectory	Retrieves information about the AWS Directory Service directories in the region that are registered with Amazon	List			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
	WorkSpaces and are available to your account.				
DescribeWorkspaces	Retrieves a list that describes one or more specified images.	List			
DescribeWorkspaces	Obtains information about the specified WorkSpaces.	List			
DescribeWorkspaces	Describes the connection status of a specified WorkSpace.	Read			
DisassociateIpGroups	Disassociates the specified IP access control group from the specified directory.	Write			
ImportWorkspaceImages	Import a licensed EC2 image to Amazon WorkSpaces.	Write			
ListAvailableManagerCidrRanges	List available CIDR ranges for a constraint.	List			
ModifyAccount	Modify the configuration of bring your own license (BYOL) for the specified account.	Write			
ModifyClientProperties	Modify the client properties of a specified resource.	Write	directoryid* (p. 1772)		
ModifyWorkspaceProperties	Modifies the WorkSpace Properties, including the running mode and AutoStop time.	Write	workspaceid* (p. 1772)		
ModifyWorkspaceState	Modify the state of specified WorkSpaces.	Write	workspaceid* (p. 1772)		
RebootWorkspaces	Reboots the specified WorkSpaces.	Write	workspaceid* (p. 1772)		
RebuildWorkspaces	Rebuilds the specified WorkSpaces.	Write	workspaceid* (p. 1772)		
RevokeIpRules	Removes one or more rules from the specified IP access control group.	Write	workspaceipgroup* (p. 1772)		
StartWorkspaces	Starts the specified WorkSpaces.	Write	workspaceid* (p. 1772)		
StopWorkspaces	Stops the specified WorkSpaces.	Write	workspaceid* (p. 1772)		
TerminateWorkspaces	Terminates the specified WorkSpaces.	Write	workspaceid* (p. 1772)		

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
UpdateRulesOfIpGroup	Replaces the current rules of the specified IP access control group with the specified rules.	Write	workspaceipgroup* (p. 1772)		

Resource Types Defined by Amazon WorkSpaces

The following resource types are defined by this service and can be used in the Resource element of IAM permission policy statements. Each action in the [Actions table \(p. 1769\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
workspacebundle	arn:\${Partition}:workspaces:\${Region}: \${Account}:workspacebundle/\${BundleId}	
workspaceipgroup	arn:\${Partition}:workspaces:\${Region}: \${Account}:workspaceipgroup/\${GroupId}	
directoryid	arn:\${Partition}:workspaces:\${Region}: \${Account}:directory/\${DirectoryId}	
workspaceid	arn:\${Partition}:workspaces:\${Region}: \${Account}:workspace/\${WorkspaceId}	

Condition Keys for Amazon WorkSpaces

WorkSpaces has no service-specific context keys that can be used in the Condition element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for Amazon WorkSpaces Application Manager

Amazon WorkSpaces Application Manager (service prefix: wam) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by Amazon WorkSpaces Application Manager \(p. 1773\)](#)
- [Resource Types Defined by Amazon WorkSpaces Application Manager \(p. 1773\)](#)
- [Condition Keys for Amazon WorkSpaces Application Manager \(p. 1773\)](#)

Actions Defined by Amazon WorkSpaces Application Manager

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
AuthenticatePackage [permission only]	Allows the Amazon WAM packaging instance to access your application package catalog.	Write			

Resource Types Defined by Amazon WorkSpaces Application Manager

Amazon WorkSpaces Application Manager does not support specifying a resource ARN in the `Resource` element of an IAM policy statement. To allow access to Amazon WorkSpaces Application Manager, specify "Resource": "*" in your policy.

Condition Keys for Amazon WorkSpaces Application Manager

WAM has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Actions, Resources, and Condition Keys for AWS X-Ray

AWS X-Ray (service prefix: `xray`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

References:

- Learn how to [configure this service](#).
- View a list of the [API operations available for this service](#).
- Learn how to secure this service and its resources by [using IAM permission policies](#).

