



Cloud Security with AWS IAM



Honesty Dogunro

Policy editor

Visual JSON Actions ▾

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

Edit statement

Select a statement

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

+ Add new statement



Introducing today's project!

What is AWS IAM?

AWS IAM (Identity and Access Management) is a service that manages user access to AWS resources securely. It's useful for providing fine-grained permission control, ensuring only authorized users can access specific services and resources.

How I'm using AWS IAM in this project

I used AWS IAM to create user roles and policies for secure access to resources in today's project. This ensured that team members had the appropriate permissions for their tasks while maintaining security and compliance across the AWS environment.

One thing I didn't expect...

One thing I didn't expect in this project was the complexity of managing permissions with AWS IAM. I used IAM to create user roles and policies, ensuring secure access to resources while navigating unexpected challenges in permission configurations.

This project took me...

This project took me approximately 2 hours to complete. i took breaks



Tags

Tags are labels or keywords that categorize digital content, enhancing its organization and discoverability. They improve user experience by simplifying navigation, aid in SEO, and facilitate content filtering in social media and e-commerce platforms

'The tag I've used on my EC2 instances is called Env. The value I've assigned for my instances are development and production, these are the 2 environment that will be used in this project

▼ Name and tags [Info](#)

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



IAM Policies

IAM Policies are rules that define permissions for AWS resources, controlling who can access and perform actions on those resources to ensure security and compliance.

The policy I set up

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

I've created a policy that restricts EC2 actions based on the environment tag. It allows all actions for resources tagged as "development," permits describing resources, and denies creating or deleting tags across all resources.

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

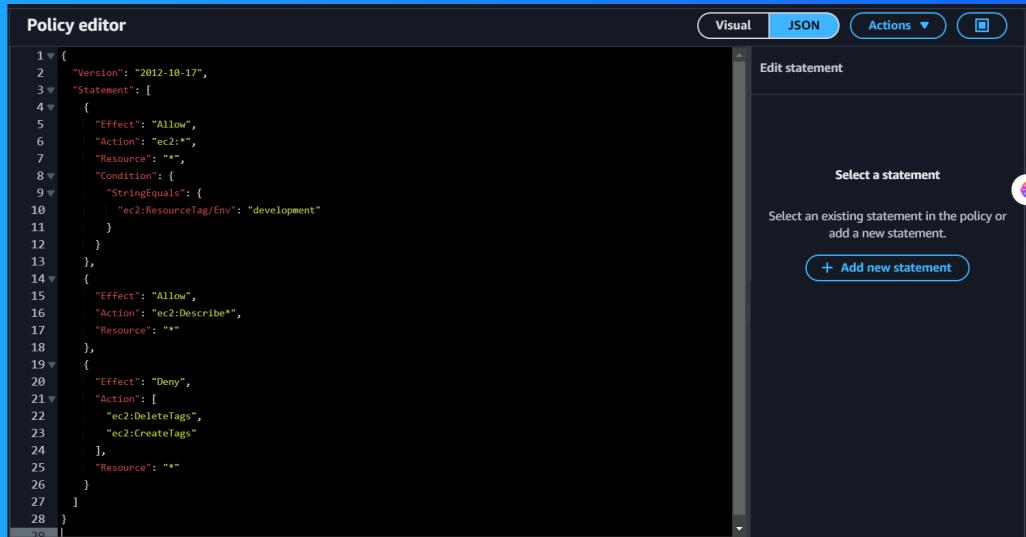
The Effect, Action, and Resource attributes of a JSON policy means: Effect: Determines if access is allowed or denied. Action: Specifies what operations are permitted. Resource: Defines which resources the policy applies to.

H

Honesty Dogunro
NextWork Student

NextWork.org

My JSON Policy



The screenshot shows the AWS IAM Policy editor interface. At the top, there are tabs for "Visual", "JSON" (which is selected), and "Actions". Below the tabs, the policy document is displayed in JSON format:

