AWS Identity and Access Management

Presented by – Khalid Bin Sattar

Agenda

- Overview of AWS IAM.
- Introduction to AWS IAM.
- What Is IAM?
- Security Feature outside of IAM.
- Permission and Policies.
- Create an IAM Admin User and Group.
- IAM Console and Sign-in Page.
- Users.
- Groups.
- Roles.
- Best Practices and Use Cases.

Overview of AWS IAM

What Is IAM?

- AWS Identity and Access Management (IAM) is a web service for securely controlling access to AWS services. With IAM, you can centrally manage users, security credentials such as access keys, and permissions that control which AWS resources users and applications can access.
- You use IAM to control who can use your AWS resources (authentication) and how they can use resources (authorization).

AWS IAM Features

Overview of AWS IAM

AWS IAM Features

- 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 password or access key.
- Granular permissions:-You can grant different permissions to different people for different resources. For example, you might allow some users complete access to Amazon Elastic Compute Cloud (Amazon EC2), Amazon Simple Storage Service (Amazon S3), Amazon DynamoDB, Amazon Redshift, and other AWS services. For other users, 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.
- Secure access to AWS resources for applications that run on Amazon EC2:-You can use IAM features to securely give applications that run on EC2 instances the credentials that they need in order to access other AWS resources, like S3 buckets and RDS or DynamoDB databases.

Overview of AWS IAM

AWS IAM Features

- Multi-factor authentication (MFA):- You can add two-factor authentication to your account and to individual users for extra security. With MFA you or your users must provide not only a password or access key to work with your account, but also a code from a specially configured device.
- Identity federation:- You can allow users who already have passwords elsewhere—for example, in your corporate network or with an Internet identity provider—to get temporary access to your AWS account.
- Identity information for assurance:- If you use AWS CloudTrail, you receive log records that include information about those who made requests for resources in your account. That information is based on IAM identities.

Overview of AWS IAM

AWS IAM Features

- Eventually Consistent:- IAM, like many other AWS services, is eventually consistent. IAM achieves high availability by replicating data across multiple servers within Amazon's data centers around the world. 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. Such changes include creating or updating users, groups, roles, or policies. AWS recommend that you do not include such IAM changes in the critical, high-availability code paths of your application. Instead, make IAM changes in a separate initialization or setup routine that you run less frequently.
- Free to use:- AWS Identity and Access Management is a feature of your AWS account offered at no additional charge. You will be charged only for use of other AWS products by your IAM users. AWS Security Token Service is an included feature of your AWS account offered at no additional charge. You are charged only for the use of other AWS services that are accessed by your AWS STS temporary security credentials.

How to Access AWS IAM

Introduction of AWS IAM

Accessing IAM

- AWS Management Console:-The console is a browser-based interface to manage IAM and AWS resources.

- AWS Command Line Tools:-You can use the AWS command line tools to issue commands at your system's command line to perform IAM and AWS tasks; this can be faster and more convenient than using the console. The command line tools are also useful if you want to build scripts that perform AWS tasks. AWS provides two sets of command line tools: the AWS Command Line Interface (AWS CLI) and the AWS Tools for Windows PowerShell.

Introduction of AWS IAM

Accessing IAM

- AWS SDKs:- AWS provides SDKs (software development kits) that consist of libraries and sample code for various programming languages and platforms (Java, Python, Ruby, .NET, iOS, Android, etc.). The SDKs provide a convenient way to create programmatic access to IAM and AWS. For example, the SDKs take care of tasks such as cryptographically signing requests, managing errors, and retrying requests automatically.

- IAM HTTPS API:- You can access IAM and AWS programmatically by using the IAM HTTPS API, which lets you issue HTTPS requests directly to the service. When you use the HTTPS API, you must include code to digitally sign requests using your credentials.

Overview of Identity Management: Users

Overview of Identity Management: Users

For greater security and organization, you can give access to your AWS account to specific users—identities that you create with custom permissions. You can further simplify access for those users by federating existing identities into AWS.

Topics:-

- First-Time Access Only: Your Root User Credentials.
- IAM Users.
- Federating Existing Users.