Topics

- [Actions Defined by AWS X-Ray \(p. 1774\)](#)
- [Resource Types Defined by AWS X-Ray \(p. 1775\)](#)
- [Condition Keys for AWS X-Ray \(p. 1776\)](#)

Actions Defined by AWS X-Ray

You can specify the following actions in the `Action` element of an IAM policy statement. Use policies to grant permissions to perform an operation in AWS. When you use an action in a policy, you usually allow or deny access to the API operation or CLI command with the same name. However, in some cases, a single action controls access to more than one operation. Alternatively, some operations require several different actions.

The **Resource Types** column indicates whether each action supports resource-level permissions. If there is no value for this column, you must specify all resources ("*") in the `Resource` element of your policy statement. If the column includes a resource type, then you can specify an ARN of that type in a statement with that action. Required resources are indicated in the table with an asterisk (*). If you specify a resource-level permission ARN in a statement using this action, then it must be of this type. Some actions support multiple resource types. If the resource type is optional (not indicated as required), then you can choose to use one but not the other.

For details about the columns in the following table, see [The Actions Table \(p. 686\)](#).

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
BatchGetTraces	Retrieves a list of traces specified by ID. Each trace is a collection of segment documents that originates from a single request. Use <code>GetTraceSummaries</code> to get a list of trace IDs.	Read			
CreateGroup	Creates a group resource with a name and a filter expression.	Write	group* (p. 1776)		
CreateSamplingRule	Creates a rule to control sampling behavior for instrumented applications.	Write	sampling-rule* (p. 1776)		
DeleteGroup	Deletes a group resource.	Write	group* (p. 1776)		
DeleteSamplingRule	Deletes a sampling rule.	Write	sampling-rule* (p. 1776)		
GetEncryptionConfig	Retrieves the current encryption configuration for X-Ray data.	Permissions management			
GetGroup	Retrieves group resource details.	Read	group* (p. 1776)		
GetGroups	Retrieves all active group details.	Read			
GetSamplingRules	Retrieves all sampling rules.	Read			
GetSamplingStatistics	Retrieves information about recent sampling results for all sampling rules.	Read			

Actions	Description	Access Level	Resource Types (*required)	Condition Keys	Dependent Actions
GetSamplingTargets	Requests a sampling quota for rules that the service is using to sample requests.	Read			
GetServiceGraph	Retrieves a document that describes services that process incoming requests, and downstream services that they call as a result.	Read			
GetTimeSeriesStatistics	Get an aggregation of service statistics defined by a specific time range bucketed into time intervals.	Read			
GetTraceGraph	Retrieves a service graph for one or more specific trace IDs.	Read			
GetTraceSummaries	Retrieves IDs and metadata for traces available for a specified time frame using an optional filter. To get the full traces, pass the trace IDs to BatchGetTraces.	Read			
PutEncryptionConfig	Updates the encryption configuration for X-Ray data.	Permissions management			
PutTelemetryRecords	Used by the AWS X-Ray daemon to send telemetry to the service.	Write			
PutTraceSegment	Uploads segment documents to AWS X-Ray. The X-Ray SDK generates segment documents and sends them to the X-Ray daemon, which uploads them in batches.	Write			
UpdateGroup	Updates a group resource.	Write	group* (p. 1776)		
UpdateSamplingRules	Modifies a sampling rule's configuration.	Write	sampling-rule* (p. 1776)		

Resource Types Defined by AWS X-Ray

The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action in the [Actions table \(p. 1774\)](#) identifies the resource types that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the last column of the table. For details about the columns in the following table, see [The Resource Types Table \(p. 687\)](#).

Resource Types	ARN	Condition Keys
group	arn:\${Partition}:xray:\${Region}: \${Account}:group/\${GroupName}/\${Id}	
sampling-rule	arn:\${Partition}:xray:\${Region}: \${Account}:sampling-rule/\${SamplingRuleName}	

Condition Keys for AWS X-Ray

X-Ray has no service-specific context keys that can be used in the `Condition` element of policy statements. For the list of the global context keys that are available to all services, see [Available Keys for Conditions](#) in the *IAM Policy Reference*.

Resources

IAM is a rich product, and you'll find many resources to help you learn more about how IAM can help you secure your AWS account and resources.

