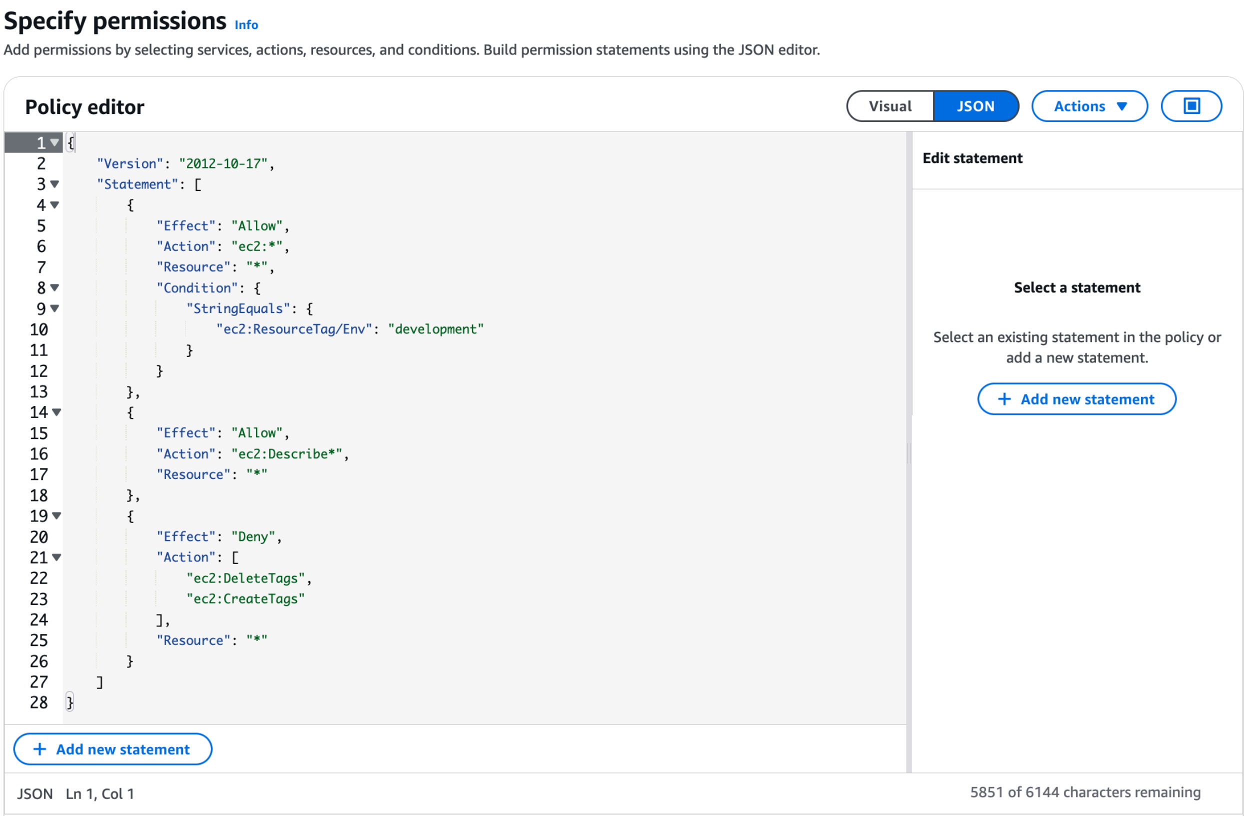
Cloud Security with AWS IAM



Specify permissions Info

Add permissions by selecting services, actions, resources, and conditions. Build permission statements using the JSON editor.

