IDENTITY AND ACCESS MANAGEMENT (IAM)

IAM refers to a framework or policy and technologies for ensuring that the proper people in an organization have the appropriate access to technology resources.

OR

AWS Identity and Access Management is a web service that you security control access to AWS resources. We use IAM to control who is authenticated (signed-in) and authorized (has permission) to use resources.

- When you first create AWS account, you begin in a single sign-in identity that has completely access to all AWS services and resources in the account.
- This identity is called the AWS account "Root-User" and is accessed by sighed-in with the email address and password that you used to create the account.
- AWS strongly recommends that you do not use the root user for you everyday tasks, even the administrative ones.
- Use other IAM user account to manage the administrative task of your account and securely lock away the root user credentials and use them to perform only a few account and service management task.
- IAM user limit is 5000 per AWS account. You can add up to 10 users at one time.
- You are also limit to 300 groups per AWS account.
- Default limits of managed policies attached to an IAM role and IAM user is 10.
- IAM user can be a member of maximum 10 groups.
- We can assign maximum two access keys to an IAM user.

IAM Features:

1. **Shared access to your AWS account:**

   You can grant other people permission to administer and use resources in your AWS account without having to share your access credentials.

2. **Granular permission:**

   - You can grant different permission to different people for different resources.
   - For instance, you can allow some users complete access to EC2, S3, Dynamo DB, Redshift while for others, you can allow read only access to just some S3 buckets, or permission to administer just some EC2 instances or to access your billing information but nothing else.

3. **Secure access to AWS resources for application that run on Amazon EC2:**

   You can use IAM features to securely give application that run on EC2 instances the credentials that they need in order to access other AWS resources. For example, include S3 buckets and RDS or Dynamo DB databases.

4. **Multifactor Authentication (MFA):**

   You can add two factor authentication to your account and to individual users for extra security. You can use physical hardware or virtual MFA (for e.g: Google Authenticator)

5. **Identity federation:**

   You can allow users who already have passwords elsewhere. For e.g: in your corporate network of with an internet identity provider to get temporary access to your AWS account.

6. **Identity information for assurance:**

   If you use AWS Cloud Trail, you receive log records that include information about those who made request for resources in your account. That information is based on IAM Identities.

7. **PCI-DSS compliance:**

   IAM supports the processing, storage and transmission of credit cards by a merchant of service provider, and has been validated as being complaint with payment card industries (PCI) data security standards (DSS).

8. **Eventually consistent:**

   - If a request to change some data is successful, the change is committed and safely stored. However the change must be replicated across IAM which can take some time.
   - IAM achieves high availability by replicating data across multiple servers within AWS data centre around the world.

Fee to Use: AWS IAM is a feature of AWS account offered at no additional charge. You will be charged only for use of other AWS products by your IAM users.

IAM Terms:

Following are the major terms which are used in an IAM account.

1. Principal
2. Request
3. Authentication
4. Authorization
5. Action/Operation
6. Resources

1. Principal:

- A principal is a person or application that can make a result for an action or operation on an AWS resources.
- Your administrative IAM user is your first principal.
- You can allow users and services to assume a role.
- IAM users, roles, federated users and application are all AWS principals.
- You can support federated users of programmatic access to allow an application to access your AWS account.

2. Request:

When a principal tries to use the AWS management console, the AWS API of the AWS CLI that principal sends a request to AWS. The request includes the following information:

- Actions: That the principal wants to perform.
- Resources: upon which the actions are performed.
- Principal information: it's including the environment from which the request was made.
  Request context: before AWS can evaluate and authorize a request, AWS gathers the request information. Principal (the requester) which is determined based on the authorization data. This includes the aggregate permissions that the associated with that principal.
- Environment data: such as IP address, user agent, SSL enabled status, or the time of the day.
- Resource data: it is related to the resource that is being requested.

3. Authentication:

- A principal sending a request must be authenticated (sighed into AWS) to send a request to AWS.
- Some AWS services, like AWS S3 allow request from anonymous users, they are exception to the role.