Topics

- [Users and Groups \(p. 1777\)](#)
- [Credentials \(Passwords, Access Keys, and MFA devices\) \(p. 1777\)](#)
- [Permissions and Policies \(p. 1777\)](#)
- [Federation and Delegation \(p. 1778\)](#)
- [IAM and Other AWS Products \(p. 1778\)](#)
- [General Security Practices \(p. 1779\)](#)
- [General Resources \(p. 1779\)](#)

Users and Groups

Consult these resources for creating, managing, and using users and groups.

- [Creating Your First IAM Admin User and Group \(p. 20\)](#) – A step-by-step procedure that shows how to create an IAM users and assign permissions.
- [Identities \(Users, Groups, and Roles\) \(p. 82\)](#) – An in-depth discussion of how to administer IAM users and groups.
- [Guidelines for When to Use Accounts, Users, and Groups](#) – An AWS Security Blog post that discusses how to organize user access with separate AWS accounts or with IAM users and groups in a single account.

Credentials (Passwords, Access Keys, and MFA devices)

Review the following guides to manage passwords for your AWS account and for IAM users. You'll also find information about *access keys*—the secret key that you use to make programmatic calls to AWS.

- [AWS Security Credentials](#) – Describes the types of credentials you use to access Amazon Web Services, explains how to create and manage them, and includes recommendations for managing access keys securely.
- [Managing Passwords \(p. 100\)](#) and [Managing Access Keys for IAM Users \(p. 110\)](#) – Describes options for managing credentials for IAM users in your account.
- [Using Multi-Factor Authentication \(MFA\) in AWS \(p. 119\)](#) – Describes how to configure your account and IAM users to require both a password and a one-time use code that is generated on a device before sign-in is allowed. (This is sometimes called two-factor authentication.)

Permissions and Policies

Learn the inner workings of IAM policies and find tips on the best ways to confer permissions:

- [Policies and Permissions \(p. 356\)](#) – Introduces the policy language that is used to define permissions. Describes how permissions can be attached to users or groups or, for some AWS products, to resources themselves.
- [IAM JSON Policy Elements Reference \(p. 598\)](#) – Provides descriptions and examples of each policy language element.
- [Example IAM Identity-Based Policies \(p. 394\)](#) – Shows examples of policies for common tasks in various AWS products.
- [AWS Policy Generator](#) – Create custom policies by choosing products and actions from a list.
- [IAM Policy Simulator](#) – Test whether a policy would allow or deny a specific request to AWS.

Federation and Delegation

You can grant access to resources in your AWS account for users who are authenticated (signed in) elsewhere. These can be IAM users in another AWS account (known as *delegation*), users who are authenticated with your organization's sign-in process, or users from an Internet identity provider like Login with Amazon, Facebook, Google, or any other OpenID Connect (OIDC) compatible identity provider. In these cases, the users get temporary security credentials to access AWS resources.

- [Tutorial: Delegate Access Across AWS Accounts Using IAM Roles \(p. 30\)](#) – Guides you through granting cross-account access to an IAM user in another AWS account.
- [Common Scenarios for Temporary Credentials \(p. 308\)](#) – Describes ways in which users can be federated into AWS after being authenticated outside of AWS.
- [Web Identity Federation Playground](#) – Lets you experiment with Login with Amazon, Google, or Facebook to authenticate and then make a call to Amazon S3.

IAM and Other AWS Products

Most AWS products are integrated with IAM so that you can use IAM features to help protect access to the resources in those products. The following resources discuss IAM and security for some of the most popular AWS products. For a complete list of products that work with IAM, including links to more information on each, see [AWS Services That Work with IAM \(p. 586\)](#).

Using IAM with Amazon EC2

- [Controlling Access to Amazon EC2 Resources](#) – Describes how to use IAM features to permit users to administer Amazon EC2 instances, volumes, and more.
- [Using Instance Profiles \(p. 276\)](#) – Describes how to use IAM roles to securely provide credentials for applications that run on Amazon EC2 instances and that need access to other AWS products.

