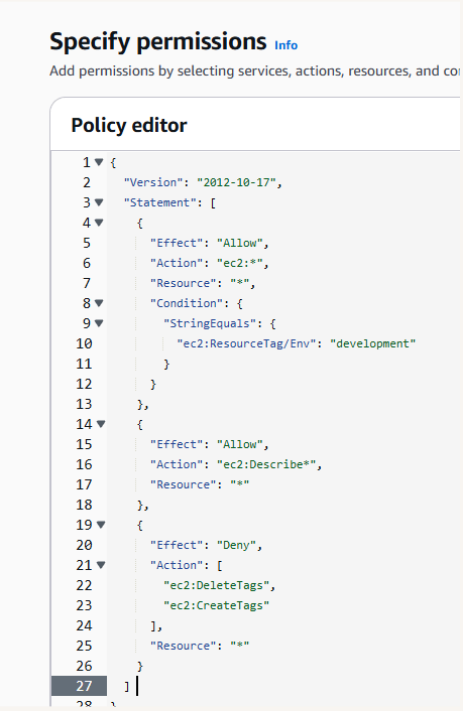
nextwork.org

Cloud Security with AWS IAM

Louis Moyo



L
Louis Moyo
NextWork Student

Introducing Today's Project!

In this project, I will demonstrate identity and access control on AWS by launching an EC2 instance and using IAM policies, users, and groups to enforce least-privilege access. I'm doing this to learn how authentication and authorisation work together in practice.

Tools and concepts

Services I used were Amazon EC2 and AWS IAM. Key concepts I learnt include launching and tagging EC2 instances, creating IAM policies with conditions, using user groups and users for access control, and securing logins with an account alias.

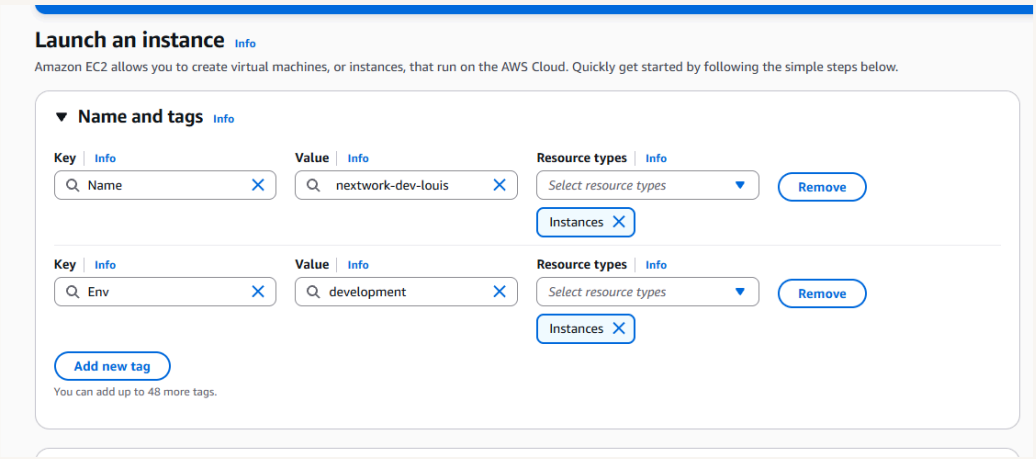
Project reflection

This project took me approximately 1 hour. The most challenging part was understanding how IAM policies use conditions with tags. It was most rewarding to see the intern user successfully access only the development instance while being blocked from production.

Tags

Tags are key–value pairs that I can attach to AWS resources, such as EC2 instances. They are useful for organising and identifying resources, tracking costs, applying automation rules, and controlling access through IAM policies. Tags make it easier to manage resources at scale by grouping them by project, environment, or owner.

The tags I've used on my EC2 instances are called Name and Env. For the Name tag, the value I assigned is nextwork-dev-louis. For the Env tag, the value I assigned is development. These tags help me easily identify the instance by project and environment.



Louis Moyo
NextWork Student

IAM Policies

IAM Policies are rules that define what actions users or groups can perform on AWS resources.

