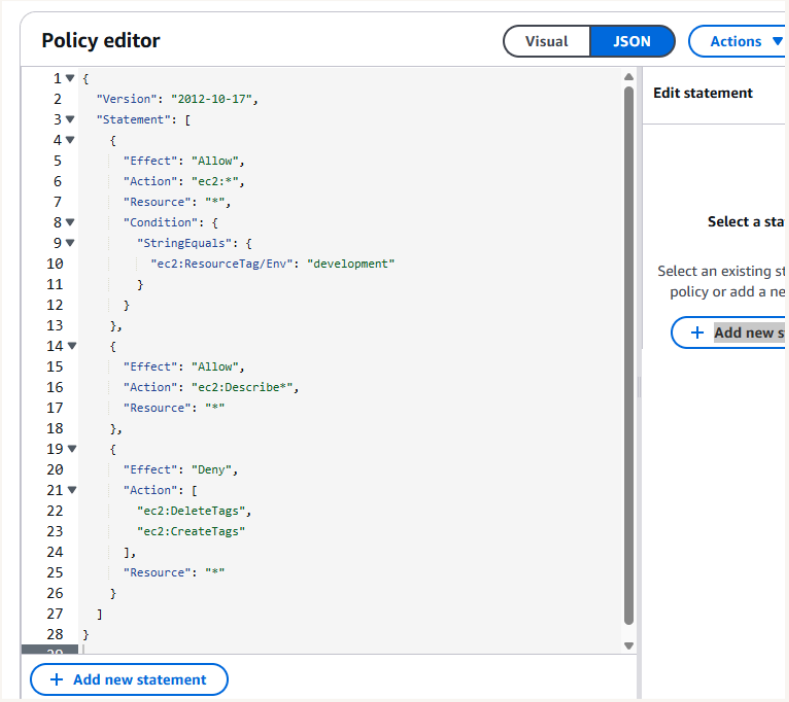
Cloud Security with AWS IAM

Seth Sekyere



Seth Sekyere

Introducing Today's Project!

In this project, I will demonstrate how to use AWS Identity and Access Management (IAM) to control access to cloud resources. The goal is to launch an EC2 instance and manage who can access it by creating IAM users, groups, roles, and policies.

Tools and concepts

I learned key AWS IAM concepts like users, groups, policies, and account aliases, and used EC2 to test permissions based on tags—gaining hands-on experience managing secure, role-based access.

Project reflection

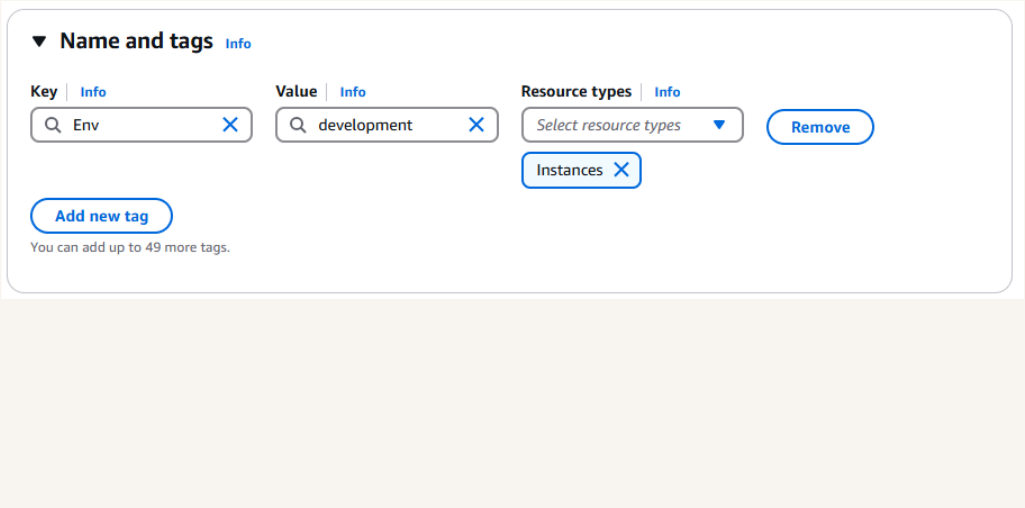
This project took me approximately 15 minutes. The most challenging part was navigating the IAM dashboard and testing policy effects. It was most rewarding to see how user permissions worked exactly as expected!

Seth Sekyere

Tags

Tags are key-value pairs attached to AWS resources. They help organize and manage resources by enabling easy filtering, cost allocation, automation, and policy application, making resource tracking and management more efficient.