Policy editor

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:*",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "ec2:ResourceTag/Env": "development"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "ec2:Describe*",
            "Resource": "*"
        },
        {
            "Effect": "Deny",
            "Action": [
                "ec2:DeleteTags",
                "ec2:CreateTags"
            ],
            "Resource": "*"
        }
    ]
}
```

Edit statement

Select a statement

Select an existing statement in the policy or add a new statement.

JSON Ln 1, Col 1

5851 of 6144 characters remaining

Introducing today's project!

What is AWS IAM?

AWS IAM (Identity and Access Management) is a service that helps securely control access to AWS resources by managing authentication (who can sign in) and authorization (what they can do).

How I'm using AWS IAM in this project

In today's project, I used AWS IAM to manage access for two EC2 instances by creating groups, users, and policies. I attached policies to users, ensuring proper permissions, and verified the setup for accuracy and security.

This project took me...

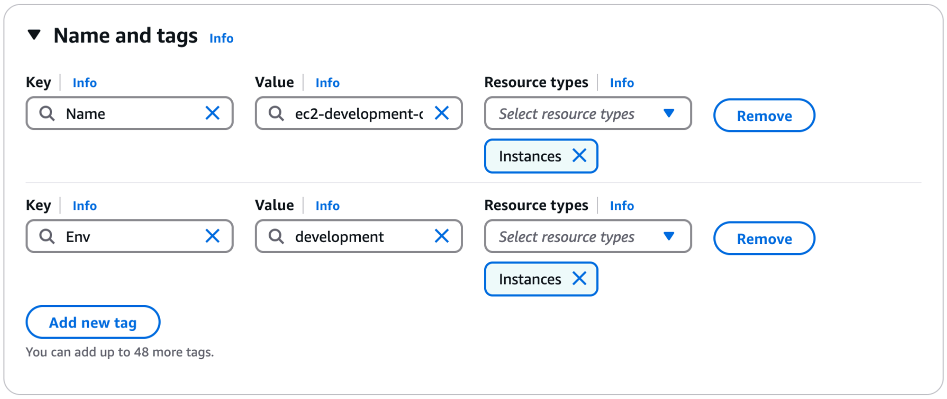
The project was completed in 25 minutes, covering planning, setting up groups and users, configuring EC2 instances, and testing the setup.

Task	Time Spent
Planning IAM Policies	10 minutes
Setting Up Groups and Users	5 minutes
Configuring EC2 Instances	5 minutes
Testing the Setup	5 minutes
Total	**25 minutes**

Tags

Tags in AWS EC2 are key-value pairs used to organize and manage resources. They help with cost tracking, access control, automation, and compliance. Tags enable easy identification and filtering of instances, enhancing efficiency and security.

The tag I've used on my EC2 instances is called "Env." The values I've assigned for my instances are "production" and "development." This represents two different environments.



Name and tags

Key	Value	Resource types
Name	ec2-development-	Instances
Env	development	Instances

You can add up to 48 more tags.

IAM Policies

IAM (Identity and Access Management) Policies are JSON (JavaScript Object Notation) documents that define permissions for users, groups, and roles in AWS. They specify **who has** access to **which AWS resources** and **what actions** they can perform on those resources.

The policy I set up

For this project, I've set up a policy using JSON, as it provides precise control over permissions and is ideal for customizing access rules to meet specific requirements.

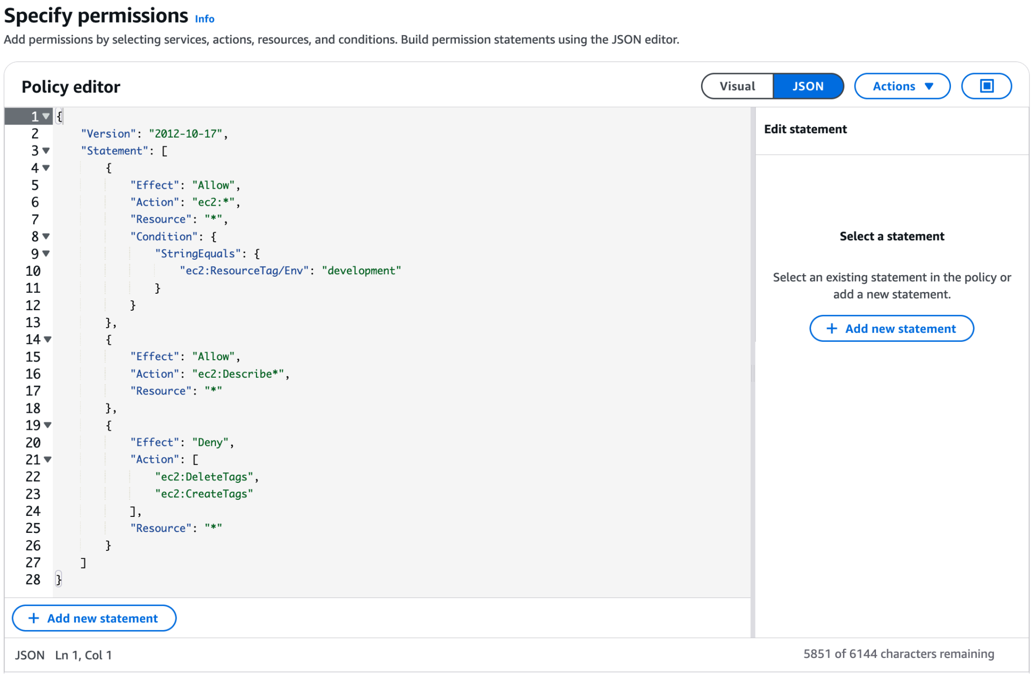
I've created a policy that allows full EC2 access to resources tagged with Env=development, permits describing any EC2 resource for monitoring, and denies creating or deleting tags to ensure tagging policies are enforced and unaltered.

When creating a JSON policy, you have to define its Effect, Action and Resource.

The Effect, Action, and Resource attributes in a JSON policy define: "Effect" specifies Allow or Deny for access. "Action" lists permitted/denied operations like ec2:DescribeInstances. "Resource" specifies applicable resources using ARNs or wildcards.

My JSON Policy

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "ec2:*",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "ec2:ResourceTag/Env": "development"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": "ec2:Describe*",
      "Resource": "*"
    },
    {
      "Effect": "Deny",
      "Action": [
        "ec2:DeleteTags",
        "ec2:CreateTags"
      ],
      "Resource": "*"
    }
  ]
}
```