Overview of Identity Management: Users

First-Time Access Only: Your Root User Credentials:-

- When you create an AWS account, you create an AWS account root user identity, which you use to sign in to AWS. You can sign in to the AWS Management Console using this root user identity—that is, the email address and password that you provided when creating the account. This combination of your email address and password is also called your root user credentials.
- When you use your root user credentials, you have complete, unrestricted access to all resources in your AWS account, including access to your billing information and the ability to change your password. This level of access is necessary when you first set up your account. However, AWS recommend that you don't use root user credentials for everyday access. AWS specially recommend that you do not share your root user credentials with anyone, because doing so gives them unrestricted access to your account. It is not possible to restrict the permissions that are granted to the root user.

Overview of Identity Management: Users

IAM Users:-

- The “identity” aspect of AWS Identity and Access Management (IAM) helps you with the question “Who is that user?”, often referred to as authentication. Instead of sharing your root user credentials with others, you can create individual IAM users within your account that correspond to users in your organization.
- IAM users are not separate accounts; they are users within your account. Each user can have its own password for access to the AWS Management Console.
- You can also create an individual access key for each user so that the user can make programmatic requests to work with resources in your account.

Overview of Identity Management: Users

Federating Existing Users:-

- If your users already have a way to be authenticated—for example, by signing in to your corporate network—you can federate those user identities into AWS. A user who has already logged in replaces his or her existing identity with a temporary identity in your AWS account. This user can work in the AWS Management Console. Similarly, an application that the user is working with can make programmatic requests using permissions that you define.

Federation is particularly useful in below cases:

1. Your users already have identities in a corporate directory.
2. Your users already have Internet identities.

Overview of Access Management: Permissions and Policies

Overview of Identity Management: Users

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 and then attached to users, groups, or roles.

Policies and Users:-

- 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, resources, and optional conditions.

