



Cloud Security with AWS IAM

J

Joseph Cole-Showers

Policy editor

```
1▼ {  
2  "Version": "2012-10-17",  
3▼ "Statement": [  
4▼   {  
5    "Effect": "Allow",  
6    "Action": "ec2:*",  
7    "Resource": "*",  
8▼     "Condition": {  
9▼       "StringEquals": {  
10      "ec2:ResourceTag/Env": "development"  
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 }
```



Introducing today's project!

What is AWS IAM?

Services I used were Amazon EC2, and AWS IAM. Key concepts I learnt include IAM users, role, user groups, policies, account alias, how to launch an instance, how to tag an instance, how to log in as another user.

How I'm using AWS IAM in this project

This project took me approximately 1hr 20 mins. The most challenging part was understanding the IAM policy written in JSON and it contained multiple statements. It was most rewarding to see "permission denied" when an intern tried to delete prod.

One thing I didn't expect...

I chose to do this project today because IAM and Cloud security are key skills in Cloud computing.



Tags

Tags are organizational tools that let us to label our resources. They are helpful for grouping resources, applying policies for all resources with the same tag and cost allocation.

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

Launch an instance Info

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

Name and tags Info

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



IAM Policies

IAM Policies are like roles that determine who can do what in an AWS account. I'm using policies to control who has access to the production environment instance.

The policy I set up

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

I've created a policy that allows the policy holder to have permission to do anything they want with any instance tagged with dev. They can also see information related for any instance, but they are denied access to delete/create any instance

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 whether or not the policy is allowing or denying action (i.e Effect) What the policy hold can or cannot do (i.e Action) and the specific resources the policy relates to (resources).

My JSON Policy

Policy editor

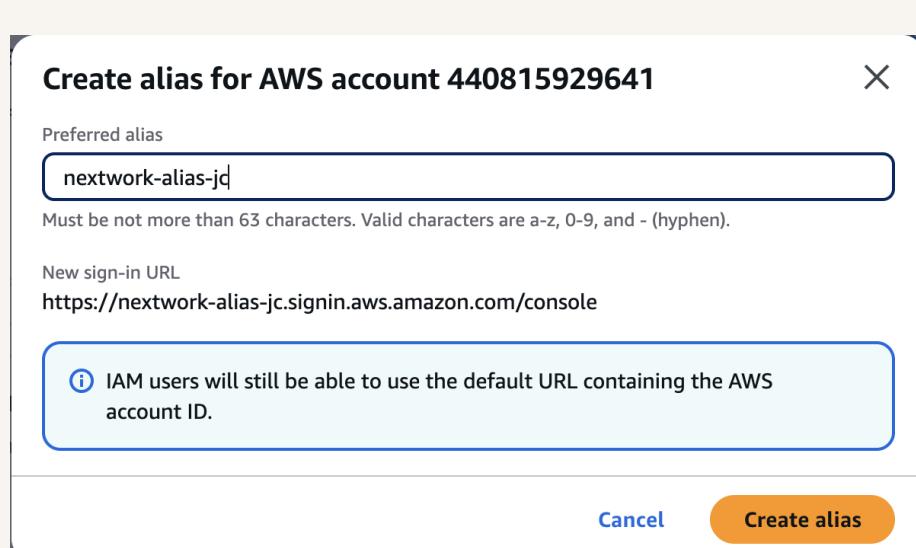
```
1▼ {
2    "Version": "2012-10-17",
3▼   "Statement": [
4▼     {
5         "Effect": "Allow",
6         "Action": "ec2:*",
7         "Resource": "*",
8▼         "Condition": {
9▼             "StringEquals": {
10                "ec2:ResourceTag/Env": "development"
11            }
12        }
13    },
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 }
```



Account Alias

An account alias is simply a nickname for an AWS account. Instead of a log account ID, we can reference the account alias instead.

Creating an account alias took me 30 seconds. Now, my new AWS console sign-in URL uses the alias I created instead of my account ID.





IAM Users and User Groups

Users

IAM users are people or entities that have access / login to your aws account.

User Groups

IAM user groups are like folders that collect IAM users so that you can apply permission settings at the group level.

I attached the policy I created to this user group, which means any user created inside this group will automatically get the permissions attached to the policy



Logging in as an IAM User

The first way is to email sign in instructions to the user, and the second way is to download the CSV file with the sign in details inside.