Specify permissions Info

Add permissions by selecting services, actions, resources, and conditions. Build permission statements using the JSON editor.

Policy editor

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:*",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "ec2:ResourceTag/Env": "development"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "ec2:Describe*",
            "Resource": "*"
        },
        {
            "Effect": "Deny",
            "Action": [
                "ec2:DeleteTags",
                "ec2:CreateTags"
            ],
            "Resource": "*"
        }
    ]
}
```

Edit statement

Select a statement

Select an existing statement in the policy or add a new statement.

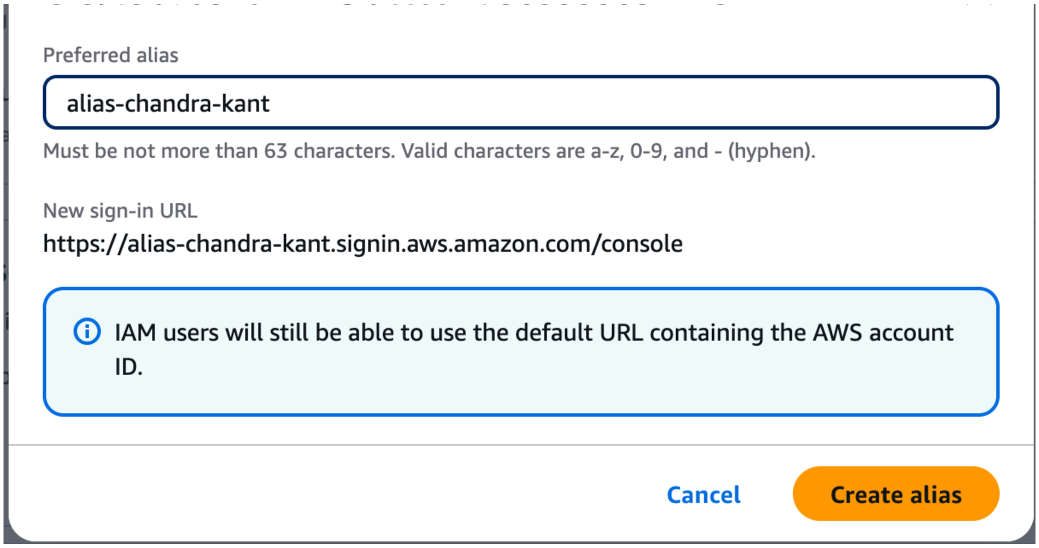
JSON Ln 1, Col 1

5851 of 6144 characters remaining

Account Alias

An account alias is a user-friendly name you can assign to your AWS account ID, replacing the default 12-digit number. It simplifies account identification, especially when managing multiple AWS accounts, making it easier to recognize and remember.

Creating an account alias took me just a few minutes. Now, my new AWS console sign-in URL is https://alias-chandra-kant.signin.aws.amazon.com/console making it easier to access my AWS account.



Preferred alias

alias-chandra-kant

Must be not more than 63 characters. Valid characters are a-z, 0-9, and - (hyphen).

New sign-in URL

https://alias-chandra-kant.signin.aws.amazon.com/console

IAM users will still be able to use the default URL containing the AWS account ID.

IAM Users and User Groups

Users

IAM users are entities within AWS that represent individuals or applications needing access to AWS services and resources. They have unique credentials and permissions, allowing secure control and management of access to AWS infrastructure.

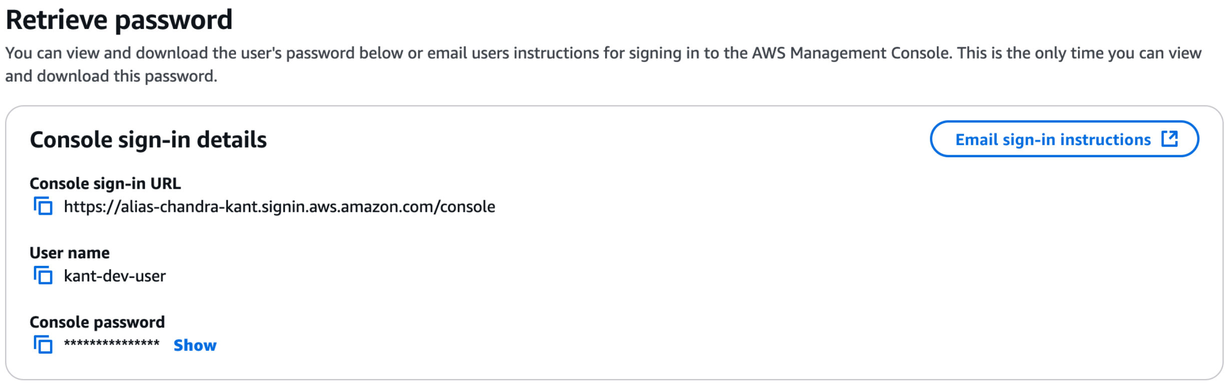
User Groups

IAM user groups are collections of IAM users. They allow you to manage permissions for multiple users at once by attaching policies to the group, simplifying access control and enhancing security management across your AWS environment.