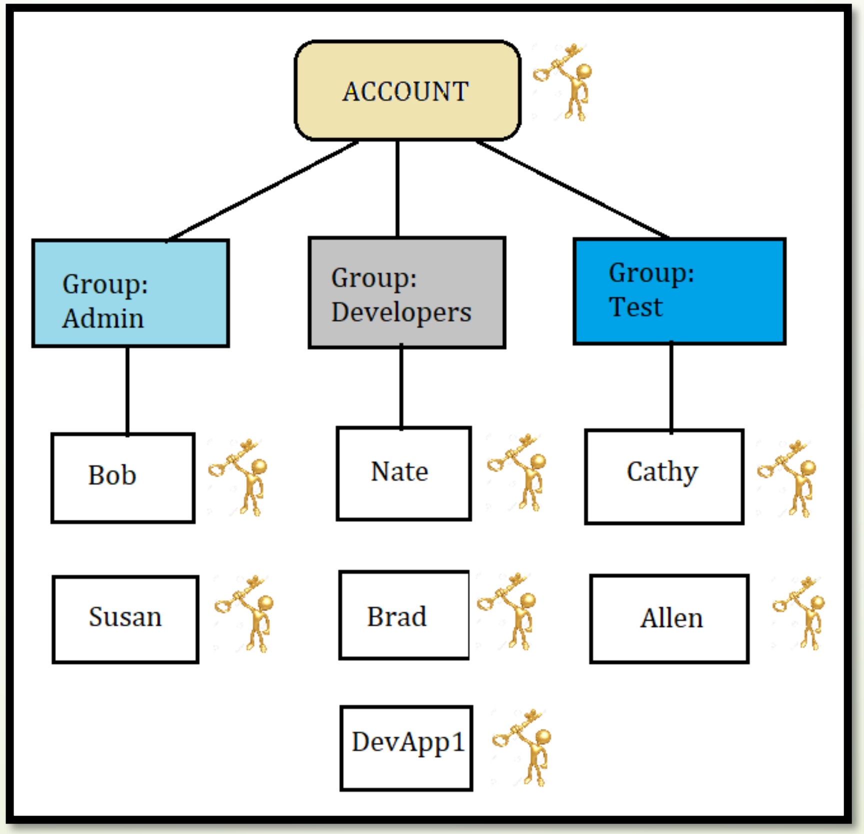
Overview of Identity Management: Users

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "dynamodb:*",
        "Resource": "arn:aws:dynamodb:us-east-2:123456789012:table/Books“
    }
}
```

- ✓ When you attach the policy to a user, group, or role, then the IAM entity has those DynamoDB permissions. Typically, users in your account have multiple policies that together represent the permissions for that user.
- ✓ Any actions or resources that are not explicitly allowed are denied by default. For example, if the above policy is the only policy attached to a user, then that user is allowed to perform DynamoDB actions on the Books table only. Actions on all other tables are prohibited. Similarly, the user is not allowed to perform any actions in Amazon EC2, Amazon S3, or in any other AWS service, because permissions to work with those services are not included in the policy.

Overview of Identity Management: Users

Policies and Groups:-You can organize IAM users into IAM groups and attach a policy to a group. In that case, individual users still have their own credentials, but all the users in a group have the permissions that are attached to the group.



Overview of Identity Management: Users

User-based and Resource-based Policies:-

In some cases you can attach a policy to a resource in addition to attaching it to a user or group. For example, in Amazon S3, you can attach a policy to a bucket. A resource-based policy contains slightly different information than a user-based policy. In a resource-based policy you specify what actions are permitted and what resource is affected (just like a user-based policy). However, you also explicitly list who is allowed access to the resource.

