**IAM (Identity and Access Management) in AWS**

It is a service that help you securely manage user access to AWS resources. It allows you to define who (users) can access what (resources) and under what condition.

**Key Components of IAM**

1. Users
2. Groups
3. Roles
4. Policies

**Users**

Individual identities that need access to AWS services. User could be people or application that interact with AWS resources. Each user has a unique name, access credential (password, access key).

**Ex**= Users can have different permissions, such as full access or read-only access, based on attached policies.

**Groups**

An IAM Group is a collection of users that share the same permissions. Instead of assigning permissions to each user individually, you can attach a policy to the group, and all users in that group will inherit those permissions.

**Example:** Suppose you have a team of developers. Instead of giving each developer access separately, you create a "Developers" group and attach a policy that allows them to access EC2 and S3. Now, whenever a new developer joins, you just add them to the "Developers" group, and they automatically get the required permissions.

**Roles:**

An **IAM Role** is like a temporary identity that AWS services or users can "assume" to access resources **without using passwords or access keys.** Unlike IAM users (who have long-term credentials), **roles do not belong to any single person or service**—instead, they can be used by anyone or anything that is **allowed** to assume them

**Example:** Imagine an EC2 instance needs to access an S3 bucket. Instead of creating a user and managing access keys, you create an **"S3AccessRole"** and attach a policy that allows S3 access. Then, you assign this role to the EC2 instance. Now, the instance can access S3 **without needing a username or password!**

**Policies:**

**A policy in IAM is a set of permissions that define what actions are allowed or denied on AWS resources. Policies are written in JSON format and can be attached to users, groups, or roles to control access.**

**Example: If you want a user to only read objects from an S3 bucket but not delete them, you create a read-only policy and attach it to the user. This ensures they can view files but not remove them.**

**Types of IAM Policies**

* + **Managed Policies:** Predefined policies provided by AWS or created by administrators.
  + **Inline Policies:** Policies attached directly to a user, group, or role (that can be created or manages by user).