I attached the policy I created to this user group, which means all users in the group now have the specified permissions. This allows them to perform defined actions on AWS resources, ensuring consistent access control across the group.

Logging in as an IAM User

The first way is to send IAM user credentials securely via encrypted email or a password manager. The second way is to use AWS Security Token Service (STS) to provide temporary credentials, ensuring secure, limited-time access.

Once I logged in as my IAM user, I noticed limited access to AWS services and resources in the dashboard. This was because the IAM user was assigned restrictive policies, adhering to the principle of least privilege to enhance security.



Retrieve password

You can view and download the user's password below or email users instructions for signing in to the AWS Management Console. This is the only time you can view and download this password.

Console sign-in details

Console sign-in URL
https://alias-chandra-kant.signin.aws.amazon.com/console

User name
kant-dev-user

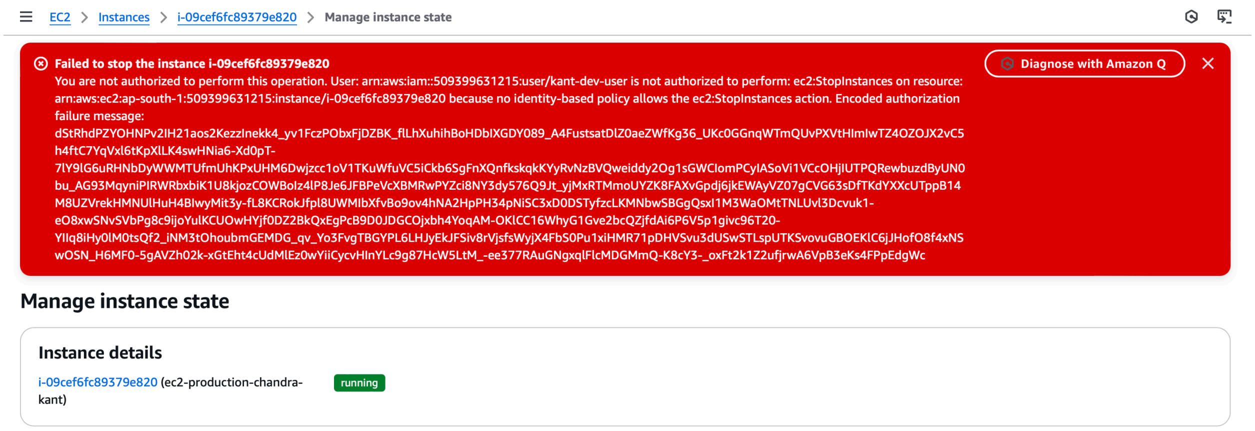
Console password
*************** **Show**

Testing IAM Policies

I tested my JSON IAM policy by attempting to stop two EC2 instances: one tagged as development and the other as production. I successfully stopped the development instance but was denied stopping the production instance.

Stopping the production instance

When I tried to stop the production instance, an error message stopped me. This was because the IAM user lacked the necessary permissions to perform this action.


EC2 > Instances > i-09cef6fc89379e820 > Manage instance state
Diagnose with Amazon Q

Failed to stop the instance i-09cef6fc89379e820
You are not authorized to perform this operation. User: arn:aws:iam::509399631215:user/kant-dev-user is not authorized to perform: ec2:StopInstances on resource: arn:aws:ec2:ap-south-1:509399631215:instance/i-09cef6fc89379e820 because no identity-based policy allows the ec2:StopInstances action. Encoded authorization failure message:

