Himanshu Pal
A045
86062300021

AWS Identity and Access Management (IAM) is a powerful service that helps control access to AWS resources securely. Through IAM, administrators can manage permissions to ensure that users, groups, and roles can only access specific resources under certain conditions. Here's a breakdown of key components:

1) **IAM Users and Groups**

- **IAM Users**: IAM users represent individual accounts within AWS. Each IAM user has a unique identity within an AWS account, allowing them to access specific resources. Users typically receive unique credentials, which can include a username, password, and programmatic access keys, allowing access to resources either through the AWS Management Console or APIs. Permissions are assigned to users via policies, defining what actions and resources they can access.
- **IAM Groups**: Groups are collections of users with a common set of permissions. By creating groups (like "Admin," "Developers," or "Support"), administrators can grant or revoke permissions to multiple users simultaneously, making management easier and more efficient. For example, if you have several users who need read-only access to S3 buckets, adding them to a "Read-Only" group with specific policies is simpler than assigning policies individually.

2) Identity and Access Management (IAM)

IAM is a core service for securely managing access to AWS resources. With IAM, administrators can create and manage **users**, **groups**, **roles**, and **policies** to control access.

- **IAM Users**: Represent individual accounts, each with unique credentials. Users have no default permissions, so access must be granted via policies.
- **IAM Groups**: Collections of users with shared permissions. Attaching policies to a group applies the permissions to all members, simplifying management.
- **IAM Policies**: JSON documents that define permissions. They control who can do what, including conditions like IP restrictions or requiring MFA.
- **IAM Roles**: Temporary identities that services or users can assume, typically without permanent credentials. Roles are used for secure, temporary access by AWS services (e.g., EC2 accessing S3) or for cross-account access.

IAM enables **granular control**, enhancing security by following the **least privilege** principle. It's essential for managing access across AWS resources efficiently and securely.

3) IAM Roles

- **IAM Roles**: Roles are like users but without long-term credentials. They are temporary identities with specific permissions. Roles are typically assumed by trusted entities, like other AWS services (e.g., EC2 instances, Lambda functions), or even external accounts, allowing access to resources without sharing long-term credentials. Each role has a defined trust policy that specifies which entities can assume the role and a permissions policy defining the actions the role can perform. For example, an EC2 instance role might have permissions to access specific S3 buckets without the need to hard-code credentials in the application code running on the instance.
- **Cross-Account Access**: Roles can also be used for cross-account access, where entities from one AWS account are permitted to access resources in another. This is crucial for environments where multiple accounts are used for different teams or projects but need to share resources securely.

In summary, **AWS IAM** enables centralized access control, using users, groups, policies, and roles to create secure and efficient environments. By carefully structuring IAM permissions, AWS users can safeguard resources while providing necessary access to individuals, groups, and applications across an organization.