



Placement Empowerment Program

Cloud Computing and DevOps Centre

Set Up IAM Roles and Permissions : Create an IAM role on your cloud platform. Assign the role to your VM to restrict/allow specific actions.

Name: Akshaya S

Department: CSE

Introduction

This Proof of Concept (PoC) demonstrates the process of setting up and utilizing IAM roles and permissions in AWS. The goal is to show how to secure AWS resources by managing access through roles rather than hardcoding credentials. Specifically, this PoC focuses on creating an IAM role, assigning it to an EC2 instance, and verifying the instance's access to AWS services such as Amazon S3.

Overview

The process is divided into several key steps:

- 1. Create an IAM Role:** Define a role in AWS IAM and attach policies that grant permissions for specific AWS services.
- 2. Launch an EC2 Instance:** Create a virtual machine (VM) in AWS and configure it for testing the assigned IAM role.
- 3. Assign the IAM Role to the EC2 Instance:** Attach the created IAM role to the EC2 instance to enable access to AWS services without using access keys.
- 4. Verify Access:** Test the EC2 instance to confirm that it has the appropriate permissions by interacting with services like Amazon S3.

Objectives

This PoC aims to achieve the following objectives:

1. **Secure Access:** Implement IAM roles to grant temporary permissions to AWS resources without embedding credentials.
2. **Demonstrate Role-Based Permissions:** Show how roles can restrict or allow actions based on attached policies.
3. **Test Least Privilege Principle:** Ensure that the EC2 instance only has the permissions it needs to perform specific tasks.
4. **Hands-On Learning:** Provide practical experience with IAM roles and their applications in a cloud environment.

Importance

IAM roles and permissions are fundamental to securing cloud environments. They allow for fine-grained access control and improve operational efficiency by:

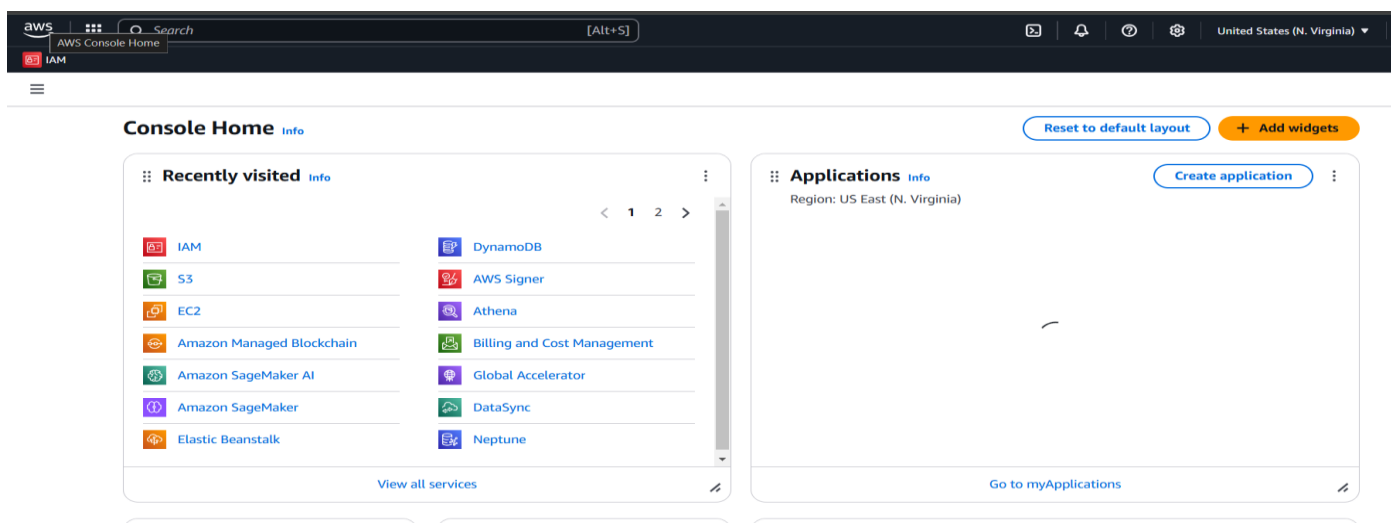
1. **Eliminating Hardcoded Credentials:** Reducing security risks by avoiding the storage of access keys in applications or instances.
2. **Granting Least Privilege Access:** Ensuring users and resources only have the permissions they require, minimizing potential misuse.
3. **Improving Compliance:** Enforcing organizational policies and audit requirements.

