**IAM:-**

**IAM:-uses:-Securely control access to IAM Resources,**

\*with IAM we can manage to control AWS Users

EX:when you create AWS account it will ask to sign in

**Three ways to login AWS account:-**

**CLI,SDK,MFA**

**When will we USE IAM?**

:If we want to deal with multiple users and You want to restrict their access the users up to only some level then we will attach some policies to restrict the users

**Role:-**

A role is used to grant permissions to perform specific tasks, either temporarily or across different services,

**why to use role?**

we use role to access certain permissions to users

**when to use role?**

Roles are ideal when you want to grant temporary permissions for a specific task or period

**when you dont use role?**

For long-term, fixed permissions: If a user or application needs persistent access to resources, it's better to create an IAM user with long-term credentials rather than relying on roles

**where to use role?**

When services (like EC2, Lambda, S3, etc.) need to interact with other AWS resources, roles are used to provide them with the necessary permissions without using long-term credentials

**to whom we can create role?**

AWs service,account,webidentity,custom trust policy

**How to create role?**

1)Go to roles

2)create role

3)which permission policy you wan to attach 4)role name user defined create

**user:-who user IAM user to interact with resources**

user is used to access root user instances up to only some resources that resources also will be kept by root user.

Permissions:-uisers or gropoups assign to JSON documents are policies

USE LEAST PRIVILIGE PRINCIPLE

**how to create user ?**

1)**user name**

2)uncheck permission to access console

3) password

4)create group 5)attach policies 7)create user

**USERGROUPS:-**Groups are collection of multiple users

**3)IAM POLICY STRUCTURE?**

**Version**:-we have version which is by default 2012-10-17

**Statement:**the entire body will come under srtatement.

Statement must be start and end with square brackets

**SID:-some name**

**Effect**:effect must be in two ways allow or deny

**Allow:**he action is permitted

**Action:**which permission to allow we give in action

**Resources** :start with arn:aws:Ec2:avalaibility zone:”account no”:instance/iinstanc id

Give resources of which account and which instance we should give permissions

**IAM security tools:-**

**IAM credentials report:-**

A report that lists all accounts users..

**IAM Access Advisor:**

Permission given to users where last accessed

**IDENTITY Center or SSO:-** AWS IAM Identity Centre provides an identity federation solution that enables organizations to centrally manage user identities, authentication, and authorization without needing to maintain separate AWS-specific IAM user accounts

In simple words if we have multiple users then we cannot access at a time we should login or logout and another user etc for this the identity provider will give one sign in access so we cannot access to all

**Services**:EC2,S3,RDS,API gateway etc…

**Steps to create identity center**

1)**Enable identity center** 2)**choose identity source**(where users and groups are managed by default it will be root) 3)**create** predefined or custom **permission set** (in that how many hours we should control we should specify) 4)**set up account for access identity center** ( that is pick created users and groups in it)5)**sign in to it**

**SAML (Security Assertion Markup Language)** is an open standard used to exchange authentication and authorization data between parties, particularly between an Identity Provider (IdP) and a Service Provider (SP). In AWS, SAML is used to provide Single Sign-On (SSO) for users from external identity systems (like Active Directory, Okta, or others) to access AWS resources.

**OpenID Connect (OIDC) is an authentication layer on t**op of OAuth 2.0, allowing users to authenticate using an identity provider (IdP) and receive an identity token. AWS can integrate with external identity providers that support OpenID Connect to enable Single Sign-On (SSO) and federated access to AWS services.

**Web Identity refers to** using an external service (like Google, Facebook, or Amazon Cognito) to authenticate users and allow them to access AWS resources without needing separate AWS credentials.

**how to create user ?**

1)**user name**

2)uncheck permission to access console

3) password

4)create group 5)attach policies 7)create user

**How to create role?**

1)Go to roles

2)create role

3)which permission policy you wan to attach 4)role name user defined create

**Policy structure**

**Version**:-we have version which is by default 2012-10-17

**Statement:**the entire body will come under srtatement.

Statement must be start and end with square brackets

**SID:-some name**

**Effect**:effect must be in two ways allow or deny

**Allow:**he action is permitted

**Action:**which permission to allow we give in action

**Resources** :start with arn:aws:Ec2:avalaibility zone:”account no”:instance/iinstanc id

Give resources of which account and which instance we should give permissions

**Steps to create identity center**

1)**Enable identity center** 2)**choose identity source**(where users and groups are managed by default it will be root) 3)**create** predefined or custom **permission set** (in that how many hours we should control we should specify) 4)**set up account for access identity center** ( that is pick created users and groups in it)5)**sign in to it**

USE CASES:-

**Use case**: A company wants to grant access to specific AWS resources in one account to users in another AWS account, e.g., for auditing purposes.

Here are the steps to grant cross-account access for auditing purposes in one line:

1. **Identify resources** in Account A (e.g., S3, CloudWatch) for access.
2. **Create an IAM role** in Account A with necessary permissions and a trust policy for Account B.
3. **Attach permissions** to the role in Account A for resource access (e.g., S3 read access).
4. **Create a policy in Account B** allowing users to assume the role in Account A.
5. **Grant users in Account B** permission to assume the IAM role in Account A via the AssumeRole action.
6. **Users in Account B** assume the role via AWS STS to get temporary credentials.
7. **Use temporary credentials** to access resources in Account A.
8. **Monitor access** using AWS CloudTrail and CloudWatch for auditing purposes.