The policy I set up

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

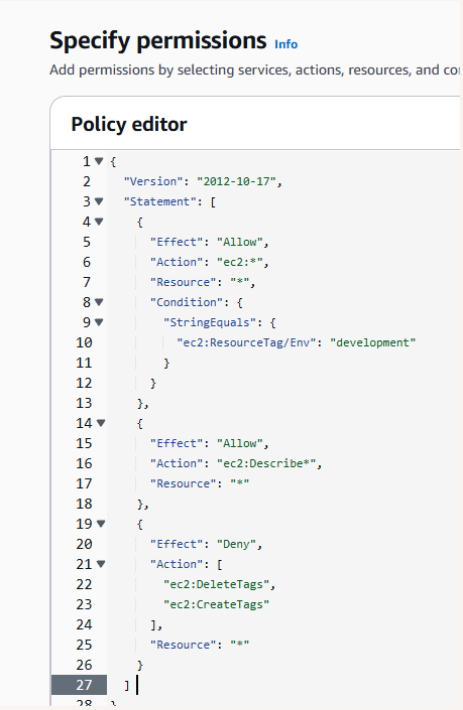
I've created a policy that lets the intern perform all EC2 actions on instances tagged with Env=development, while still allowing them to view (describe) all EC2 resources, and explicitly denying them the ability to create or delete tags.

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

Effect tells AWS whether the statement allows or denies the action. Action specifies which AWS operations are being controlled (e.g., ec2:StartInstances). Resource defines which resources the action applies to (e.g., a specific instance or * for all).

Louis Moyo
NextWork Student

My JSON Policy

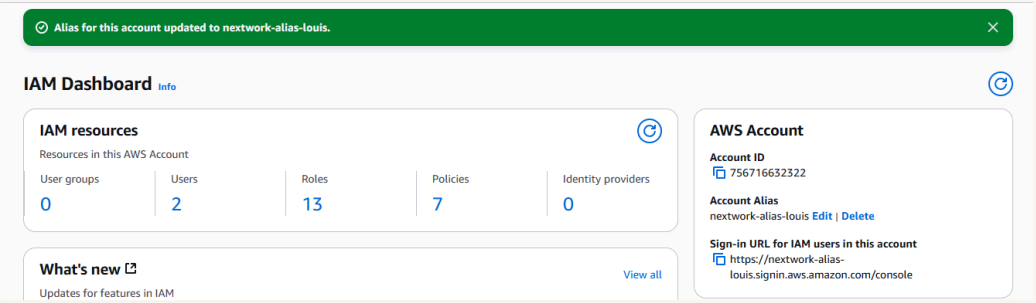


Louis Moyo
NextWork Student

Account Alias

An account alias is a custom name that replaces the 12-digit AWS account ID in the login URL, making it easier for users to sign in.

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



IAM Users and User Groups

Users

IAM users are individual identities in AWS with their own credentials. They represent people or applications that need access to AWS resources.

User Groups

IAM user groups are collections of IAM users that share the same permissions. By attaching policies to a group, all users in that group automatically inherit those permissions.

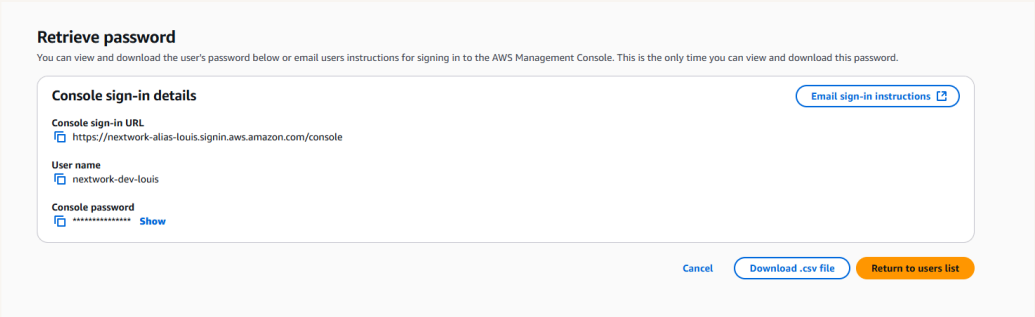
I attached the policy I created to this user group, which means every intern in the group can access and manage only the development EC2 instance while being restricted from production resources.

Louis Moyo
NextWork Student

Logging in as an IAM User

The first way is to download the user's sign-in credentials as a .csv file and share it securely. The second way is to copy the sign-in link, username, and password from the console and provide them securely to the new user.

Once I logged in as my IAM user, I noticed that several dashboard panels showed "Access denied." This was because the user only has limited permissions from the policy I created, which restricts access to everything except the development EC2 instance.