Using IAM with Amazon S3

- [Managing Access Permissions to Your Amazon S3 Resources](#) – Discusses the Amazon S3 security model for buckets and objects, which includes IAM policies.
- [Writing IAM Policies: Grant Access to User-Specific Folders in an Amazon S3 Bucket](#) – Discusses how to let users protect their own folders in Amazon S3. (For more posts about Amazon S3 and IAM, choose the **S3** tag below the title of the blog post.)

Using IAM with Amazon RDS

- [Using AWS Identity and Access Management \(IAM\) to Manage Access to Amazon RDS Resources](#) – Describes how to use IAM to control access to database instances, database snapshots, and more.
- [A Primer on RDS Resource-Level Permissions](#) – Describes how to use IAM to control access to specific Amazon RDS instances.

Using IAM with Amazon DynamoDB

- [Using IAM to Control Access to DynamoDB Resources](#) – Describes how to use IAM to permit users to administer DynamoDB tables and indexes.
- The following video (8:55) explains how to provide access control for individual DynamoDB database items or attributes (or both).

[Getting Started with Fine-Grained Access Control for DynamoDB](#)

General Security Practices

Find expert tips and guidance on the best ways to secure your AWS account and resources:

- [AWS Security Best Practices \(PDF\)](#) – Provides an in-depth look at how to manage security across AWS accounts and products, including suggestions for security architecture, use of IAM, encryption and data security, and more.
- [IAM Best Practices \(p. 60\)](#) – Offers recommendations for ways to use IAM to help secure your AWS account and resources.
- [AWS CloudTrail User Guide](#) – Use AWS CloudTrail to track a history of API calls made to AWS and store that information in log files. This helps you determine which users and accounts accessed resources in your account, when the calls were made, what actions were requested, and more.

General Resources

Explore the following resources to learn more about IAM and AWS.

- [Product Information for IAM](#) – General information about the AWS Identity and Access Management product.
- [Discussion Forums for AWS Identity and Access Management](#) – A community forum for customers to discuss technical questions related to IAM.
- [Classes & Workshops](#) – Links to role-based and specialty courses as well as self-paced labs to help sharpen your AWS skills and gain practical experience.
- [AWS Developer Tools](#) – Links to developer tools, SDKs, IDE toolkits, and command line tools for developing and managing AWS applications.
- [AWS Whitepapers](#) – Links to a comprehensive list of technical AWS whitepapers, covering topics such as architecture, security, and economics and authored by AWS Solutions Architects or other technical experts.
- [AWS Support Center](#) – The hub for creating and managing your AWS Support cases. Also includes links to other helpful resources, such as forums, technical FAQs, service health status, and AWS Trusted Advisor.

- **AWS Support** – The primary web page for information about AWS Support, a one-on-one, fast-response support channel to help you build and run applications in the cloud.
- **Contact Us** – A central contact point for inquiries concerning AWS billing, account, events, abuse, and other issues.
- **AWS Site Terms** – Detailed information about our copyright and trademark; your account, license, and site access; and other topics.

Calling the API by Making HTTP Query Requests

Topics

- [Endpoints \(p. 1781\)](#)
- [HTTPS Required \(p. 1782\)](#)
- [Signing IAM API Requests \(p. 1782\)](#)

This section contains general information about using the Query API for AWS Identity and Access Management (IAM) and AWS Security Token Service (AWS STS). For details about the API actions and errors, go to the [IAM API Reference](#) or the [AWS Security Token Service API Reference](#).

Note

Instead of making direct calls to the IAM or AWS STS API operations, you can use one of the AWS SDKs. The AWS SDKs consist of libraries and sample code for various programming languages and platforms (Java, Ruby, .NET, iOS, Android, etc.). The SDKs provide a convenient way to create programmatic access to IAM and AWS. For example, the SDKs take care of tasks such as cryptographically signing requests (see below), managing errors, and retrying requests automatically. For information about the AWS SDKs, including how to download and install them, see the [Tools for Amazon Web Services](#) page.