```
{
 "Version": "2012-10-17",
 "Statement": {
  "Effect": "Allow",
  "Principal": {"AWS": "arn:aws:iam::777788889999:user/bob"},
  "Action": [
   "s3:PutObject",
   "s3:PutObjectAcl"
  ],
  "Resource": "arn:aws:s3:::example-bucket/*"
 }
}
```

Quick Links to Common Tasks

Quick Links to Common Tasks

Contents:-

- ✓ Sign in as an IAM user.
- ✓ Manage passwords for IAM users.
- ✓ Manage permissions for IAM users.
- ✓ List the users in your AWS account and get information about their credentials.
- ✓ Get an access key.

Getting Started to AWS IAM

Getting Started to AWS IAM

This topic shows you how to give access to your AWS resources by creating AWS Identity and Access Management (IAM) users under your AWS account. First, you'll learn about IAM concepts you should understand before you create groups and users, and then you'll walk through how to perform the necessary tasks using the AWS Management Console. The first task is to set up an administrators group for your AWS account.

Tasks:-

- ✓ Create an Administrators group and give the group permission to access all of your AWS account's resources.
- ✓ Create a user for yourself and add that user to the Administrators group.
- ✓ Create a password for your user so you can sign in to the AWS Management Console.

How Users Sign In to Your Account

Getting Started to AWS IAM

- ✓ After you create IAM users and passwords for each, users can sign in to the AWS Management Console for your AWS account using your account ID or alias, or from a special URL that includes your account ID.

- ✓ By default, the sign-in URL for your account includes your account ID. You can create a unique sign-in URL for your account so that the URL includes a name instead of an account ID.
- ✓ The sign-in endpoint follows this pattern:

https://AWS-account-ID-or-alias.signin.aws.amazon.com/console

Tutorial: Create and Attach Your First Customer Managed Policy

Create & Attach Your First Customer Managed Policy

In this tutorial, we will use the AWS Management Console to create a customer-managed policy and then attach that policy to an IAM user in your AWS account. The policy you create allows an IAM test user to sign in directly to the AWS Management Console with read only permissions.

This workflow has three basic steps:

- ✓ Step 1: Create the Policy:- By default, IAM users do not have permissions to do anything. They cannot access the AWS Management Console or manage the data within unless you allow it. In this step, you create a customer managed policy that allows any attached user to sign-in to the console.
- ✓ Step 2: Attach the Policy:- When you attach a policy to a user, the user inherits all of the access permissions that are associated with that policy. In this step, you attach the new policy to a test user account.
- ✓ Step 3: Test User Access:-Once the policy is attached, you can sign in as the user and test the policy.

IAM Best Practices: Part 1

IAM Best Practices : Part 1

To help secure your AWS resources, follow below recommendations for the AWS Identity and Access Management (IAM) service.

Topics:-

- ✓ Lock Away Your AWS Account Root User Access Keys.
- ✓ Create Individual IAM Users.
- ✓ Use AWS Defined Policies to Assign Permissions Whenever Possible.
- ✓ Use Groups to Assign Permissions to IAM Users.
- ✓ Grant Least Privilege.
- ✓ Use Access Levels to Review IAM Permissions.
- ✓ Configure a Strong Password Policy for Your Users.

IAM Best Practices : Part 1

- Lock Away Your AWS Account Root User Access Keys:- You use an access key (an access key ID and secret access key) to make programmatic requests to AWS. However, do not use your AWS account root user access key. The access key for your AWS account gives full access to all your resources for all AWS services, including your billing information. You cannot restrict the permissions associated with your AWS account access key.

- Create Individual IAM Users:-

1. Don't use your AWS account root user credentials to access AWS, and don't give your credentials to anyone else. Instead, create individual users for anyone who needs access to your AWS account. Create an IAM user for yourself as well, give that user administrative privileges, and use that IAM user for all your work.
2. By creating individual IAM users for people accessing your account, you can give each IAM user a unique set of security credentials. You can also grant different permissions to each IAM user. If necessary, you can change or revoke an IAM user's permissions any time.

IAM Best Practices : Part 1

- Use AWS Defined Policies to Assign Permissions Whenever Possible:-

1. AWS recommend that you use the managed policies that are created and maintained by AWS to grant permissions whenever possible. A key advantage of using these policies is that they are maintained and updated by AWS as new services or new APIs are introduced.
2. AWS managed policies for job functions can span multiple services and align with common job functions in the IT industry.

- Use Groups to Assign Permissions to IAM Users:- Instead of defining permissions for individual IAM users, it's usually more convenient to create groups that relate to job functions (administrators, developers, accounting, etc.). Next, define the relevant permissions for each group. Finally, assign IAM users to those groups. All the users in an IAM group inherit the permissions assigned to the group. That way, you can make changes for everyone in a group in just one place. As people move around in your company, you can simply change what IAM group their IAM user belongs to.

IAM Best Practices : Part 1

- Grant Least Privilege:- When you create IAM policies, follow the standard security advice of granting least privilege—that is, granting only the permissions required to perform a task. Determine what users need to do and then craft policies for them that let the users perform only those tasks.

- Use Access Levels to Review IAM Permissions:-To improve the security of your AWS account, you should regularly review and monitor each of your IAM policies. Make sure that your policies grant the least privilege that is needed to perform only the necessary actions.

- Configure a Strong Password Policy for Your Users:- If you allow users to change their own passwords, require that they create strong passwords and that they rotate their passwords periodically. On the Account Settings page of the IAM console, you can create a password policy for your account. You can use the password policy to define password requirements, such as minimum length, whether it requires non-alphabetic characters, how frequently it must be rotated, and so on.

IAM Best Practices: Part 2

IAM Best Practices : Part 2

Topics:-

- ✓ Enable MFA for Privileged Users.
- ✓ Use Roles for Applications That Run on Amazon EC2 Instances.
- ✓ Delegate by Using Roles Instead of by Sharing Credentials.
- ✓ Rotate Credentials Regularly.
- ✓ Remove Unnecessary Credentials.
- ✓ Use Policy Conditions for Extra Security.

IAM Best Practices: Part 2

