



Cloud Security with AWS IAM

M

Mohammed Dawoud Mota

Policy editor

```
1 * {
2     "Version": "2012-10-17",
3     "Statement": [
4         {
5             "Effect": "Allow",
6             "Action": "ec2:Describe",
7             "Resource": "*",
8         },
9         {
10            "Condition": {
11                "StringEquals": {
12                    "ec2:ResourceTag/Env": "development"
13                }
14            }
15        },
16        {
17            "Effect": "Allow",
18            "Action": "ec2:Describe",
19            "Resource": "*"
20        },
21        {
22            "Effect": "Deny",
23            "Action": "ec2:DeleteTags",
24            "Resource": "*"
25        }
26    ]
27 }
28 }
```

[+ Add new statement](#)

Visual [JSON](#) Actions ▾

Edit statement

Select a statement

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

[+ Add new statement](#)

JSON Ln 29, Col 0 5851 of 6144 characters remaining

Security: 0 Errors: 0 Warnings: 0 Suggestions: 0



Introducing Today's Project!

In this project, I will demonstrate cloud security. I'm doing this project to learn how AWS IAM is used to implement security and access in the cloud!

Tools and concepts

Services I used were EC2 and IAM. Key concepts I learnt about EC2 include Tags, AMI, Instance Types. And for IAM, I learnt about policies, users, groups, and aliases.

Project reflection

This project took me approximately 30 minutes to complete. It was most rewarding to see that the IAM user was able to access the correct resources, while not being allowed to access other resources.

Tags

Tags are like labels that users can attach to their AWS resources for organization. Tagging helps identify resources with the same tag at once. Helps categorize and find resources easily.

The tag I've used on my EC2 instances is called Env. The values I've assigned for my instances are production and development.

▼ Name and tags [Info](#)

Key Info <input type="text" value="Name"/> X	Value Info <input type="text" value="nextwork-dev-mdn"/> X	Resource types Info <input type="button" value="Select resource types"/> ▼ Remove
Instances X		
Key Info <input type="text" value="Env"/> X	Value Info <input type="text" value="development"/> X	Resource types Info <input type="button" value="Select resource types"/> ▼ Remove
Instances X		
Add new tag		
You can add up to 48 more tags.		

IAM Policies

An IAM policy is a rule for who can do what with your AWS resources. It tells a user, group, or role what it can or can not do with AWS resources.

The policy I set up

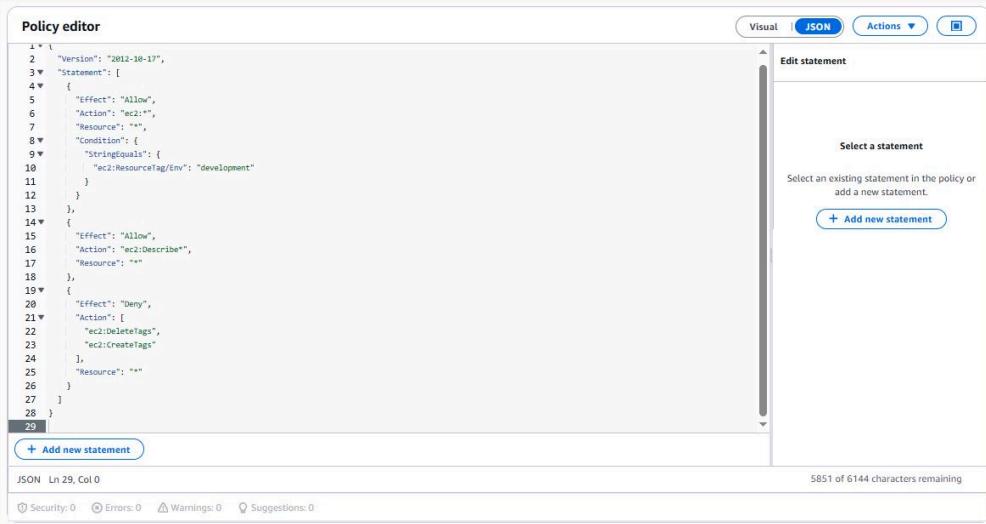
For this project, I've set up a policy using the JSON policy editor.

This IAM policy allows full management of EC2 resources tagged with Env=development, grants read-only access to all EC2 resources via Describe actions, and denies the ability to create or delete tags on any resource.

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

In an IAM JSON policy, Effect defines whether access is allowed or denied, Action specifies which AWS operations are permitted or blocked, and Resource identifies which AWS resources the policy applies to.

My JSON Policy



The screenshot shows a 'Policy editor' interface with a 'JSON' tab selected. The main area displays a JSON policy document with line numbers 1 through 29. The policy includes statements for allowing EC2 actions based on resource tags and denying specific EC2 tag-related actions.

```
1 * {
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "ec2:Describe",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "ec2:ResourceTag/Env": "development"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": "ec2:DeleteTags",
      "Resource": "*"
    },
    {
      "Effect": "Deny",
      "Action": [
        "ec2:DeleteTags",
        "ec2:CreateTags"
      ],
      "Resource": "*"
    }
  ]
}
```