The Query API for IAM and AWS STS lets you call service actions. Query API requests are HTTPS requests that must contain an `Action` parameter to indicate the action to be performed. IAM and AWS STS support GET and POST requests for all actions. That is, the API does not require you to use GET for some actions and POST for others. However, GET requests are subject to the limitation size of a URL; although this limit is browser dependent, a typical limit is 2048 bytes. Therefore, for Query API requests that require larger sizes, you must use a POST request.

The response is an XML document. For details about the response, see the individual action pages in the [IAM API Reference](#) or the [AWS Security Token Service API Reference](#).

Endpoints

IAM and AWS STS each have a single global endpoint:

- (IAM) <https://iam.amazonaws.com>
- (AWS STS) <https://sts.amazonaws.com>

Note

AWS STS also supports sending requests to regional endpoints in addition to the global endpoint. Before you can use AWS STS in a Region, you must first activate STS in that Region for your AWS account. For more information about activating additional Regions for AWS STS, see [Managing AWS STS in an AWS Region \(p. 333\)](#).

For more information about AWS endpoints and Regions for all services, see [Regions and Endpoints](#) in the [AWS General Reference](#).

HTTPS Required

Because the Query API returns sensitive information such as security credentials, you must use HTTPS with all API requests.

Signing IAM API Requests

Requests must be signed using an access key ID and a secret access key. We strongly recommend that you do not use your AWS account root user credentials for everyday work with IAM. You can use the credentials for an IAM user or you can use AWS STS to generate temporary security credentials.

To sign your API requests, we recommend using AWS Signature Version 4. For information about using Signature Version 4, go to [Signature Version 4 Signing Process](#) in the *AWS General Reference*.

If you need to use Signature Version 2, information about using Signature Version 2 is available in the [AWS General Reference](#).

For more information, see the following:

- [AWS Security Credentials](#). Provides general information about the types of credentials used for accessing AWS.
- [IAM Best Practices \(p. 60\)](#). Presents a list of suggestions for using IAM service to help secure your AWS resources.
- [Temporary Security Credentials \(p. 307\)](#). Describes how to create and use temporary security credentials.

Document History for IAM

The following table describes major documentation updates for IAM.

update-history-change	update-history-description	update-history-date
Policy Simulator adds support for Permissions Boundaries	You can now test the effect of permissions boundaries on IAM entities with the IAM policy simulator.	January 23, 2020
Cross-Account Policy Evaluation	You can now learn how AWS evaluates policies for cross-account access. This occurs when a resource in a trusting account includes a resource-based policy that allows a principal in another account to access the resource. The request must be allowed in both accounts.	January 2, 2020
Session Tags	You can now include tags when you assume a role or federate a user in AWS STS. When you perform the <code>AssumeRole</code> or <code>GetFederationToken</code> operation, you can pass the session tags as attributes. When you perform the <code>AssumeRoleWithSAML</code> or <code>AssumeRoleWithWebIdentity</code> operations, you can pass attributes from your corporate identities to AWS.	November 22, 2019
Control access for groups of AWS accounts in AWS Organizations	You can now reference organizational units (OUs) from AWS Organizations in IAM policies. If you use Organizations to organize your accounts into OUs, you can require that principals belong to a specific OU before granting access to your resources. Principals include AWS account root user, IAM users and IAM roles. To do this, specify the OU path in the <code>aws:PrincipalOrgPaths</code> condition key in your policies.	November 20, 2019
Role Last Used	You can now view the date, time, and Region where a role was last used. This information also helps you identify unused roles in your account. You can use	November 19, 2019

	<p>the AWS Management Console, AWS CLI and AWS API to view information about when a role was last used.</p>	
Update to the Global Condition Context Keys Page	<p>You can now learn when each of the global condition keys is included in the context of a request. You can also navigate to each key more easily using the page table of contents (TOC). The information on the page helps you to write more accurate policies. For example, if your employees use federation with IAM roles, you should use the <code>aws:userId</code> key and not the <code>aws:userName</code> key. The <code>aws:userName</code> key applies only to IAM users and not roles.</p>	October 6, 2019
ABAC in AWS	<p>Learn how attribute-based access control (ABAC) works in AWS using tags, and how it compares to the traditional AWS authorization model. Use the ABAC tutorial to learn how to create and test a policy that allows IAM roles with principal tags to access resources with matching tags. This strategy allows individuals to view or edit only the AWS resources required for their jobs.</p>	October 3, 2019
AWS STS GetAccessKeyInfo Operation	<p>You can review the AWS access keys in your code to determine whether the keys are from an account that you own. You can pass an access key ID using the <code>aws sts get-access-key-info</code> AWS CLI command or the <code>GetAccessKeyInfo</code> AWS API operation.</p>	July 24, 2019