- To authenticate from the console as a root user, you must sign-in with your user name and password.
- To authenticate from the API to CLI, you must provide your access key and secrete key.
- You might also be required to provide additional security information like MFA (e.g: Google Authentication)

4. Authorization:

- To authorize request, IAM uses value from the request context to check, for matching policies and determine whether to allow or deny the request.
- IAM policies are stored in IAM as JSON documents and specify the permission that are allowed or denied.
- **User (identity) Based Policy** specifies permission allowed/denied for principals.
- **Note:** by default the AWS root user access to all the resources in that account.

- **Resource Based Policies:**

  - It specifies permission allowed/denied for resources. Popular for granting cross account permission.
  - IAM checks each policy that matches the context of your request.
  - If a single policy includes a denied actions, IAM denies the entire request and stop evaluating. This is called **explicit deny**.

  The evaluation logic follows these Rules:

  - By default, all request are denied.
  - An explicit allow overrides this default.
  - An explicit deny overrides any allows.

- You can create a new IAM policy in the AWS management console using one of the following ways:
  - JSON: you can create your own JSON syntax.
  - Visual Editor: you can construct a new policy from scratch in the visual editor. If you can use the visual editor you do not have to understand JSON syntax.
  - Import: you can import a managed policy with in your account and then edit the policy to customize it to your specific requirement.

5. Actions:

- Actions are defined by a service, and more the things that you can do to a resource such as viewing, creating, editing, and deleting that resource.
- IAM supports approx. 40 actions for a user resource including create user, delete user etc.

- Any actions or resources that are not explicitly allowed are denied by default.
- After your request has been authenticated and authorized, AWS approves the actions in your request.

6. **Resource:**

- A resource is an entity that exists within a service.
- Examples are EC2 instances, S3 bucket, IAM users, and Dynamo DB table.
- Each AWS service defines a set of actions that can be performed on each resource.
- After AWS approves the actions in your request those actions can be performed on the related resources within your account.
- If you create a request to perform an unrelated action on a resource that request is denied.
- When you provide permissions using an identity based policy in IAM then you provide permissions to access resources only within the same account.

Identity Federation:

- If your account users already have a way to be authenticated such as authentication through your corporate network, you can federated those user identities into AWS.
- A user who has already logged to the corporate using their corporate identity, the corporate can replace their existing identity with a temporary identity in your AWS account.
- The user can work in the AWS management console.
- Similarly, an application that the user is working with can make programmatic request using permission that you make.

Federation is particularly useful in those cases:-

1. **If your corporate direct ory is compatible with Security Assertion Markup Language (2.0):**

- You can configure your corporate directory to provide Single Sign-On (SSO) access to the AWS management console for your users.
- If your corporate directory is not compatible with SAML 2.0, you can create identity broker application to provide single sign-on access to the AWS management console for your users.
- If your corporate directory is Microsoft active directory, you can use AWS directory service to establish trust between your corporate directory and your AWS account.

2. **Your users already have Internet Identities:**

    - if you are creating a mobile app or web-based app that can let users identity themselves through an internet identity provider like login with amazon, facebook, google or any open ID connect (OIDC) compatible identity provider, the app can use web federation to access AWS.
    - AWS recommends to use AWS Cognito for identity federation.

IAM Users and SSO:

- IAM users in your account have access only to the AWS resources that you specify in the policy that is attached to the user of to an IAM group that use the user belongs to.
- To work in the console user must have permissions to perform the actions that the console performs such as listing and creating AWS resources.

IAM Identities:

- IAM identities is what you create under your AWS account to provide authentication for people, application and process in your AWS account.
- Identities represents the user and can be authenticated and then authorized to perform actions in AWS.
- Each of these can be associated with one or more policies to determine what actions a user, role or member of the group can do with which resources and under what conditions.
- IAM group is a collection of IAM user.
- IAM role is very limit IAM user.

A. IAM Users:

- An IAM user is an entity that you create in AWS. It represents the person or service who uses the IAM user to interact with AWS.
- You can create 5 users at time.
- An IAM user can represent an actual person or an application that requires AWS access to perform actions on AWS resources.
- A primary use of IAM users is to give people the ability to sign-in to the AWS management console for interactive task and to make programmatic request to AWS services using the API or CLI.
- For any user you can assign them:
    - A username and password to access the AWS console.
    - An access key ID and secrete key that can use for programmatic access.
    - The newly created IAM user have no password and no access key. You need to create the user password.
    - Each IAM user is associated with one and only one AWS account.