Once I logged in as my IAM user, I noticed that the user (intern) is already denied access to items under AWS dashboard. This was because I only setup permissions to our development EC2 instance.

The screenshot shows the 'Create user' process in the AWS Management Console. The user has just completed Step 4, 'Retrieve password'. A green success message at the top states: 'User created successfully. You can view and download the user's password and email instructions for signing in to the AWS Management Console.' Below this, 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 current active step. The main content area is titled 'Retrieve password' and contains 'Console sign-in details'. It shows the 'Console sign-in URL' as [https://nextwork-alia...console](https://nextwork-alias-jc.signin.aws.amazon.com/console). Under 'User name', it shows 'nextwork-dev-jc'. Under 'Console password', it shows a masked password followed by a 'Show' link. There are three buttons at the bottom right: 'Cancel', 'Download .csv file' (highlighted in blue), and 'Return to users list'.



Testing IAM Policies

I tested my JSON IAM policy by attempting to stop both the development and production instances.

Stopping the production instance

When I tried to stop the production instance I got an error. This was because the production instance is tagged with the production label which is outside of the scope of the permission policy. Interns are only allowed to do work in the dev instance.

 Failed to stop the instance i-061d4573f97cb4416
You are not authorized to perform this operation. User: arn:aws:iam::440815929641:user/nextwork-dev-jc is not authorized to perform: ec2:StopInstances on resource: arn:aws:ec2:eu-west-2:440815929641:instance/i-061d4573f97cb4416 because no identity-based policy allows the ec2:StopInstances action. Encoded authorization failure message: CPi9baZdrf5iFxSkLlqS1suFuWDFj_lpx1X2QcmhYz05bC8-GjaKEdsjzhqjN_rJJu-dV1jNi434h-YWBbCin-hcAGFTf9Epdd5S1vczjlLVmcPKz6kxLjZzkMgw-VqG9lxmtg8Ubcqmq2x-fa5DPx4tIN9_ellxDgaXQgPiHCCzIM9Kz8HW9T-q-
7ywgAX7D1RAgbzqU4dpAaTaZtDB_4h7KyXLrYshCgSrQgv85RyOYTBJ8xUtpT01D7ogvSV17k_00s72Bh3u07ypYrNmXHGxallkbDUOgm_EhJaVoOwU6UFC70fHONjFOa2905zrF_B9z7ap1pZMsPzr6CC0JUXN3N03R_MgZTsxC8TLAPAg7Wkczfrq5AXSkBOHPjQru9eAauKIT39q3TDB_2RghC1plekXTA2zCzJNB4hCDymI3g9Q08CfOjkIDzoyXURAZF1RondzTUxew-BBawlk25qgXkfQkv2fLcIWFGszUK-6wx5Kli6Xd1GVAl8xCeSYI-EB7h4qTlbznlBPHMavG-acxxNIIdK15TYJve3T1IVQ3tReffhnNohvN3SSlw-IQETYVYXQZl8sqE2AEFk6gJYClqbzcWQejodPxJuTo0_3lIKOOnNlxipNmFE2y7h5yE5AB31niAT_ROLGloKCbW7FIN40658TOoFdz78PCovvhA1pPmLoYbfz6mEPqCGShVWv9-TIAk7dgWt6e7RW-al02vYctp-30hrXy6QmxvYJGXCWwpkaSjsfAkntQ7dYcDiGEL45VbcL
SnxsdxxZDaHv7mk101Rc5EL5ocFckuXB6LglPtNIUSHk4vfhlbT4xpq3mMzsorZACzboH4vpr39vs



Testing IAM Policies

Stopping the development instance

Next, when I tried to stop the development instance I successfully saw the instance state changed to stopping and then stopped. This was because the permission policy I set allows the user in the dev group to stop instances.

The screenshot shows the AWS CloudWatch Instances console. A green success message at the top reads: "Successfully initiated stopping of i-042b16c6b44a21137". Below it, the "Instances (1/2) Info" table displays two instances:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
nextwork-pro...	i-061d4573f97cb4416	Running	t2.micro	2/2 checks passed	User: arn:aws:...	eu-west-2a
nextwork-dev-jc	i-042b16c6b44a21137	Stopping	t2.micro	2/2 checks passed	User: arn:aws:...	eu-west-2a



NextWork.org

Everyone should be in a job they love.

Check out nextwork.org for
more projects