4. **Enhancing Automation:** Allowing resources like EC2 instances to securely interact with other AWS services.

Step-by-Step Overview Step

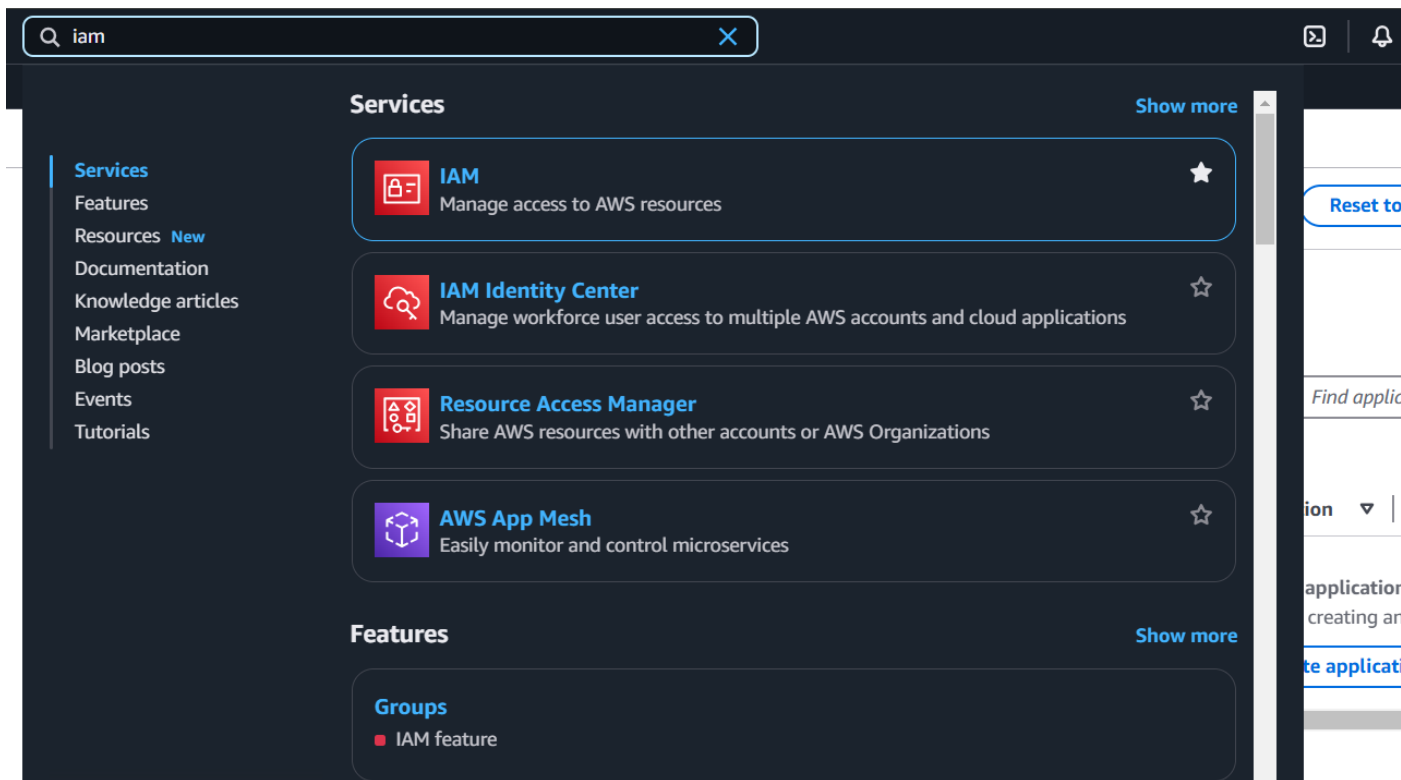
1:

1. Go to [AWS Management Console](#).
2. Enter your username and password to log in.



Step 2:

1. In the AWS Management Console, type "**IAM**" in the search bar at the top.
2. Click on **IAM** from the search results.