- Users are defined within your account, so user do not have to do payment. Bill would be pay by the parent account.

B. IAM Groups:

- An IAM group is a collection of IAM users.
- It is a way to assign permission/policies to multiple users at once.
- Use groups to specify permissions for a collection of users, which can make those permissions easier to manage for those users.
- For E.g: you could have a group called HR and give that group the types of permissions that HR department typically needs.
- Any user in that group automatically has the permission that are assigned to the group.
- If a new user joins your organization and should have administrator privileges, you can assign the appropriate permissions by adding the user to that group.
- If a person changes job in your organization, instead of editing that user's permission, you can remove him or her from the old groups and add him or her to the appropriate new groups.

IAM Group Limitations:

- A group is not truly an identity in IAM because it cannot be identified as a principal in a permission policy.
- Group cannot be nested.
- One have a limit of 300 groups in an AWS account.
- A user can be a member of up to 10 IAM groups.

C. IAM Roles:

- An IAM role is very similar to a user, in that it is an identity with permission policies that determine what the identity can and cannot do in AWS.
- An IAM role does not have any credentials (password or access key) associated with it.
- Instead of being associated with one person, a role is intended to be assumable by anyone who needs it.
- An IAM user can assume a role to temporarily take on different permissions for a specific task.
- An IAM role can be assigned to a federated user who sign-in by using an external identity provider instead of IAM.

IAM Temporary Credentials:

- Temporary credential are primary used with IAM roles but there are also other uses.
- You can request temporary credentials that have a more restricted set of permissions than your standard IAM users.
- This prevent you from accidentally performing task that are permitted by the more restricted credentials.
- A benefit of temporary credentials in that they expire automatically after a set period of time.

Permissions and Policies:

- The access management portion of AWS Identity and Access Management (IAM) helps you to define what a user or other entity is allowed to do in an account, often referred to as authorization.
- Permissions are granted through policies that are created then attached to user, groups of roles.

Policies and User:

- By default, IAM users can't access anything in your account.
- You grant permissions to a user by creating a policy, which is a document that defines the effect, actions, resource and optional conditions.
- Any actions or resources that are not explicitly allowed are denied by default.

IAM Multiple Policies:

- Users of groups can have multiple policies attached to them that grant different permission.
- In the case of multiple policies attached to a user or group, the user's permission are calculated based on the combination of policies.

Federated Users and Roles:

- Federated user don't have permanent identities in your account the way that IAM users do.
- To assign permissions to federated users you can create an entity referred to as a role and define permission for the role.
- When a federated user sign-in to AWS the users is associated with the role and is granted the permission that are defined in the role.

Resource based Policy:

- In some cases like S3 bucket, you can attach a policy to a resource in addition to attaching it to a group or user. This is called a resource based policy.
- A resource based policy contains slightly different information than user-based policy.
- In resource based policy you specify what actions are permitted and what resource is affected.
- You also explicitly list who is allowed access to the resource (a principal).
- Resource based policies include a principal element that specifies who is granted the permissions.

IAM User-The Root User:

- When you first create an AWS account, you create an account (or root user) identity, which you use to sign-in to AWS.
- The account root user credentials are the e-mail address to create the account and a password which can be used to sign-in to the AWS Management console as the root user.
- When you sign-in as root user, you have complete unrestricted access to all resources in your account including access to your billing information and the ability to change your password.
- The level of access is necessary when you initially set up the account.
- It is not possible to restrict the permission that are granted to the AWS account.

AWS Recommends That:

- AWS recommends that you don't use root user credentials for everyday access.
- Also AWS recommends that you do not share your root user credentials with anyone because doing so gives them unrestricted access to your account.
- Create an IAM user for yourself and then assign yourself administrative permission for your account.
- You can then sign-in as that user to add more users as needed.
- An IAM user with administrator permissions is not the same things as the AWS account root users.