- Enable MFA for Privileged Users:- For extra security, enable multi-factor authentication (MFA) for privileged IAM users (users who are allowed access to sensitive resources or APIs). With MFA, users have a device that generates a unique authentication code (a one-time password, or OTP). Users must provide both their normal credentials (like their user name and password) and the OTP. The MFA device can either be a special piece of hardware, or it can be a virtual device (for example, it can run in an app on a smartphone).

- Use Roles for Applications That Run on Amazon EC2 Instances:- Applications that run on an Amazon EC2 instance need credentials in order to access other AWS services. To provide credentials to the application in a secure way, use IAM roles. A role is an entity that has its own set of permissions, but that isn't a user or group. Roles also don't have their own permanent set of credentials the way IAM users do. In the case of Amazon EC2, IAM dynamically provides temporary credentials to the EC2 instance, and these credentials are automatically rotated for you.

IAM Best Practices: Part 2

- Delegate by Using Roles Instead of by Sharing Credentials:- You might need to allow users 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 that specifies what permissions the IAM users in the other account are allowed. You can also designate which AWS accounts have the IAM users that are allowed to assume the role.

- Rotate Credentials Regularly:- Change your own passwords and access keys regularly, and make sure that all IAM users in your account do as well. That way, if a password or access key is compromised without your knowledge, you limit how long the credentials can be used to access your resources. You can apply a password policy to your account to require all your IAM users to rotate their passwords, and you can choose how often they must do so.

IAM Best Practices: Part 2

- Remove Unnecessary Credentials:- Remove IAM user credentials (that is, passwords and access keys) that are not needed. For example, an IAM user that is used for an application does not need a password (passwords are necessary only to sign in to AWS websites). Similarly, if a user does not and will never use access keys, there's no reason for the user to have them. Passwords and access keys that have not been used recently might be good candidates for removal. You can find unused passwords or access keys using the console, using the API, or by downloading the credentials report.

- Use Policy Conditions for Extra Security:- To the extent that it's practical, define the conditions under which your IAM policies allow access to a resource. For example, you can write conditions to specify a range of allowable IP addresses that a request must come from. You can also specify that a request is allowed only within a specified date range or time range. You can also set conditions that require the use of SSL or MFA (multi-factor authentication). For example, you can require that a user has authenticated with an MFA device in order to be allowed to terminate an Amazon EC2 instance.

Identities (Users, Groups, and Roles)

Identities (Users, Groups, and Roles)

- This session describes IAM identities, which you create to provide authentication for people and processes in your AWS account. This session also describes IAM groups, which are collections of IAM users that you can manage as a unit.

- Identities represent 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 a group can do with which AWS resources and under what conditions.

Identities (Users, Groups, and Roles)

IAM Users:-

- An IAM user is an entity that you create in AWS. The IAM user represents the person or service who uses the IAM user to interact with AWS. A primary use for IAM users is to give people the ability to sign in to the AWS Management Console for interactive tasks and to make programmatic requests to AWS services using the API or CLI.

- A user in AWS consists of a name, a password to sign into the AWS Management Console, and up to two access keys that can be used with the API or CLI. When you create an IAM user, you grant it permissions by making it a member of a group that has appropriate permission policies attached (recommended), or by directly attaching policies to the user.

- You can also clone the permissions of an existing IAM user, which automatically makes the new user a member of the same groups and attaches all the same policies.

Identities (Users, Groups, and Roles)

IAM Groups:-

- An IAM group is a collection of IAM users. You can use groups to specify permissions for a collection of users, which can make those permissions easier to manage for those users. For example, you could have a group called Admins and give that group the types of permissions that administrators typically need.
- Any user in that group automatically has the permissions 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.
- Similarly, if a person changes jobs in your organization, instead of editing that user's permissions, you can remove him or her from the old groups and add him or her to the appropriate new groups.
- Note that a group is not truly an identity because it cannot be identified as a Principal in an access policy. It is only a way to attach policies to multiple users at one time.

Identities (Users, Groups, and Roles)

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. However, a role does not have any credentials (password or access keys) associated with it.

- ❑ Instead of being uniquely 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.

- ❑ A role can be assigned to a federated user who signs in by using an external identity provider instead of IAM. AWS uses details passed by the identity provider to determine which role is mapped to the federated user.

Identities (Users, Groups, and Roles)

Temporary Credentials:-

- Temporary credentials are primarily 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 user. This prevents you from accidentally performing tasks that are not permitted by the more restricted credentials. A benefit of temporary credentials is that they expire automatically after a set period of time. You have control over the duration that the credentials are valid.

The AWS Account 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. You can sign in to the AWS Management Console as the root user—that is, the email address and password that you provide when you create the account. This combination of your email address and password is called your root user credentials.

Creating an IAM User in Your AWS Account

Creating an IAM User in Your AWS Account

- You can create one or more IAM users in your AWS account. You might create an IAM user when someone joins your organization, or when you have a new application that needs to make API calls to AWS.

Topics:-

- Creating IAM Users (Console)
- Creating IAM Users (AWS CLI, Tools for Windows PowerShell, or IAM HTTP API)

Managing IAM Users

Managing IAM Users

- Amazon Web Services offers multiple tools for managing the IAM users in your AWS account.

Topics:-

- Listing IAM Users.
- Renaming an IAM User.
