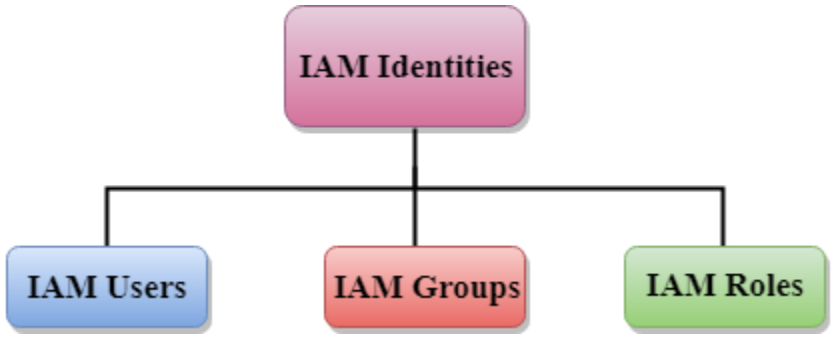
IAM Identities

IAM identities are created to provide authentication for people and processes in your aws account.

IAM identities are categorized as given below:



- IAM Users
- IAM Groups
- IAM Roles

AWS Account Root User

- When you first create an AWS account, you create an account as a root user identity which is used to sign in to AWS.
- You can sign to the AWS Management Console by entering your email address and password. The combination of email address and password is known as **root user credentials**.
- When you sign in to AWS account as a root user, you have unrestricted access to all the resources in AWS account.

- The Root user can also access the billing information as well as can change the password also.

What is a Role?

- A role is a set of permissions that grant access to actions and resources in AWS. These permissions are attached to the role, not to an IAM User or a group.
- An IAM User can use a role in the same AWS account or a different account.
- An IAM User is similar to an IAM User; role is also an AWS identity with permission policies that determine what the identity can and cannot do in AWS.
- A role is not uniquely associated with a single person; it can be used by anyone who needs it.
- A role does not have long term security credential, i.e., password or security key. Instead, if the user uses a role, temporarily security credentials are created and provided to the user.
- You can use the roles to delegate access to users, applications or services that generally do not have access to your AWS resources.

Situations in which "IAM Roles" can be used:

- Sometimes you want to grant the users to access the AWS resources in your AWS account.
- Sometimes you want to grant the users to access the AWS resources in another AWS account.
- It also allows the mobile app to access the AWS resources, but not want to store the keys in the app.
- It can be used to grant access to the AWS resources which have identities outside of AWS.
- It can also be used to grant access to the AWS resources to the third party so that they can perform an audit on AWS resources.

Following are the important terms associated with the "IAM Roles":

- **Delegation:** Delegation is a process of granting the permissions to the user to allow the access to the AWS resources that you control. Delegation sets up the trust between a trusted account (an account that owns the resource) and a trusting account (an account

that contains the users that need to access the resources).

The trusting and trusted account can be of three types:

- Same account
- Two different accounts under the same organization control
- Two different accounts owned by different organizations.

To delegate permission to access the resources, an IAM role is to be created in the trusting account that has the two policies attached.

Permission Policy: It grants the user with a role the needed permissions to carry out the intended tasks.

Trust Policy: It specifies which trusted account members can use the role.

- **Federation:** Federation is a process of creating the trust relationship between the external service provider and AWS. For example, Facebook allows the user to login to different websites by using their facebook accounts.
- **Trust policy:** A document was written in JSON format to define who is allowed to use the role. This document is written based on the rules of the IAM Policy Language.
- **Permissions policy:** A document written in JSON format to define the actions and resources that the role can use. This document is based on the rules of the IAM Policy Language.
- **Permissions boundary:** It is an advanced feature of AWS in which you can limit the maximum permissions that the role can have. The permission boundaries can be applied to IAM User or IAM role but cannot be applied to the service-linked role.
- **Principal:** A principal can be AWS root account user, an IAM User, or a role. The permissions that can be granted in one of the two ways:
  - Attach a permission policy to a role.
  - The services that support resource-based policies, you can identify the principal in the principal element of policy attached to the resource.
- **Cross-account access: Roles vs Resource-Based Policies:** It allows you to grant access to the resources in one account to the trusted principal in another account is known as cross-account access. Some services allow you to attach the policy directly, known as

Resource-Based policy. The services that support Resource-Based Policy are Amazon S3 buckets, Amazon SNS, Amazon SQS Queues.