Step 3:

1. On the IAM dashboard, click on "**Roles**" in the left-hand menu.
2. On the Roles page, click the "**Create Role**" button.

aws [Search] [Alt+S]

Identity and Access Management (IAM) <

Search IAM

Dashboard

▼ Access management

- User groups
- Users
- Roles**
- Policies
- Identity providers
- Account settings
- Root access management **New**

▼ Access reports

- Access Analyzer
- External access
- Unused access

Roles (44) [Info](#)

An IAM role is an identity you can create that has specific permissions with credentials that are valid for short durations. Roles can be assumed by ent

Search

<input type="checkbox"/>	Role name	Trusted entities	Last activity
<input type="checkbox"/>	AmazonSageMaker-ExecutionRole-20241219T224079	AWS Service: sagemaker	-
<input type="checkbox"/>	AmazonSageMakerCanvasBedrockRole-20241219T224078	AWS Service: bedrock	-
<input type="checkbox"/>	AmazonSageMakerCanvasEMRSEExecutionAccess-20241219T224078	AWS Service: emr-serverless	-
<input type="checkbox"/>	AmazonSageMakerCanvasForecastRole-20241219T224078	AWS Service: forecast	-
<input type="checkbox"/>	AmazonSageMakerServiceCatalogProductsApiGatewayRole	AWS Service: apigateway	-
<input type="checkbox"/>	AmazonSageMakerServiceCatalogProductsCloudformationRole	AWS Service: cloudformation	-
<input type="checkbox"/>	AmazonSageMakerServiceCatalogProductsCodeBuildRole	AWS Service: codebuild	-
<input type="checkbox"/>	AmazonSageMakerServiceCatalogProductsCodePipelineRole	AWS Service: codepipeline	-
<input type="checkbox"/>	AmazonSageMakerServiceCatalogProductsEventsRole	AWS Service: events	-
<input type="checkbox"/>	AmazonSageMakerServiceCatalogProductsExecutionRole	AWS Service: sagemaker	-

Step 4:

1. On the **"Create Role"** page, under **Trusted Entity Type**, select **AWS Service** (it should be selected by default).
2. In the **Use Case** dropdown, choose **EC2**.

Click **Next** to continue

Step 2
Add permissions

Step 3
Name, review, and create

Trusted entity type

- ☒ **AWS service**
Allow AWS services like EC2, Lambda, or others to perform actions in this account.
- ☐ **AWS account**
Allow entities in other AWS accounts belonging to you or a 3rd party to perform actions in this account.
- ☐ **Web identity**
Allows users federated by the specified external web identity provider to assume this role to perform actions in this account.
- ☐ **SAML 2.0 federation**
Allow users federated with SAML 2.0 from a corporate directory to perform actions in this account.
- ☐ **Custom trust policy**
Create a custom trust policy to enable others to perform actions in this account.

Use case

Allow an AWS service like EC2, Lambda, or others to perform actions in this account.

Service or use case

EC2 ▼

Choose a use case for the specified service.

Use case

- ☒ **EC2**

Step 5:

1. On the **Permissions** page, you'll see a list of policies.
2. Select a policy based on what actions you want the VM to perform. For example:

To give the VM **read-only access to S3**, select **AmazonS3ReadOnlyAccess**.

You can search for policies in the search bar (e.g., type "S3" for S3 policies).

3. Once you've selected a policy, click **Next**.

Trust policy

```
1 {  
2   "Version": "2012-10-17",  
3   "Statement": [  
4     {  
5       "Effect": "Allow",  
6       "Action": [  
7         "sts:AssumeRole"  
8       ],  
9       "Principal": {  
10        "Service": [  
11          "ec2.amazonaws.com"  
12        ]  
13      }  
14    }  
15  ]  
16 }
```

Step 2: Add permissions

[Edit](#)

Permissions policy summary

Policy name 	Type	Attached as
AmazonS3ReadOnlyAccess	AWS managed	Permissions policy

Step 6:

1. On the **Role Details** page:

- Enter a name for your role (e.g., My-EC2-S3-Access-Role).
- (Optional) Add a description or tags if you'd like.

2. Click **Create Role** to finish.

Name, review, and create

Role details

Role name

Enter a meaningful name to identify this role.

Akshu12

Maximum 64 characters. Use alphanumeric and '+=, @-/_[]!#\$%^&*()~;' characters.

Description

Add a short explanation for this role.

Allows EC2 instances to call AWS services on your behalf.

Maximum 1000 characters. Use letters (A-Z and a-z), numbers (0-9), tabs, new lines, or any of the following characters: '+=, @-/_[]!#\$%^&*()~;'.