- Deleting an IAM User.

Creating IAM Groups

Creating IAM Groups

- To set up a group, you need to create the group, give it permissions based on the type of work that you expect the users in the group to do, and then add users to the group.

Topics:-

- Creating IAM Group(Console)
- Creating IAM Group (AWS CLI, Tools for Windows PowerShell, or IAM HTTP API)

Managing IAM Groups

Creating IAM Groups

- Amazon Web Services offers multiple tools for managing IAM groups.

Topics:-

- Listing IAM Groups.
- Adding and Removing Users in an IAM Group.
- Attaching a Policy to an IAM Group.
- Renaming an IAM Group.
- Deleting an IAM Group.

IAM Roles

IAM Roles

- An IAM role is similar to a user, in that it is an AWS identity with permission policies that determine what the identity can and cannot do in AWS. However, instead of being uniquely associated with one person, a role is intended to be assumable by anyone who needs it. Also, a role does not have any credentials (password or access keys) associated with it. Instead, if a user is assigned to a role, access keys are created dynamically and provided to the user.

- You can use roles to delegate access to users, applications, or services that don't normally have access to your AWS resources. For example, you might want to grant users in your AWS account access to resources they don't usually have, or grant users in one AWS account access to resources in another account. Or you might want to allow a mobile app to use AWS resources, but not want to embed AWS keys within the app.

- Sometimes you want to give AWS access to users who already have identities defined outside of AWS, such as in your corporate directory. Or, you might want to grant access to your account to third parties so that they can perform an audit on your resources.

Roles Terms and Concepts

Roles Terms and Concepts

Role:-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 group. Roles can be used by the following:

- ❑ An IAM user in the same AWS account as the role
- ❑ An IAM user in a different AWS account as the role
- ❑ A web service offered by AWS such as Amazon Elastic Compute Cloud (Amazon EC2)
- ❑ An external user authenticated by an external identity provider (IdP) service that is compatible with SAML 2.0 or OpenID Connect, or a custom-built identity broker.

AWS service role:-

- ❑ A role that a service assumes to perform actions on your behalf. When you set up most AWS service environments, you must define a role for the service to assume.
- ❑ This service role must include all the permissions required for the service to access the AWS resources that it needs. Service roles vary from service to service, but many allow you to choose your permissions, as long as you meet the documented requirements for that service. You can create, modify, and delete a service role from within IAM.

Roles Terms and Concepts

AWS service role for an EC2 instance:-

- A special type of service role that a service assumes to launch an Amazon EC2 instance that runs your application. This role is assigned to the EC2 instance when it is launched. AWS automatically provides temporary security credentials that are attached to the role and then makes them available for the EC2 instance to use on behalf of its applications.

AWS service-linked role:-

- A unique type of service role that is linked directly to an AWS service. Service-linked roles are predefined by the service and include all the permissions that the service requires to call other AWS services on your behalf. The linked service also defines how you create, modify, and delete a service-linked role.
- A service might automatically create or delete the role. It might allow you to create, modify, or delete the role as part of a wizard or process in the service. Or it might require that you use IAM to create or delete the role. Regardless of the method, service-linked roles make setting up a service easier because you don’t have to manually add the necessary permissions.

Roles Terms and Concepts

Delegation:-The granting of permission to someone to allow access to resources 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 any of the following:

- The same account.
- Two accounts that are both under your (organization's) control.
- Two accounts owned by different organizations.

Principal:-

- An entity in AWS that can perform actions and access resources. A principal can be an AWS account root user, an IAM user, or a role. You can grant permissions to access a resource in one of two ways:
- You can attach a permissions policy to a user (directly, or indirectly through a group) or to a role.
- For those services that support resource-based policies, you can identify the principal in the Principal element of a policy attached to the resource.

Create an IAM Role for AWS EC2

Managing IAM Roles

Roles Terms and Concepts

Occasionally you need to modify or delete the roles that you have created. To change a role you can modify the policies associated with the role, change who can access the role, and edit the permissions that the role grants to users.

You can also delete roles that are no longer needed. You can manage your roles from the AWS Management Console, the AWS CLI, and the API.

Topics :-

- ❑ Modifying a Role.
- ❑ Deleting Roles or Instance Profiles.

Overview of AWS IAM Permissions

Overview of AWS IAM Permissions

- Permissions let you specify who has access to AWS resources, and what actions they can perform on those resources. Every IAM user starts with no permissions. In other words, by default, users can do nothing, not even view their own access keys. To give a user permission to do something, you can add the permission to the user (that is, attach a policy to the user) or add the user to a group that has the desired permission.

- For example, you might grant a user permission to list his or her own access keys. You might also expand that permission and also let each user create, update, and delete their own keys.

- When you give permissions to a group, all users in that group get those permissions. For example, you can give the Admins group permission to perform any of the IAM actions on any of the AWS account resources. Another example: You can give the Managers group permission to describe the AWS account's Amazon EC2 instances.

Overview of IAM Policies

Overview of IAM Policies

This section provides an overview of IAM policies. A policy is a document that formally defines permissions.

To assign permissions to a user, group, role, or resource, you create a policy, which is a document that defines permissions. The policy document includes the following elements:

- Effect – whether the policy allows or denies access.
- Action – the list of actions that are allowed or denied by the policy.
- Resource – the list of resources on which the actions can occur.
- Condition (Optional) – the circumstances under which the policy grants permission.

Overview of IAM Policies

Policies are documents that are created using JSON. A policy consists of one or more statements, each of which describes one set of permissions.