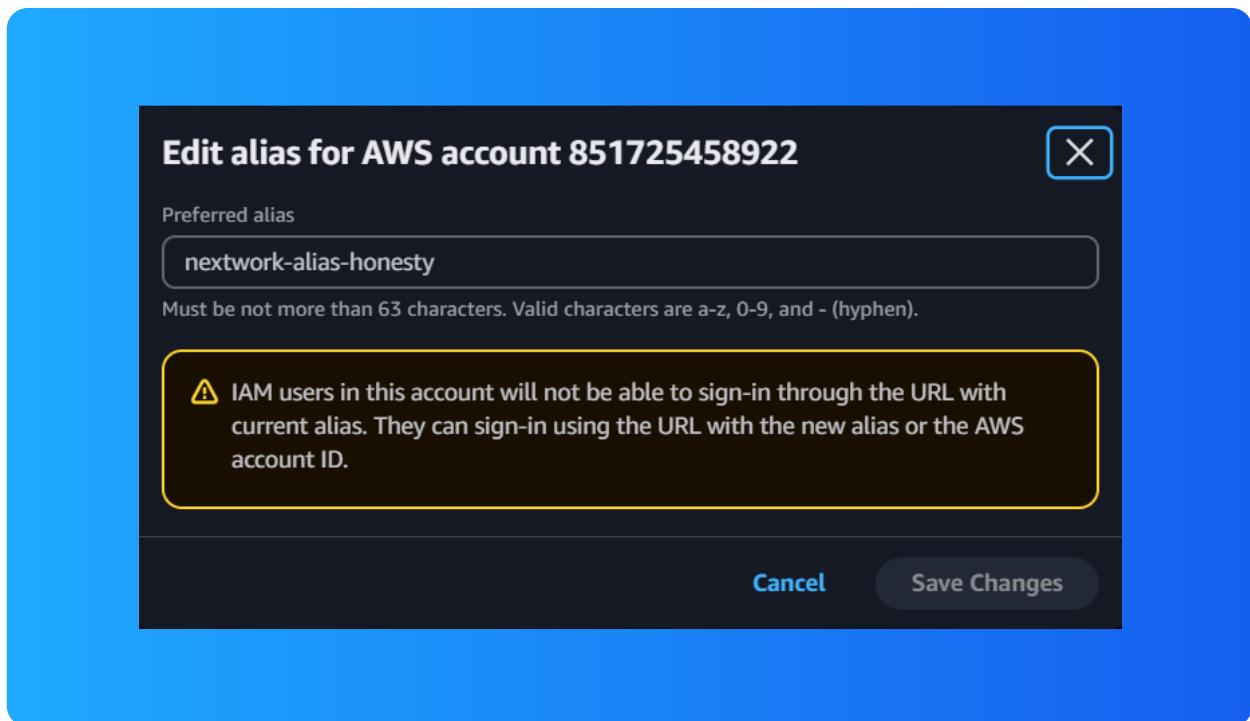
```
1 {  
2     "Version": "2012-10-17",  
3     "Statement": [  
4         {  
5             "Effect": "Allow",  
6             "Action": "ec2:*",  
7             "Resource": "*"  
8         },  
9             {"  
10                 "Condition": {  
11                     "StringEquals": {  
12                         "ec2:ResourceTag/Env": "development"  
13                     }  
14                 }  
15             },  
16             {"  
17                 "Effect": "Allow",  
18                 "Action": "ec2:Describe*",  
19                 "Resource": "*"  
20             },  
21             {"  
22                 "Effect": "Deny",  
23                 "Action": [  
24                     "ec2:DeleteTags",  
25                     "ec2:CreateTags"  
26                 ],  
27                 "Resource": "*"  
28             }  
29         ]  
30     }  
31 }
```

To the right of the JSON pane, there is a sidebar titled "Edit statement" with the sub-instruction "Select a statement". It also includes a button labeled "+ Add new statement".

Account Alias

An account alias is a secondary identifier that allows users to send and receive messages without revealing their primary email, enhancing privacy and managing communications.

Creating an account alias took me less than a minute Now, my new AWS console sign-in URL is <https://nextwork-alias-honesty.signin.aws.amazon.com/console>





IAM Users and User Groups

Users

IAM users are individual identities in AWS IAM that allow secure access to AWS services, with specific permissions tailored for tasks and resources.

User Groups

IAM user groups are collections of IAM users in AWS that simplify permissions management. By assigning policies to a group, all members inherit those permissions, making it easier to manage access for multiple users based on their roles.

I attached the policy I created to this user group, which means all group members will inherit the defined permissions, simplifying access management.



Logging in as an IAM User

The first way is to send the user an email with their sign-in URL, username, and a temporary password. The second way is to provide them the details verbally or through a secure messaging method, ensuring they change their password upon first sign-in

I HAVE LIMITED ACCESS

The screenshot shows the AWS Management Console interface for creating a new IAM user. The top navigation bar includes the AWS logo, search bar, and various global settings. The main content area is titled 'Create user' under the 'IAM > Users' section. A green success message box at the top states 'User created successfully' and provides instructions to view and download the user's password and email instructions for signing in to the AWS Management Console. Below this, a progress bar indicates 'Step 4: Retrieve password' is the current step. To the left, a sidebar lists the steps: Step 1 (Specify user details), Step 2 (Set permissions), Step 3 (Review and create), and Step 4 (Retrieve password), with Step 4 being the active one. The central panel displays 'Console sign-in details' for the newly created user 'nextwork-dev-honesty'. It shows the 'Console sign-in URL' as <https://nextwork-alias-honesty.sigin.aws.amazon.com/console>, the 'User name' as 'nextwork-dev-honesty', and the 'Console password' as a masked string. There is also a 'Show' link next to the password field. On the right side of the panel, there is a 'Email sign-in instructions' button. At the bottom of the page are 'Cancel', 'Download .csv file', and 'Return to users list' buttons.