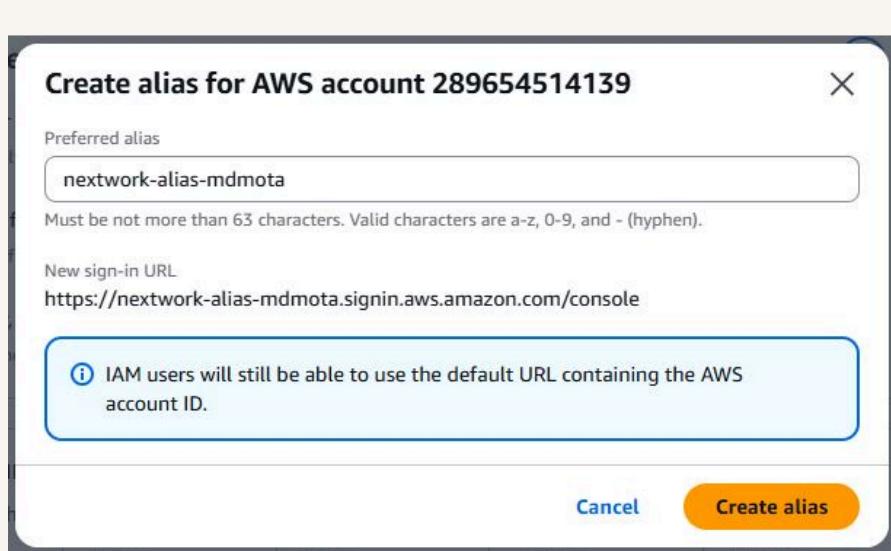
The right side of the interface features a sidebar titled 'Edit statement' with the sub-section 'Select a statement'. It contains the instruction 'Select an existing statement in the policy or add a new statement.' and a blue button labeled '+ Add new statement'.

At the bottom of the editor, there are status indicators: 'JSON Ln 29, Col 0', '5851 of 6144 characters remaining', and a row of icons for security, errors, warnings, and suggestions.

Account Alias

An Account Alias is a friendly name for your AWS account that you can use instead of your account ID to sign in to the AWS Management Console.

Creating an account alias took me 10 seconds. Now, my new AWS console sign-in URL is <https://nextwork-alias-mdmota.signin.aws.amazon.com/console>



IAM Users and User Groups

Users

IAM users are individual identities in AWS created to represent people or applications. Each user has unique credentials, such as a username, password, or access keys, and can be assigned permissions to access and manage AWS resources.

User Groups

IAM user groups are collections of IAM users in AWS that share the same permissions. Instead of assigning policies to each user individually, you attach policies to the group, and all users in that group inherit those permissions.

Attaching a policy to an IAM user group gives all users in that group the permissions defined in the policy, so they can perform the allowed actions on the specified resources while being restricted by any denies in the policy.

Logging in as an IAM User

You can share a new IAM user's sign-in details by giving them the temporary password or by providing the AWS sign-in URL with their username and temporary password so they can log in and set a new password.

Once I logged in as my IAM user, I noticed that AWS is giving a short tutorial on how things work, and that many of the panels on the console show access denied.

The screenshot shows a modal window titled "Retrieve password". It contains the following information:

- Console sign-in details**:
 - Console sign-in URL**: <https://nextwork-alias-mdmota.signin.aws.amazon.com/console>
 - User name**: nextwork-dev-mdmota
 - Console password**: ***** [Show](#)
- Email sign-in instructions** (button)

At the bottom of the modal are three buttons: "Cancel", "Download .csv file", and "Return to users list".

Testing IAM Policies

I tested my JSON IAM policy by clicking stop for both of the EC2 instances.

Stopping the production instance

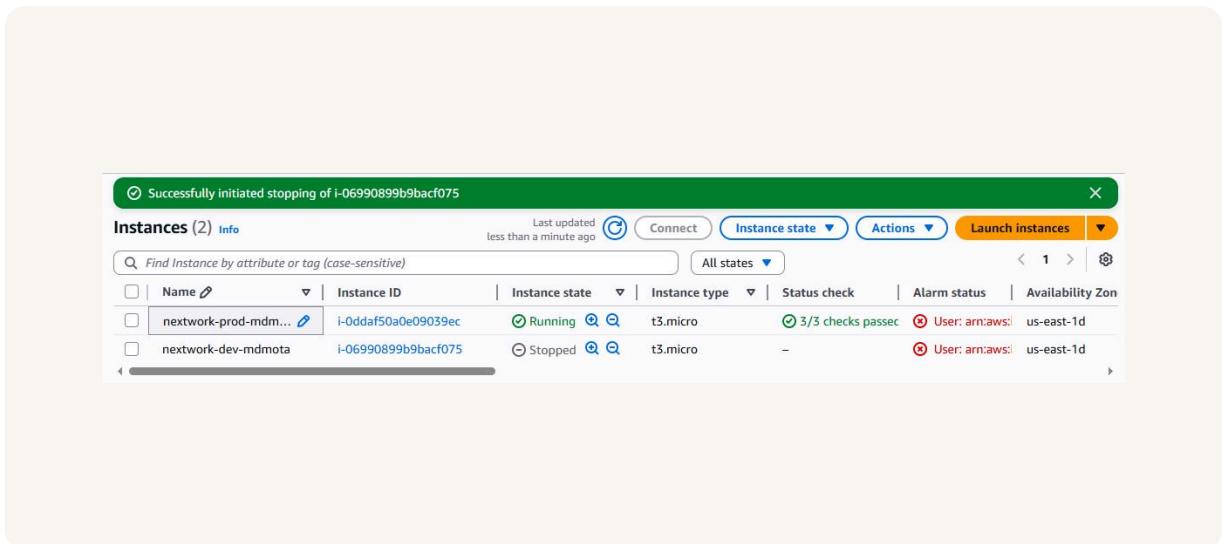
When I tried to stop the production instance, the AWS console gave me an error saying that I am not authorized to perform this operation. This was because this user does not have the permissions for this.



Testing IAM Policies

Stopping the development instance

Next, when I tried to stop the development instance, it allowed me to stop it. This was because the user had the permissions to do that to the development instance.





nextwork.org

The place to learn & showcase your skills

Check out nextwork.org for more projects