Step 1: Select trusted entities

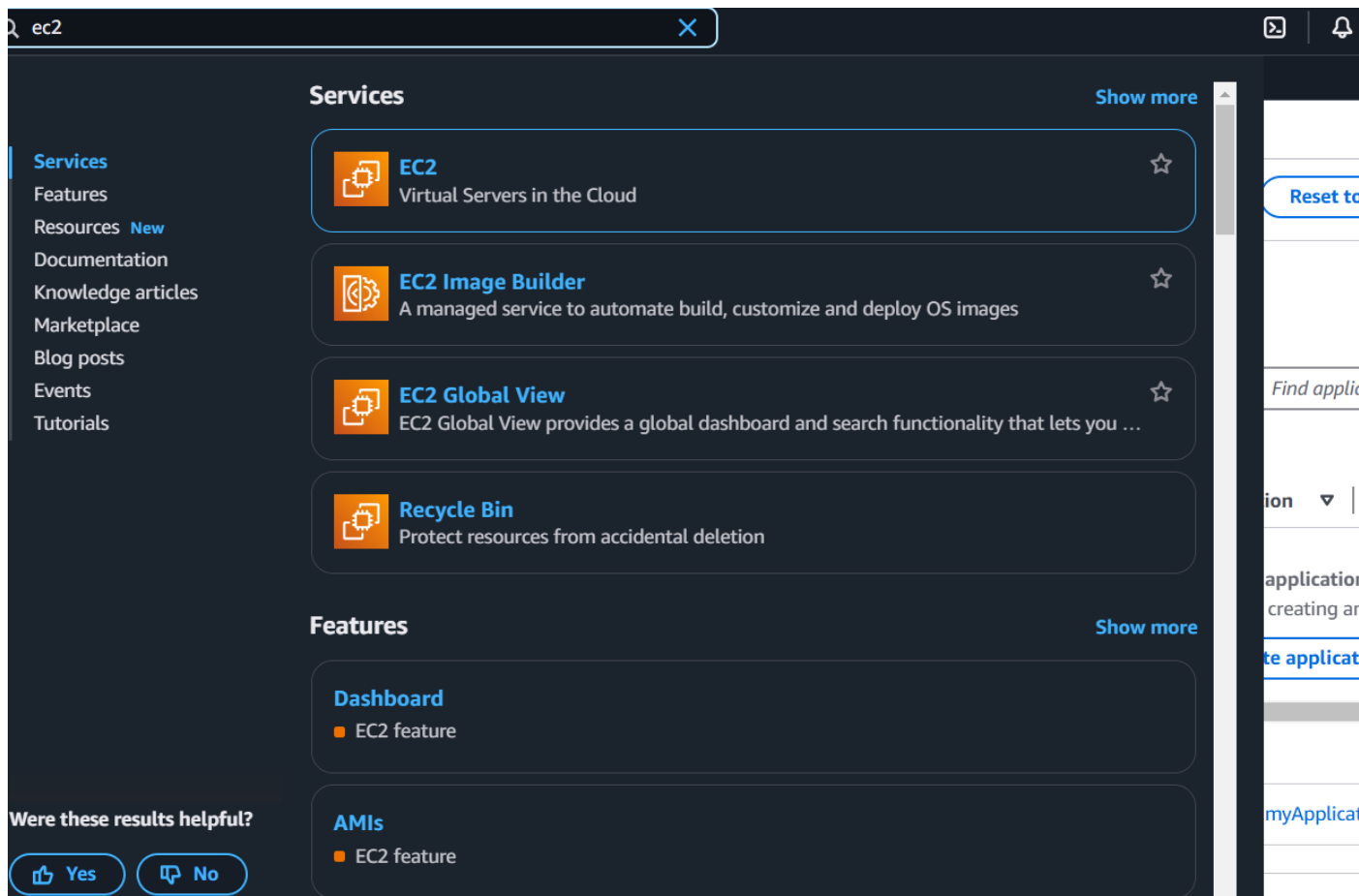
[Edit](#)

Trust policy

```
1 {  
2   "Version": "2012-10-17",  
3   "Statement": [  
4     {  
5       "Effect": "Allow",  
6       "Principal": {  
7         "Service": "ec2.amazonaws.com"  
8       },  
9       "Action": "sts:AssumeRole"  
10    }  
11  ]  
12 }
```

Step 7:

1. In the AWS Management Console, search for **EC2** and click to open the **EC2 Dashboard**.
2. Select the instance (VM) you want to assign the IAM role to.



