**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](https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies_managed-vs-inline.html#aws-managed-policies)
* [Customer Managed Policies](https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies_managed-vs-inline.html#customer-managed-policies)
* [Inline Policies](https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies_managed-vs-inline.html#inline-policies)
* [Choosing Between Managed Policies and Inline Policies](https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies_managed-vs-inline.html" \l "choosing-managed-or-inline)
* [Deprecated AWS Managed Policies](https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies_managed-deprecated.html)

## AWS Managed Policies

An AWS managed policy is a standalone policy that is created and administered by AWS. Standalone policymeans that the policy has its own [Amazon Resource Name (ARN)](http://docs.aws.amazon.com/general/latest/gr/aws-arns-and-namespaces.html" \t "_blank) that includes the policy name.

AWS managed policies are designed to provide permissions for many common use cases. There are AWS managed policies that define typical permissions for service administrators and grant full access to the service, such as [AmazonDynamoDBFullAccess](https://console.aws.amazon.com/iam/home" \l "policies/arn:aws:iam::aws:policy/AmazonDynamoDBFullAccess" \t "_blank) and [IAMFullAccess](https://console.aws.amazon.com/iam/home" \l "policies/arn:aws:iam::aws:policy/IAMFullAccess" \t "_blank). Some AWS managed policies are designed for power users, such as [AWSCodeCommitPowerUser](https://console.aws.amazon.com/iam/home" \l "policies/arn:aws:iam::aws:policy/AWSCodeCommitPowerUser" \t "_blank) and [AWSKeyManagementServicePowerUser](https://console.aws.amazon.com/iam/home" \l "policies/arn:aws:iam::aws:policy/AWSKeyManagementServicePowerUser" \t "_blank), which also depends on IAM permissions. Other AWS managed policies provide various levels of access to AWS services, such as [AmazonMobileAnalyticsWriteOnlyAccess](https://console.aws.amazon.com/iam/home" \l "policies/arn:aws:iam::aws:policy/AmazonMobileAnalyticsWriteOnlyAccess" \t "_blank)and [AmazonEC2ReadOnlyAccess](https://console.aws.amazon.com/iam/home" \l "policies/arn:aws:iam::aws:policy/AmazonEC2ReadOnlyAccess" \t "_blank), which provides access to additional related 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.

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](https://console.aws.amazon.com/iam/home" \l "policies/arn:aws:iam::aws:policy/AdministratorAccess" \t "_blank) 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](https://console.aws.amazon.com/iam/home" \l "policies/arn:aws:iam::aws:policy/PowerUserAccess" \t "_blank) job function, For a list and descriptions of the job function policies, see [AWS Managed Policies for Job Functions](https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies_job-functions.html).

You cannot change the permissions defined in AWS managed policies. AWS will occasionally update 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.


        Diagram of AWS managed policies
      

## 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)](http://docs.aws.amazon.com/general/latest/gr/aws-arns-and-namespaces.html" \t "_blank) 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.


        Diagram of customer managed policies
      

## Inline Policies

An inline policy is a policy that's embedded in a principal entity (a user, group, or role)—that is, the policy is an inherent part of the principal entity. You can create a policy and embed it in a principal entity, either when you create the principal entity 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.


        Diagram of inline policies
      

## 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 principal entity 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 [Versioning IAM Policies](https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies_managed-versioning.html). To learn more about the Version policy element see [IAM JSON Policy Elements: Version](https://docs.aws.amazon.com/IAM/latest/UserGuide/reference_policies_elements_version.html).

**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 admins—that is, admins that can create, update, and delete policies. You can then designate other users as limited admins. That is, admins 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](https://docs.aws.amazon.com/IAM/latest/UserGuide/access_controlling.html#access_controlling-policies).

**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 principal entity that it's applied to. For example, you want to be sure that the permissions in a policy are not inadvertently assigned to a principal entity 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 principal entity. In addition, when you use the AWS Management Console to delete that principal entity, the policies embedded in the principal entity are deleted as well. That's because they are part of the principal entity.