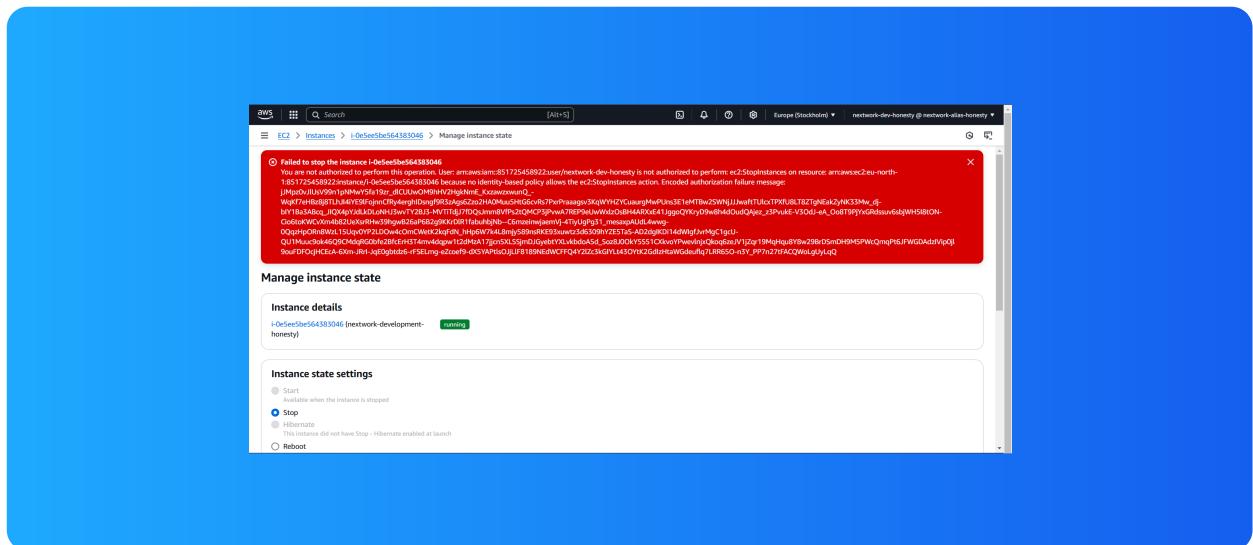


Testing IAM Policies

I tested my JSON IAM policy by attaching it to two EC2 instances. I performed actions like launching and stopping the instances to verify that the policy properly granted or restricted access to the specified AWS resources.

Stopping the production instance

When I tried to stop the production instance, I received an authorization error. This was because my IAM user did not have the necessary permissions to perform the `ec2:StopInstances` action on that resource.

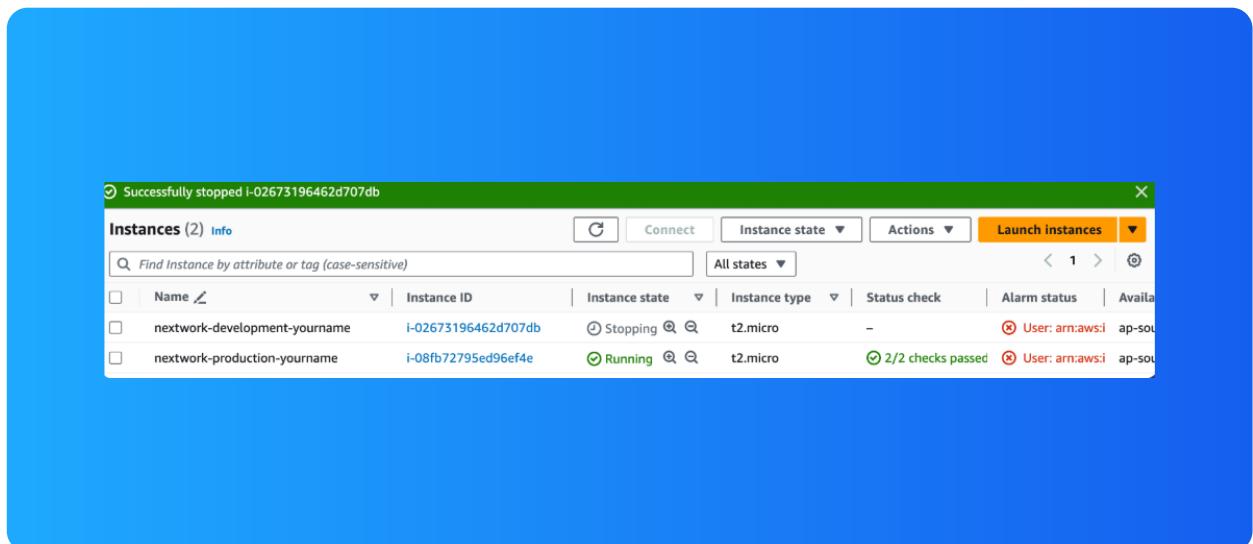




Testing IAM Policies

Stopping the development instance

Next, when I tried to stop the development instance it stopped. This was because of the policy that was created..





NextWork.org

Everyone should be in a job they love.

Check out nextwork.org for
more projects