Viewing Organizations Service Last Accessed Data in IAM	You can now view service last accessed data for an AWS Organizations entity or policy in the AWS Organizations section of the IAM console. You can also use the AWS CLI or AWS API to retrieve the data report. This data includes information about the allowed services that principals in an Organizations account last attempted to access and when. You can use this information to identify unnecessary permissions so that you can refine your Organizations policies to better adhere to the principle of least privilege.	June 20, 2019
Using a Managed Policy as a Session Policy	You can now pass up to 10 managed policy ARNs when you assume a role. This allows you to limit the permissions of the role's temporary credentials.	May 7, 2019
AWS STS Region Compatibility of Session Tokens for the Global Endpoint	You can now choose whether to use version 1 or version 2 global endpoint tokens. Version 1 tokens are valid only in AWS Regions that are available by default. These tokens will not work in manually enabled Regions, such as Asia Pacific (Hong Kong). Version 2 tokens are valid in all Regions. However, version 2 tokens are longer and might affect systems where you temporarily store tokens.	April 26, 2019
Allow Enabling and Disabling AWS Regions	You can now create a policy that allows an administrator to enable and disable the Asia Pacific (Hong Kong) Region (ap-east-1).	April 24, 2019
IAM User My Security Credentials Page	IAM users can now manage all of their own credentials on the My Security Credentials page. This AWS Management Console page displays account information such as the account ID and canonical user ID. Users can also view and edit their own passwords, access keys, X.509 certificates, SSH keys, and Git credentials.	January 24, 2019

Access Advisor API	You can now use the AWS CLI and AWS API to view service last accessed data.	December 7, 2018
Tagging IAM Users and Roles	You can now use IAM tags to add custom attributes to an identity (IAM user or role) using a tag key-value pair. You can also use tags to control an identity's access to resources or to control what tags can be attached to an identity.	November 14, 2018
U2F security keys	You can now use U2F security keys as a multi-factor authentication (MFA) option when signing in to the AWS Management Console.	September 25, 2018
Support for Amazon VPC endpoints	You can now establish a private connection between your VPC and AWS STS in the US West (Oregon) Region.	July 31, 2018
Permissions Boundaries	New feature makes it easier to grant trusted employees the ability to manage IAM permissions without also granting full IAM administrative access.	July 12, 2018
aws:PrincipalOrgID	New condition key provides an easier way to control access to AWS resources by specifying the AWS organization of IAM principals.	May 17, 2018
aws:RequestedRegion	New condition key provides an easier way to use IAM policies to control access to AWS Regions.	April 25, 2018
Increased session duration for IAM roles	An IAM role can now have a session duration of 12 hours.	March 28, 2018
Updated role-creation workflow	New workflow improves the process of creating trust relationships and attaching permissions to roles.	September 8, 2017
AWS account sign-in process	Updated AWS sign-in experience allows both root users and IAM users to use the Sign In to the Console link on the AWS Management Console's home page.	August 25, 2017
Example IAM policies	Documentation update features more than 30 example policies.	August 2, 2017

IAM best practices	Information added to the Users section of the IAM console makes it easier to follow IAM best practices.	July 5, 2017
Auto Scaling resources	Resource-level permissions can control access to and permissions for Auto Scaling resources.	May 16, 2017
Amazon RDS for MySQL and Amazon Aurora databases	Database administrators can associate database users with IAM users and roles and thus manage user access to all AWS resources from a single location.	April 24, 2017
Service-linked roles	Service-linked roles provide an easier and more secure way to delegate permissions to AWS services.	April 19, 2017
Policy summaries	New policy summaries make it easier to understand permissions in IAM policies.	March 23, 2017

AWS Glossary

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