IAM Users:

- An IAM user is an entity that you create in AWS. It represents the person or service who uses the IAM user to interact with AWS.
- An IAM can represent an actual person or an application that requires AWS access to perform action on AWS resources.
- IAM users are global entities, like an AWS account is today. No region is required to be specified when you define user permissions. Users can use AWS services in any geographic region.

For Any User You can assign them:

- A user name and password to access the AWS console.
- An access key (access key and secrete key) that they can use for programmatic access (issuing request) to your AWS account.
- You assign either or both based on the user activities and needs.
- You can view and download your secret access key only when you create the access key.
- You cannot view or recover a secret access key later.
- If you lose your secrete access key, you can create a new access key.
- Each IAM user is associated with one AWS account.

By Default a new IAM User:

- A new IAM user has no permission to do anything.
- Has no password and no access key (neither an access key ID nor a secret access key). It means no credentials of any kind.
- You must create the type of credentials for an IAM users based on what the user will be doing.
- You can grant user permissions by attaching IAM policies to them directly or making them members of IAM group where they inherit the group policies/permissions.
- You can have up to 5000 user per account.

IAM Roles:

- An IAM role is a set of permissions that grant access to the actions and resources in AWS.
- These permissions area attached to the role, not to an IAM user or group. Instead of being uniquely associated with one person, a role is intended to be assumable by anyone who needs it.
- A role does not have standard long-term credentials (password or access key) associated with it.
- If a user assumes a role, temporary security credentials are created dynamically and provided to the user.
- Following entities can used role:
    - An IAM user in the same AWS account.
    - An IAM user in a different AWS account.
    - A webserver offered by AWS such as Amazon EC2.

There are Two ways to use a Role:

1. **Internally in the IAM Console:**

   - IAM users in your account using the IAM console can switch to a role to temporarily use the permissions of the role in the console.
   - The user give up their original permission and take on the permission assigned to the role.
   - When the user exists the role, their original permissions are restored.

2. **Programmatically with the AWS CLI, tools for windows powershell or API:**

   - An application or a service offered by AWS (like Amazon EC2) can assume a role by requesting temporary security credentials for a role with which to make programmatic request to AWS.
   - You use a role this way so that you don't have to share or maintain long-term security credentials for each entity that requires access to a resource.

Difference between IAM Role and Resource Based Policy:

- Unlike a user-based policy, a resource based policy specifies who can access that resource.
- Cross account access with a resource based policy has an advantage over a role, with a resource that is accessed through a resource-based policy, the user still works in the trusted account and does not have to give up this or her user permissions in place of the role permissions.
- In other words, the user continuous to have access to resources in the trusted account at the same time as he or she has access to the resource in the trusting account.
- This is useful for task such as copying information to or form the shared resource in the other account.
- Note that not all services support resource-based policy.

IAM Role Delegation:

- Delegation is the granting of permission to someone to allow access to resource that you control.
- Delegation involves setting up a trust between the account that owns the resource (the trusting account) and the account that contains the users that need to access the resource (the trusted account).
- The trusted and trusting accounts can be of the following:
  i. The same account
  ii. Two accounts that are both under your organization's control.
  iii. Two account owned by different organizations.

- To delegate permission to access a resource you create an IAM role that has two policies attached.
  i. The Trust Policy
  ii. The Permission Policy
- The trusted entity is included in the policy as the principal element in the document.
- When you create a trust policy, you cannot specify a wildcard (*) as a principal.

Cross Account Permissions:

- You might need to allow user from another AWS account to access resources in your AWS account. If so, don't share security credentials, such as access keys between accounts. Instead use IAM roles.
- You can define a role in the trusted account that specifies what permissions the IAM users in the other account are allowed.
- You can also designate which AWS account have the IAM users that are allowed to assume the role. We do not define users here rather AWS account.

Role for Cross-Account Access:

- Granting access to resources in one account to a trusted principal in a different account.
- Roles are the primary way to grant cross-account access.
- However with some of the web services offered by AWS, you can attach a policy directly to a resource. These are called resource-based policy. You can use them to grant principals in another AWS account access to the resource.
- The following services support resource-based policy:
  - Amazon S3.
  - Amazon Simple Notification Service
  - Amazon Simple Queue Service
  - Amazon Glacier Vault