
How does IAM work and what can I do with it?

IAM provides authentication and authorization for AWS services. A service evaluates if an AWS request is allowed or denied. Access is denied by default and is allowed only when a policy explicitly grants access. You can attach policies to roles and resources to control access across AWS. For more information, see [Understanding how IAM works](#).

What are least-privilege permissions?

When you set permissions with IAM policies, grant only the permissions required to perform a task. This practice is known as [granting least privilege](#). You can apply least-privilege permissions in IAM by defining the actions that can be taken on specific resources under specific conditions. For more information, see [Access management for AWS resources](#).

How do I get started with IAM?

To get started using IAM to manage permissions for AWS services and resources, create an IAM role and grant it permissions. For [workforce users](#), create a role that can be assumed by your identity provider. For systems, create a role that can be assumed by the service you are using, such as Amazon EC2 or AWS Lambda. After you create a role, you can attach a policy to the role to grant permissions that meet your needs. When you are just starting out, you might not know the specific permissions you need, so you can start with broader permissions. [AWS managed policies](#) provide permissions to help you get started and are available in all AWS accounts. Then, reduce permissions further by defining [customer managed policies](#) specific to your use cases. You can create and manage policies and roles in the IAM console, or via AWS APIs or the AWS CLI. For more information, see [Getting started with IAM](#).

IAM resources

[Close all](#)

What are IAM roles and how do they work?

Why should I use IAM roles?

You should use IAM roles to grant access to your AWS accounts by relying on short-term credentials, a security best practice. Authorized identities, which can be AWS services or users from your identity provider, can assume roles to make AWS requests. To grant permissions to a role, attach an IAM policy to it. For more information, see [Common scenarios for roles](#).

What are IAM users and should I still be using them?

IAM users are identities with long-term credentials. You might be using IAM users for [workforce users](#). In this case, AWS recommends using an identity provider and federating into AWS by assuming roles. You also can use roles to grant cross-account access to services and features such as AWS Lambda functions. In some scenarios, you might require IAM users with [access keys](#) that have long-term credentials with access to your AWS account. For these scenarios, AWS recommends using IAM [access last used information](#) to rotate credentials often and remove credentials that are not being used. For more information, see [Overview of AWS identity management: Users](#).

What are IAM policies?

IAM [policies](#) define permissions for the entities you attach them to. For example, to grant access to an IAM role, attach a policy to the role. The permissions defined in the policy determine whether requests are allowed or denied. You also can attach policies to some resources, such as Amazon S3 buckets, to grant direct, cross-account access. And you can attach policies to an AWS organization or organizational unit to restrict access across multiple accounts. AWS evaluates these policies when an IAM role makes a request. For more information, see [Identity-based policies](#).

Granting access

[Close all](#)

How do I grant access to services and resources by using IAM?

attach them to roles. Some AWS resources provide a way to grant access by defining a policy attached to resources, such as Amazon S3 buckets. These [resource-based policies](#) allow you to grant direct, cross-account access to the resources they are attached to. For more information, see [Access management for AWS resources](#).

How do I create IAM policies? —

To assign permissions to a role or resource, create a policy, which is a [JavaScript Object Notation \(JSON\)](#) document that defines permissions. This document includes permissions statements that grant or deny access to specific service actions, resources, and conditions. After you create a policy, you can attach it to one or more AWS roles to grant permissions to your AWS account. To grant direct, cross-account access to resources, such as Amazon S3 buckets, use resource-based policies. Create your policies in the IAM console or via AWS APIs or the AWS CLI. For more information, see [Creating IAM policies](#).

What are AWS managed policies and when should I use them? —

AWS managed policies are created and administered by AWS and cover common use cases. Getting started, you can grant broader permissions by using the AWS managed policies that are available in your AWS account and common across all AWS accounts. Then, as you refine your requirements, you can reduce permissions by defining customer managed policies specific to your use cases with the goal of achieving least-privilege permissions. For more information, see [AWS managed policies](#).

What are customer managed policies and when should I use them? —

To grant only the permissions required to perform tasks, you can create customer managed policies that are specific to your use cases and resources. Use customer managed policies to continue refining permissions for your specific requirements. For more information, see [Customer managed policies](#).

What are inline policies and when should I use them? —

Inline policies are embedded in and inherent to specific IAM roles. Use inline policies if you want to maintain a strict one-to-one relationship between a policy and the identity to which it is applied. For example, you can grant administrative permissions to ensure they are not attached to other roles. For more information, see [Inline policies](#).

you can attach resource-based policies to Amazon S3 buckets, Amazon SQS queues, VPC endpoints, and AWS Key Management Service encryption keys. For a list of services that support resource-based policies, see [AWS services that work with IAM](#). Use resource-based policies to grant direct, cross-account access. With resource-based policies, you can define who has access to a resource and which actions they can perform with it. For more information, see [Identity-based policies and resource-based policies](#).

