AWS IAM

AWS IAM (Identity and Access Management) is a service that helps you securely control access to AWS resources. It allows you to create and manage users, groups, roles, and policies to define who can access what.

Imagine your AWS account as a digital fortress. IAM acts as the gatekeeper, meticulously determining who (users, applications, services) can access what resources (S3 buckets, EC2 instances, etc.), when (temporary or long-term access), and under what conditions (specific actions allowed by permissions).

Key Components of AWS IAM

1. **IAM Users:** IAM allows you to create individual users with unique credentials, enabling them to access AWS resources securely. Each user can have specific permissions assigned to them based on their role within the organization.
2. **IAM Groups:** Groups in IAM simplify permission management by allowing you to organize users with similar access requirements. Instead of assigning permissions to individual users, you can assign them to groups, streamlining the process of access management.
3. **IAM Roles:** IAM roles are entities with permissions that can be assumed by users, applications, or services within your AWS environment. Roles provide temporary access to resources and are commonly used for cross-account access, federated access, and granting permissions to AWS services.
4. **IAM Policies:** IAM policies define the permissions that are granted or denied to users, groups, or roles. Policies are JSON documents that specify the actions users can perform and the resources they can access. By adhering to the principle of least privilege, you can ensure that users have only the permissions necessary to perform their tasks.

Benefits of AWS IAM

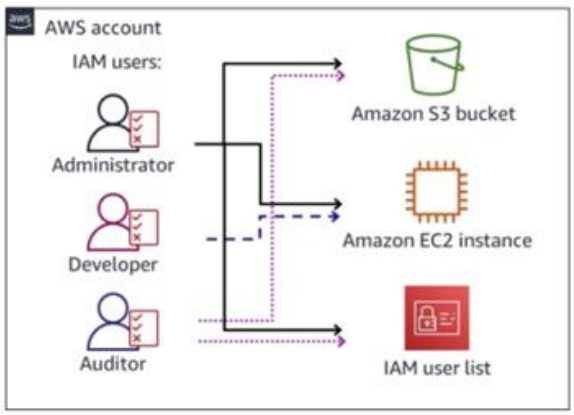
- **Granular Access Control:** IAM enables you to define fine-grained permissions, allowing you to grant only the necessary level of access to resources.
- **Security:** By following the principle of least privilege, IAM helps reduce the risk of unauthorized access and data breaches.
- **Flexibility:** IAM supports a wide range of use cases, from managing users and groups to granting temporary access through roles.

- **Compliance:** IAM features such as access logging and identity federation help organizations maintain compliance with regulatory requirements.

IAM Users

An IAM user represents a person or application that interacts with AWS. Each user has credentials such as passwords or access keys.

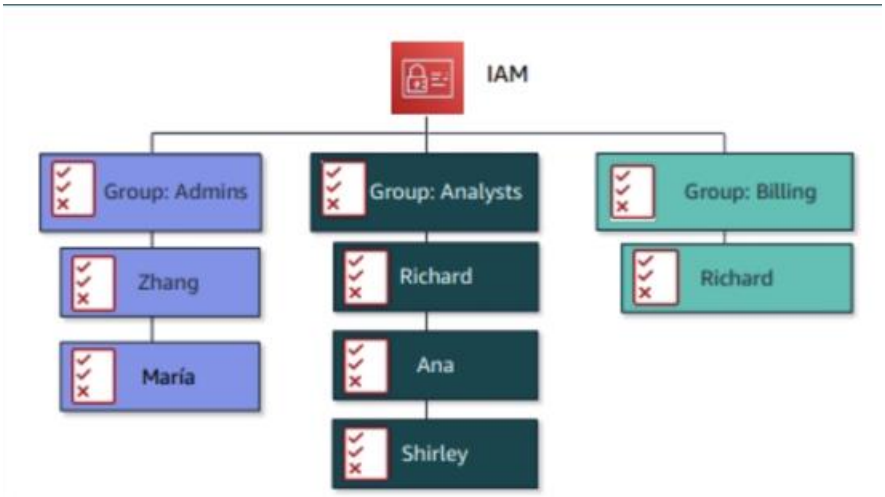
Eg: Imagine a DevOps engineer needs access to AWS to deploy applications. You create an IAM user for them with permissions to deploy resources.



IAM Groups

A group is a collection of IAM users that share the same permissions.

Eg: If you have a team of DevOps engineers, instead of assigning permissions individually, you can create a group DevOpsTeam and attach permissions to the group.



AWS IAM Policy

Policies define what actions are allowed or denied for an entity (user, group, or role). AWS provides managed policies, or you can create custom policies.

eg - You can attach this policy to a user or group:

Structure of IAM Policies

- **Version**: Specifies the version of the policy language.
- **Statement**: One or more individual statements, each containing:
- **Effect**: Whether the statement allows or denies access (Allow or Deny).
- **Action**: The specific actions covered by the statement (e.g., s3:GetObject).
- **Resource**: The resources the actions apply to (e.g., an S3 bucket ARN).
- **Condition**: Optional conditions that limit the actions (e.g., based on IP address).

WHY IAM Policies

- Use Least Privilege: Grant only necessary permissions.
- Leverage Managed Policies: Utilize AWS-managed policies whenever possible, as they are maintained by AWS and automatically updated to reflect best practices and new features. Managed policies help ensure consistency and compliance with security standards.

AWS IAM Roles

AWS IAM Roles are similar to users but are intended to be assumed by AWS services or applications.

eg - An EC2 instance needs access to an S3 bucket. Instead of storing credentials on the instance, you can create a role and attach it to the instance.

WHY IAM Roles

1. Grant access temporarily: IAM roles are ideal for situations in which access to services or resources needs to be granted temporarily, instead of long-term.
2. Cross-Account Access: Use roles to grant access between AWS accounts.
3. Avoid Long-Term Credentials: Utilize roles instead of long-term access keys. Roles provide temporary credentials that automatically rotate, reducing the risk of credential compromise.
4. IAM Roles for AWS Services: Assign roles to AWS services for dynamic permissions.
5. Role-Based Access Control (RBAC): Follow the least privilege principles when defining role permissions.