

Programatic access in IAM

Steps:-

- 1) Go inside IAM console
- 2) Select users
- 3) Select username
- 4) Click on security Credentials

Identity and Access Management (IAM)

Search IAM

Dashboard

Access management

- User groups
- Users**
- Roles
- Policies
- Identity providers
- Account settings

IAM > Users > vishal

vishal

Summary

| | |
|--|--|
| ARN arn:aws:iam::968037226900:user/vishal | Console access Enabled without MFA |
| Created March 02, 2023, 15:41 (UTC+05:30) | Last console sign-in Never |

Permissions | Groups | Tags | **Security credentials** | Access Advisor

- 5) Scroll down and click on **create access key**

Access keys (0)

Use access keys to send programmatic calls to AWS from the AWS CLI, AWS Tools for PowerShell, AWS SDKs, or direct AWS API calls. You can have a maximum of two access keys (active or inactive) at a time. [Learn more](#)

No access keys

As a best practice, avoid using long-term credentials like access keys. Instead, use tools which provide short term credentials. [Learn mo](#)

Create access key

- 6) Select Command line interface
- 7) Check the box of I Understand.....
- 8) Click on next

IAM > Users > vishal > Create access key

Step 1
Access key best practices & alternatives

Step 2 - optional
Set description tag

Step 3
Retrieve access keys

Access key best practices & alternatives

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

- ☒ **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 enable an application running on an on-premises host, or to use a local AWS client or third-party AWS plugin.
- ☐ **Other**
Your use case is not listed here.

Alternatives recommended

- Use AWS CloudShell, a browser-based CLI, to run commands. [Learn more](#)
- Use the AWS CLI V2 and enable authentication through a user in IAM Identity Center. [Learn more](#)

☒ I understand the above recommendation and want to proceed to create an access key.

Cancel

9) Give any tag

10) Click on create access key

IAM > Users > vishal > Create access key

Step 1
Access key best practices & alternatives

Step 2 - optional
Set description tag

Step 3
Retrieve access keys

Set description tag - optional

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.

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

Cancel Previous **Create access key**

11) Copy access key and secret key to notepad and click on done

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 |
|----------------------|----------------------------|
| AKIA6CY4D3GKJZ42SLXK | ***** 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

12) Now open any terminal of linux here I am using ec2 instance

13) Now write **sudo - i** command in linux to go inside root user

14) Now write **aws configure** to take programmatic access in linux

15) Paste access key and secret key from notepad

16) Hit enter enter

17) Now you have access of user by using programmatic access

```
__|  __|_ )
_| (  /   Amazon Linux 2 AMI
__|\__|__|

https://aws.amazon.com/amazon-linux-2/
ec2-user@ip-172-31-13-214 ~]$ sudo -i
[root@ip-172-31-13-214 ~]# aws configure
AWS Access Key ID [None]: AKIA6CY4D3GKJZ42SLXK
AWS Secret Access Key [None]: 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
Default region name [None]:
Default output format [None]:
[root@ip-172-31-13-214 ~]#
```