IAM Roles Use Cases

There are two ways to use the roles:

- **IAM Console:** When IAM Users working in the IAM Console and want to use the role, then they access the permissions of the role temporarily. An IAM Users give up their original permissions and take the permissions of the role. When IAM User exits the role, their original permissions are restored.
- **Programmatic Access:** An AWS service such as Amazon EC2 instance can use role by requesting temporary security credentials using the programmatic requests to AWS.

An IAM Role can be used in the following ways:

- **IAM User:** IAM Roles are used to grant the permissions to your IAM Users to access AWS resources within your own or different account. An IAM User can use the permissions attached to the role using the IAM Console. A Role also prevents the accidental access to the sensitive AWS resources.
- **Applications and Services:** You can grant the access of permissions attached with a role to applications and services by calling the AssumeRole API function. The AssumeRole function returns a temporary security credentials associated with a role. An application and services can only take those actions which are permitted by the role. An application cannot exit the role in the way the IAM User in Console does, rather it stops using with the temporary credentials and resumes its original credentials.
- **Federated Users:** Federated Users can sign in using the temporary credentials provided by an identity provider. AWS provides an IDP (identity provider) and temporary credentials associated with the role to the user. The credentials grant the access of permissions to the user.

Following are the cases of Roles:

- Switch to a role as an IAM User in one AWS account to access resources in another account that you own.
    - You can grant the permission to your IAM Users to switch roles within your AWS account or different account. For example, you have Amazon EC2 instances which are very critical to your organization. Instead of directly granting permission to users to terminate the instances, you can create a role with the privileges that allows the administrators to switch to the role when they need to terminate the instance.
    - You have to grant users permission to assume the role explicitly.
    - Multi-factor authentication role can be added to the role so that only users who sign in with the MFA can use the role.
    - Roles prevent accidental changes to the sensitive resource, especially if you combine them with the auditing so that the roles can only be used when needed.
    - An IAM User in one account can switch to the role in a same or different account. With roles, a user can access the resources permitted by the role. When user switch to the role, then their original permissions are taken away. If a user exits the role, their original permissions are restored.
- Providing access to an AWS service
    - AWS services use roles to access a AWS resources.
    - Each service is different in how it uses roles and how the roles are assigned to the service.
    - Suppose an AWS service such as Amazon EC2 instance that runs your application, wants to make request to the AWS resources such as Amazon S3 bucket, the service must have security credentials to access the resources. If you embed security credentials directly into the instance, then distributing the credentials to the multiple instances create a security risk. To overcome such problems, you can create a role which is assigned to the Amazon EC2 instance that grants the permission to access the resources.
- Providing access to externally authenticated users. Sometimes users have identities outside of AWS such as in your corporate directory. If such users want to work with the AWS resources, then they should

know the security credentials. In such situations, we can use a role to specify the permissions for a third-party identity provider (IDP).

- **SAML -based federation**
  SAML 2.0 (Security Assertion Markup Language 2.0) is an open framework that many identity providers use. SAML provides the user with the federated single-sign-on to the AWS Management Console, so that user can log in to the AWS Management Console.
  How SAML-based federation works
- **Web-identity federation**
  Suppose you are creating a mobile app that accesses AWS resources such as a game that run on a mobile device, but the information is stored using Amazon S3 and DynamoDB.
  When you create such an app, you need to make requests to the AWS services that must be signed with an AWS access key. However, it is recommended not to use long-term AWS credentials, not even in an encrypted form. An Application must request for the temporary security credentials which are dynamically created when needed by using web-identity federation. These temporary security credentials will map to a role that has the permissions needed for the app to perform a task.
  With web-identity federation, users do not require any custom sign-in code or user identities. A User can log in using the external identity provider such as login with Amazon, Facebook, Google or another OpenID. After login, the user gets the authentication token, and they exchange the authentication token for receiving the temporary security credentials.

- **Providing access to third parties**
  When third parties want to access the AWS resources, then you can use roles to delegate access to them. IAM roles grant these third parties to access the AWS resources without sharing any security credentials.
  Third parties provide the following information to create a role:
  - The third party provides the account ID that contains the IAM Users to use your role. You need to specify AWS account ID as the principal when you define the trust policy for the role.

- The external ID of the third party is used to associate with the role. You specify the external ID to define the trust policy of the role.
- The permissions are used by the third party to access the AWS resources. The permissions are associated with the role made when you define the trust policy. The policy defines the actions what they can take and what resources they can use.