What is role-based access control (RBAC)?

RBAC provides a way for you to assign permissions based on a person's job function, known outside of AWS as a role. IAM provides RBAC by defining IAM roles with permissions that align with job functions. You then can grant individuals access to assume these roles to perform specific job functions. With RBAC, you can audit access by looking at each IAM role and its attached permissions. For more information, see [Comparing ABAC to the traditional RBAC model](#).

How do I grant access with RBAC?

As a best practice, grant access only to the specific service actions and resources required to perform each task. This is known as [granting least privilege](#). When employees add new resources, you must update policies to allow access to those resources.

What is attribute-based access control (ABAC)?

ABAC is an authorization strategy that defines permissions based on attributes. In AWS, these attributes are called [tags](#), and you can define them on AWS resources, IAM roles, and in role sessions. With ABAC, you define a set of permissions based on the value of a tag. You can grant fine-grained permissions to specific resources by requiring the tags on the role or session to match the tags on the resource. For example, you can author a policy that grants developers access to resources tagged with the job title "developers." ABAC is helpful in environments that are growing rapidly by granting permissions to resources as they are created with specific tags. For more information, see [Attribute-Based Access Control for AWS](#).

How do I grant access by using ABAC?


resources with appropriate tags and restrict access to modify them. After your tags are in place, define a policy that grants access to specific actions and resource types, but only if the role or session tags match the resource tags. For a detailed tutorial that demonstrates how to use ABAC in AWS, see [IAM tutorial: Define permissions to access AWS resources based on tags](#).

Restricting access

[Close all](#)

How do I restrict access by using IAM? 

With AWS Identity and Access Management (IAM), all access is denied by default and requires a policy that grants access. As you manage permissions at scale, you might want to implement [permissions guardrails](#) and restrict access across your accounts. To restrict access, specify a Deny statement in any policy. If a Deny statement applies to an access request, it always prevails over an Allow statement. For example, if you allow access to all actions in AWS but deny access to IAM, any request to IAM is denied. You can include a Deny statement in any type of policy, including identity-based, resource-based, and service control policies with AWS Organizations. For more information, see [Controlling access with AWS Identity and Access Management](#)

What are AWS Organizations service control policies (SCPs) and when should I use them? 

[SCPs](#) are similar to IAM policies and use almost the same syntax. However, SCPs don't grant permissions. Instead, SCPs allow or deny access to AWS services for individual AWS accounts with Organizations member accounts, or for groups of accounts within an organizational unit. The specified actions from an SCP affect all IAM users and roles, including the root user of the member account. For more information, see [Policy evaluation logic](#).

Analyzing access

[Close all](#)

permissions to grant only the permissions required with the goal of achieving [least-privilege permissions](#). AWS provides tools to help you refine your permissions. You can start with [AWS managed policies](#), which are created and administered by AWS and include permissions for common use cases. As you refine your permissions, define specific permissions in [customer managed policies](#). To help you determine the specific permissions you require, use [AWS Identity and Access Management \(IAM\) Access Analyzer](#), review AWS CloudTrail logs, and inspect last access information. You also can use the [IAM policy simulator](#) to test and troubleshoot policies.

What is IAM Access Analyzer?

Achieving least privilege is a continuous cycle to grant the right fine-grained permissions as your requirements evolve. IAM Access Analyzer helps you streamline permissions management in each step of this cycle. [Policy generation](#) with IAM Access Analyzer generates a fine-grained policy based on the access activity captured in your logs. This means that after you build and run an application, you can generate policies that grant only the required permissions to operate the application. [Policy validation](#) with IAM Access Analyzer uses more than 100 policy checks to guide you to author and validate secure and functional policies. You can use these checks while creating new policies or to validate existing policies. Custom policy checks are a paid feature to validate that developer-authored policies adhere to your specified security standards ahead of deployments. Custom policy checks use the power of automated reasoning—provable security assurance backed by mathematical proof—to enable security teams to proactively detect nonconformant updates to policies.

[Public and cross-account findings](#) with IAM Access Analyzer help you verify and refine access allowed by your resource policies from outside your AWS organization or account. For more information, see [Using IAM Access Analyzer](#). Unused access with IAM Access Analyzer continuously analyzes your accounts to identify unused access and creates a centralized dashboard with findings. The findings highlight unused roles, unused access keys for IAM users, and unused passwords for IAM users. For active IAM roles and users, the findings provide

How do I remove unused permissions?

You might have IAM users, roles, and permissions that you no longer require in your AWS account. We recommend that you remove them with the goal of achieving least-privilege access. For IAM users, you can review password and access key last used information. For roles, you can review role last used information. This information is available through the IAM