Step 8:

1. In the **Instance details** section, click **Actions** in the top right corner.
2. From the dropdown, choose **Security** > **Modify IAM Role**.

The screenshot shows the AWS Management Console interface. On the left is a navigation sidebar with categories like Dashboard, EC2 Global View, Events, Instances, Images, Elastic Block Store, Network & Security, and Load Balancing. The main content area is titled 'Instances (1/4)' and shows a table of running instances. The instance 'my web' with ID 'i-Od78839df427e0afa' is selected. A dropdown menu for actions is open, showing options like 'Connect', 'View details', 'Change security groups', and 'Modify IAM role'. Below the table, the details for the selected instance are shown, including its public IPv4 address (16.170.246.201) and state (Running).

3.

Step 9:

1. In the **Modify IAM role** window, you should see a dropdown for **IAM role**.
2. Select the role you created earlier (e.g., My-EC2-S3-AccessRole).
3. Click **Update IAM role** to apply the changes.

The screenshot shows the 'Modify IAM role' window. At the top, the breadcrumb navigation reads 'EC2 > Instances > i-Od78839df427e0afa > Modify IAM role'. The main heading is 'Modify IAM role' with an 'Info' link. Below it, the text says 'Attach an IAM role to your instance.' The 'Instance ID' is 'i-Od78839df427e0afa (my web)'. The 'IAM role' section has a description: 'Select an IAM role to attach to your instance or create a new role if you haven't created any. The role you select replaces any roles that are currently attached to your instance.' A dropdown menu shows 'Akshu12' as the selected role, with a 'Create new IAM role' link next to it. At the bottom right, there are 'Cancel' and 'Update IAM role' buttons.

Step 10:

1. Open your terminal (if you're using Linux or macOS) or Command Prompt (Windows).
2. Use SSH to log in to your EC2 instance. For example:

```
ssh -i "your-key-pair.pem" ec2-user@your-ec2-public-ip
```

Step 11:

```
[ec2-user@ip-172-31-80-54 ~]$ aws ec2 describe-regions --query "Regions[*].RegionName"
```

The error confirms that your IAM role (My-EC2-S3-Access-Role) does not have permissions to perform the **ec2:DescribeRegions** action. The role currently only has S3-related permissions (e.g., AmazonS3ReadOnlyAccess) and doesn't include broader EC2 permissions.

```
PS C:\Users\Hi> cd downloads
PS C:\Users\Hi\downloads> ssh -i "newkey.pem" ec2-user@ec2-54-85-184-134.compute-1.amazonaws.com
```

The terminal window shows the SSH connection process. The prompt changes from PS to #, indicating a successful login. The system banner for Amazon Linux 2023 is displayed, followed by the AWS support URL. The user then runs two AWS CLI commands: aws s3 ls and aws ec2 describe-regions --query "Regions[*].RegionName". Both commands result in an error message stating that the user is not authorized to perform the DescribeRegions operation.

```
#_
~#   #####      Amazon Linux 2023
  ~#   #####\
    ~#   ####|
      ~#   \###|
        ~#   \#/ ---> https://aws.amazon.com/linux/amazon-linux-2023
          ~#   V~'
            ~#   /
              ~# /
                ~#/'

[ec2-user@ip-172-31-80-54 ~]$ aws s3 ls
[ec2-user@ip-172-31-80-54 ~]$ aws ec2 describe-regions --query "Regions[*].RegionName"
```

An error occurred (UnauthorizedOperation) when calling the DescribeRegions operation: You are not authorized to perform this operation. User: arn:aws:sts::343218194491:assumed-role/My-EC2-S3-Access-Role/i-05d8e9de4e855f257 is not authorized to perform: ec2:DescribeRegions because no identity-based policy allows the ec2:DescribeRegions action
[ec2-user@ip-172-31-80-54 ~]\$

Outcome

By completing this PoC of setting up IAM roles and permissions with an EC2 instance, you will:

1. Create an IAM role and attach policies to control access to specific AWS services.
2. Launch and configure an EC2 instance for testing purposes.
3. Assign the IAM role to the EC2 instance securely without using access keys.
4. Verify permissions by interacting with AWS services (e.g., listing S3 buckets) from the EC2 instance.
5. Demonstrate the principle of least privilege by ensuring only necessary permissions are granted.