



Module 2 - IAM

Saxena, Piyush

IAM (Identity and Access Management).

- IAM allows you to create and manage users, groups, roles, and policies.
- Users represent individual people who need access to your AWS resources.
- Groups are collections of users with similar access requirements.
- Roles are used to grant temporary access to external entities or services.
- Policies are written in JSON format and specify what actions are allowed or denied on specific AWS resources. These policies can be attached to IAM entities (users, groups, or roles) to grant or restrict access to AWS services and resources.

How to create user and their access keys.

- Go to the IAM services and click on Create User.
- Enter the username that you want to create and then select on AWS management console. This will help the user to use the console / AWS Portal.
- Click on Next, Now assigns the permissions that we want this user to have. Let's provide user with Admin permission.
- Click on Next and then click on create user.

Access keys.

- click on Security credentials → create access key.

Access key best practices & alternatives [Info](#)

Avoid using long-term credentials like access keys to improve your security. Consider the following use cases and alternatives.

Use case

☐ **Command Line Interface (CLI)**

You plan to use this access key to enable the AWS CLI to access your AWS account.

☐ **Local code**

You plan to use this access key to enable application code in a local development environment to access your AWS account.

☐ **Application running on an AWS compute service**

You plan to use this access key to enable application code running on an AWS compute service like Amazon EC2, Amazon ECS, or AWS Lambda to access your AWS account.

☐ **Third-party service**

You plan to use this access key to enable access for a third-party application or service that monitors or manages your AWS resources.

☐ **Application running outside AWS**

You plan to use this access key to authenticate workloads running in your data center or other infrastructure outside of AWS that needs to access your AWS resources.

☐ **Other**

Your use case is not listed here.

Cancel

Next

Set description tag - *optional* [Info](#)

The description for this access key will be attached to this user as a tag and shown alongside the access key.

Description tag value

Describe the purpose of this access key and where it will be used. A good description will help you rotate this access key confidently later.

Connect through AWS CLI

Maximum 256 characters. Allowed characters are letters, numbers, spaces representable in UTF-8, and: _ . : / = + - @

Cancel

Previous

Create access key

Download the .csv file for the future reference.

Retrieve access keys [Info](#)

Access key

If you lose or forget your secret access key, you cannot retrieve it. Instead, create a new access key and make the old key inactive.

Access key

Secret access key

AKIAJAJND2R7AKVWU6PE

***** [Show](#)

Access key best practices

- Never store your access key in plain text, in a code repository, or in code.
 - Disable or delete access key when no longer needed.
 - Enable least-privilege permissions.
 - Rotate access keys regularly.
- For more details about managing access keys, see the [best practices for managing AWS access keys](#).

[Download .csv file](#)

Done

AWS CLI installation.

Go to <https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html> and select the operating system where we want to install.

Linux –

Run the below command. OS – RedHat

```
curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
```

```
[ec2-user@ip-10-230-253-50 ~]$ curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
```

unzip awscliv2.zip

```
[ec2-user@ip-10-230-253-50 ~]$ unzip awscliv2.zip
```

sudo ./aws/install

```
[ec2-user@ip-10-230-253-50 ~]$ sudo ./aws/install
```

aws --version

```
[ec2-user@ip-10-230-253-50 ~]$ aws --version
aws-cli/2.15.0 Python/3.11.6 Linux/5.14.0-362.8.1.el9_3.x86_64 exe/x86_64.rhel.9 prompt/off
[ec2-user@ip-10-230-253-50 ~]$
```


Windows –

Download the msi package

<https://awscli.amazonaws.com/AWSCLIV2.msi>

double click on the file and install it.

Once its installed, open cmd to verify.

 Administrator: Command Prompt

```
Microsoft Windows [Version 10.0.20348.2113]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Administrator>aws --version
aws-cli/2.15.0 Python/3.11.6 Windows/10 exec-env/EC2 exe/AMD64 prompt/off

C:\Users\Administrator>
```

How to create Roles and Policies.

We will be using below policy. This Policy will restrict the user to use any services apart from the services in **US-EAST-1**

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": "*",
      "Effect": "Deny",
      "Resource": [
        "*"
      ],
      "Condition": {
        "StringNotEquals": {
          "aws:RequestedRegion": "us-east-1"
        }
      }
    }
  ]
}
```

What is Inline Policy - An inline policy is a policy created for a single IAM identity (a user, group, or role). Inline policies maintain a strict one-to-one relationship between a policy and an identity. They are deleted when you delete the identity.

Role.

- Click on Create Role.
- Select entity as AWS Services and use cases as EC2 and then click Next.
- Here we are going to assign what Permission we are going to provide the users. In this case we are going to select SSM permission and later we will use this role to attach to EC2 instance.
- So, select below two permission –
 - **AmazonEC2RoleforSSM**
 - **AmazonSSMManagedEC2InstanceDefaultPolicy**
- Click on Next
- Specify the role name and give the Description.