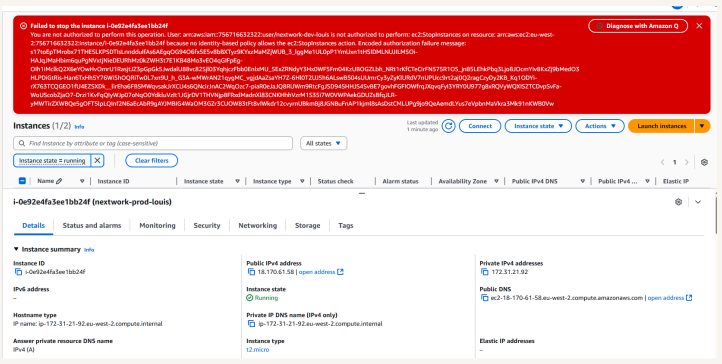
Louis Moyo
NextWork Student

Testing IAM Policies

I tested my JSON IAM policy by attempting to stop both the production and development EC2 instances.

Stopping the production instance

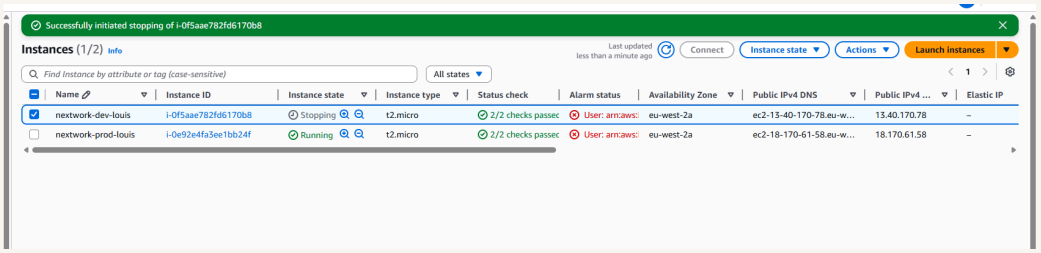
When I tried to stop the production instance, the action was denied. This was because the IAM policy I created only allows control of instances tagged with Env=development, so the user does not have permission to stop the production instance.



Testing IAM Policies

Stopping the development instance

Next, when I tried to stop the development instance, the action succeeded. This was because my IAM policy explicitly allows EC2 actions on resources tagged with Env=development.



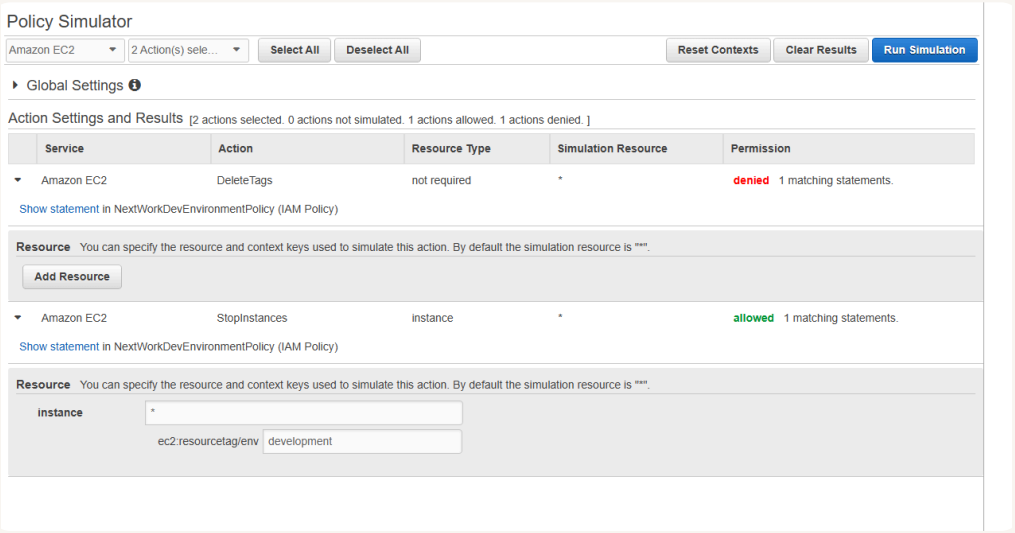
Louis Moyo
NextWork Student

The IAM Policy Simulator

The IAM Policy Simulator is a tool that lets you test and validate IAM policies without affecting real AWS resources. It's useful for checking whether a user will be allowed or denied access to specific actions before trying them in production, saving time and avoiding disruption.

How I used the simulator

I set up a simulation for the StopInstances action on the development instance. The results were denied by default because the instance had no tags, so I manually added the Env=development tag. After that, the simulator confirmed access was allowed.



The place to learn & showcase your skills

Check out nextwork.org for more projects