**AWS IAM**

AWS IAM (Identity and Access Management) is a service provided by Amazon Web Services (AWS) that enables you to manage access to your AWS resources securely. It allows you to control ***who can do what*** within your AWS account and resources. IAM is crucial for maintaining the principle of least privilege, where users and applications are granted only the minimum permissions required to perform their tasks.

**Key features of AWS IAM include:**

**1. Users:** IAM allows you to create individual user accounts for people or services that require access to your AWS resources. Each user is assigned unique credentials (username and password) or access keys (for programmatic access) to sign in.

**2. Groups:** You can group users based on their job functions. By assigning permissions to groups, you can manage access more efficiently and apply common permissions to multiple users.

**3. Roles:** IAM roles are like users, but they are not associated with a specific person. Instead, roles are assumed by AWS services or entities like EC2 instances, allowing them to temporarily take on certain permissions. Roles are often used for granting permissions to AWS services like Lambda, EC2, etc.

**4. Policies:** IAM policies are JSON documents that define permissions. You can attach policies to users, groups, or roles to specify what actions they are allowed or denied on specific AWS resources. AWS provides managed policies, and you can also create custom policies to meet specific requirements.

**5. Multi-Factor Authentication (MFA):** IAM supports enabling MFA for extra security. With MFA, users must provide a second form of authentication (e.g., a one-time password from a virtual MFA device or a hardware token) in addition to their password.

**Note:**

* An AWS Identity and Access Management (IAM) user is an entity that you create in AWS to represent the person or application that uses it to interact with AWS Services.
* AWS Identity and Access Management (IAM) is a web service that helps you securely control access to AWS resources.
* We can use IAM to control who is authenticated (signed in) and authorized (has permissions) to use resources.
* IAM helps protect against security breaches by allowing administrators to automate numerous user account related tasks.
* Best practice of using the root user only to create your first IAM user.
* Enable Multi Factory Authentication (MFA) for Root User.
* By using Google Authenticator App, we can configure "Virtual MFA".

**Best Practices:**

- When we login AWS using 'email' and 'password', that has complete access to all AWS services and resources in the account (Root account).

- Strongly recommended that you do not use the "root user" for your everyday tasks, even the administrative ones.

- Instead, adhere to the best practice of using the root user only to create your first IAM user. Then securely lock away the root user credentials and use them to perform only a few account and service management tasks.

- IAM user is truly global, i.e., once IAM user is created it can be accessible in all the regions in AWS.

- Amazon S3 is also considered as Global but, it is not truly global. When we create a bucket in S3 it displays all the buckets of other regions in one place , so that is the reason we are calling AmazonS3 is Global (but partly global).

- But IAM is 100% Global. Once you create IAM user you can use it anywhere in all the regions.

**Steps:**

1. Create an IAM user

Services - Security, Identity, & Compliance - IAM

Users---<Add user>

Username\* = Iamuser1

Access type = 'select' both "Programmatic Access"

"AWS Management Console access"

Console password = 'select'

custom Password = (\*\*\*\*\*\*\*\*somepassword eg:test1234)

click <NextPermissions>

(Note: we are not providing any permissions as of now, just <create user>)

Once the IAM user has been created.

AccessKeyID =AKIAIEJH7Z3FDKH36YWQ

Secretaccesskey=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

**Note:** Once you close this window, AccessKeyID and Secret Accesskey has gone, so save it somewhere.

- Best Practice is never giving an individual permission to the user, as users will be changed frequently, when they left the organization. So, need to create the Groups and assign the users to it.

2. Group

<create new group>

Groupname =admins

(Note: no need add any policy now).

<creategroup>

3. Add user to this group

click on newly create group 'admins'.

<Add users to Group>

GroupARN =arn:aws:iam::540105522204:group/admins

-Always add the permissions to the 'Groups' level not to the 'users' level. It’s a Best Practice in real-time.