```
{
 "Version": "2012-10-17",
 "Statement": {
  "Effect": "Allow",
  "Action": "s3:ListBucket",
  "Resource": "arn:aws:s3:::example_bucket"
 }
}
```

You can attach this policy to an IAM user or group. If that's the only policy for the user or group, the user or group is allowed to perform only this one action (ListBucket) on one Amazon S3 bucket (**example_bucket**).

Overview of IAM Policies

The following example shows a policy that might be attached to an Amazon S3 bucket and that grants permission to a specific AWS account to perform any Amazon S3 actions in mybucket. This includes both working with the bucket and with the objects in it.

```
{
 "Version": "2012-10-17",
 "Id": "S3-Account-Permissions",
 "Statement": [{
  "Sid": "1",
  "Effect": "Allow",
  "Principal": {"AWS": ["arn:aws:iam::ACCOUNT-ID-WITHOUT-HYPHENS:root"]},
  "Action": "s3:*",
  "Resource": [
   "arn:aws:s3:::mybucket",
   "arn:aws:s3:::mybucket/*"
  ]
 }]
}
```

Creating a New Policy

Overview of IAM Policies

- You have several ways to create a new IAM permission policy. You can copy a complete AWS managed policy that already does some of what you're looking for and then customize it to your specific requirements. Alternatively, you can construct the policy by selecting actions and conditions from lists in the policy generator to build the statements into a policy for you. Or you can create a policy from scratch by writing the JSON code.

- A policy consists of one or more statements. Each statement generally contains all the actions that share the same effect (Allow or Deny) and the same resources. If one action requires "*" for the resource, and another action specifies the ARN of a specific resource, then they must be in two separate statements.

AWS IAM Limits

Amazon VPC(Virtual Private Cloud)Demonstration

Demonstration of Creating AWS VPC and its components using AWS Management Console.

Amazon EC2 Demonstration

Demonstration of AWS EC2 by following ways:-

1. AWS Management Console.
2. AWS CLI (Command Line Interface).
3. AWS SDK (Software development Kits).

AWS DEMONSTRATIONS
Accelerate the adoption of AWS in the enterprise with integrated solutions in:
DevOps
Amazon WorkSpaces
ServiceNow