```
dStRhdPZYOHNPv2lH21aos2KezzInekk4_yv1FczPObxFjDZBK_flLhXuhihBoHDbIXGDY089_A4FustsatDlZ0aeZWfKg36_UKc0GGnqWTmQUvPXVtHImIwTZ4OZOJX2vC5
h4ftC7YqVxl6tKpXlLK4swHNia6-Xd0pT-
7lY9lG6uRHNbDyWWMTUfmUhKPxUHM6Dwjzcc1oV1TKuWfuVC5iCkb6SgFnXQnfkskqkKYyRvNzBVQweiddy2Og1sGWClomPCyIASoVi1VCcOHjIUTPQRewbuzdByUN0
bu_AG93MqyniPIRWRbxbiK1U8kjozCOWBoIz4lP8Je6JFBPeVcXBMRwPYZci8NY3dy576Q9Jt_yjMxRTMmoUYZK8FAXvGpdj6jkEWAyVZ07gCVG63sDfTKdYXXcUTppB14
M8UZVrekHMNUlHuH4BlwyMit3y-fL8KCRokJfpl8UWMIbXfvBo9ov4hNA2HpPH34pNiSC3xD0DSTyfzcLKMNbwSBGgQsxI1M3WaOMtTNLUvl3Dcvuk1-
eO8xwSNvSVbPg8c9ijoYulKCUOwHYjf0DZ2BkQxEgPcB9D0JDGCOjxbh4YoqAM-OKlCC16WhyG1Gve2bcQZjfdAi6P6V5p1givc96T20-
YIlq8iHy0lM0tsQf2_iNM3tOhoubmGEMDG_qv_Yo3FvgTBGYPL6LHJyEkJFSiv8rVjsfsWyjX4FbS0Pu1xiHMR71pDHVSvu3dUSwSTLspUTKSvovuGBOEKlC6jJHofO8f4xNS
wOSN_H6MF0-5gAVZh02k-xGtEht4cUdMlEz0wYiiCycvHInYLc9g87HcW5LtM_-ee377RAuGNgxqlFlcMDGMmQ-K8cY3-_oxFt2k1Z2ufjrwA6VpB3eKs4FPpEdgWc
```

Manage instance state

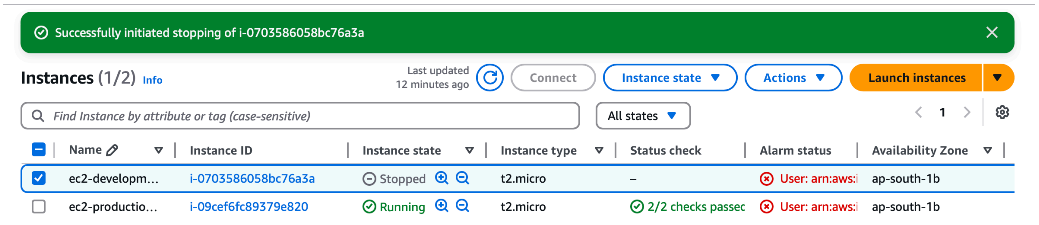
Instance details

i-09cef6fc89379e820 (ec2-production-chandra-kant) running

Testing IAM Policies

Stopping the development instance

I tested my JSON IAM policy by attempting to stop the development EC2 instance and succeeded. This was because the policy allowed all EC2 actions on resources tagged with Env=development, and the instance met this condition.



Successfully initiated stopping of i-0703586058bc76a3a

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
☑	ec2-developm…	i-0703586058bc76a3a	Stopped	t2.micro	–	User: arn:aws:i	ap-south-1b
□	ec2-productio…	i-09cef6fc89379e820	Running	t2.micro	2/2 checks passec	User: arn:aws:i	ap-south-1b