IDENTITY ACCESS MANAGEMENT(IAM)

Identity access management

Regardless of where employees are working, they need to access their organization's resources like apps, files, and data. The traditional way of doing things was to have the vast majority of workers work on-site, where company resources were kept behind a firewall. Once on-site and logged in, employees could access the things they needed.

Creating users and groups

Creating user

IAM users are like representatives for people or applications using AWS. They can access AWS through the console or programmatically. IAM users help manage access to AWS resources securely, and you can set permissions for them individually or in groups.

They have names and passwords for console access and can create access keys for programmatic access. IAM users ensure only authorized users and apps can access your AWS resources.

Creating groups

An IAM group is a collection of users who are grouped together based on the same access control policies. These policies define what actions the group members are allowed to perform on specific objects within the group's scope.

For example, let's say you have a group called "Developers." In the group's access control policy, you grant read-only access to all of your EC2 instances. Now, any user added to the "Developers" group will automatically have the permission to view information about those EC2 instances, but they won't have permission to make any changes to them.

Applying policies

You use policies to define the permissions for an identity (user, 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.

Types of policies

identity-based policies

resource-based policies

permissions boundaries

Organizations SCPs

ACLs

- s **Identity-based policies** – Attach managed and in line policies to IAM identities (users, groups to which users belong, or roles). Identity-based policies grant permissions to an identity.
- **Resource-based policies** – 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** – 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** – 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)** – 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** – 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..

Password policy

You can set a custom password policy on your AWS account to specify complexity requirements and mandatory rotation periods for your IAM users' passwords. If you don't set a custom password policy, IAM user passwords must meet the default AWS password policy.

Steps for creating password policy

When you configure a custom password policy for your account, you can specify the following conditions:

- Password minimum length – You can specify a minimum of 6 characters and a maximum of 128 characters.
- Password strength – You can select any of the following check boxes to define the strength of your IAM user passwords:
    - Require at least one uppercase letter from the Latin alphabet (A–Z)
    - Require at least one lowercase letter from the Latin alphabet (a–z)
    - Require at least one number
    - Require at least one nonalphanumeric character ! @ # $ % ^ & * ( ) _ + - = [ ] { } | '
- Turn on password expiration – You can select and specify a minimum of 1 and a maximum of 1,095 days that IAM user passwords are valid after they are set. For example, if you specify an expiration of 90 days, it immediately impacts all of your users. For users with passwords older than 90 days, when they log into the console after the change, they must set a new password. Users with passwords 75-89 days old receive an AWS Management Console warning about their password expiration. IAM users can change their password at any time if they have permission. When they set a new password, the expiration period for that password starts over. An IAM user can have only one valid password at a time.
- Password expiration requires administrator reset – Select this option to prevent IAM users from using the AWS Management Console to update their own passwords after the password expires. Before you select this option, confirm that your AWS account has more than one user with administrative permissions to reset IAM user passwords. Administrators with iam:UpdateLoginProfile permission can reset IAM user passwords. IAM users with iam:ChangePassword permission and active access keys can reset their own IAM user console password programmatically. If you clear this check box, IAM users with expired passwords must still set a new password before they can access the AWS Management Console.
- Allow users to change their own password – You can permit all IAM users in your account to change their own password. This gives users access to the iam:ChangePassword action for only their user and to the iam:GetAccountPasswordPolicy action. This option does not attach a permissions policy to each user. Rather, IAM applies the permissions at the account-level for all users. Alternatively, you can allow only some users to manage their own passwords. To do so, you clear this check box..

- Prevent password reuse – You can prevent IAM users from reusing a specified number of previous passwords. You can specify a minimum number of 1 and a maximum number of 24 previous passwords that can't be repeated.

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 update 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.