The tag I've used on my EC2 instances is Env. I've assigned production to the first instance for the live environment and development to the second for testing. This helps efficiently manage and organize resources across different environments.



Seth Sekyere

IAM Policies

IAM Policies are rules that define permissions for AWS users, groups, or roles. They specify what actions can be performed on which resources, helping manage access and control over AWS services securely.

The policy I set up

For this project, I've set up a policy using JSON editor. This method allows for more flexibility in defining specific permissions for users, groups, or roles in AWS.

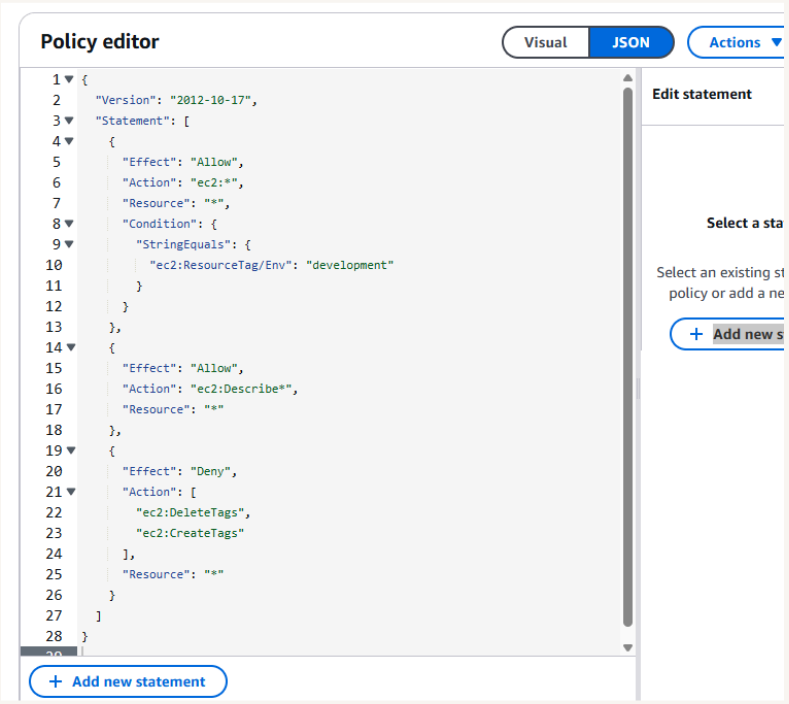
I've created a policy that allows the intern to perform all EC2 actions on instances tagged "Env = development," while denying the ability to create or delete tags on any instance. This ensures they can manage the development environment safely.

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

The Effect, Action, and Resource attributes of a JSON policy define permissions. Effect specifies "Allow" or "Deny," Action lists permitted operations (e.g., starting EC2), and Resource identifies which resources the policy applies to.

Seth Sekyere

My JSON Policy

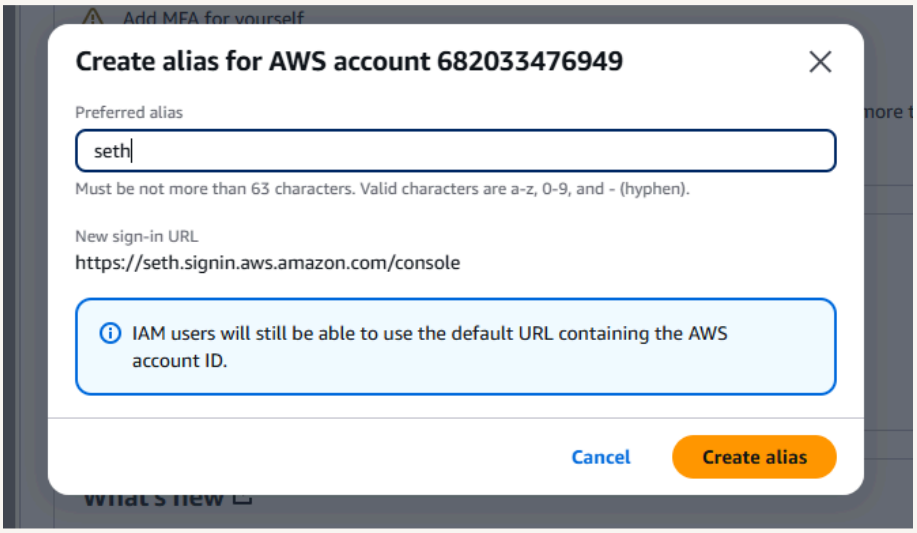


Seth Sekyere

Account Alias

An account alias is a custom name that replaces your AWS account ID in the sign-in URL, making it easier to remember and share with team members for logging into the AWS Management Console.

Creating an account alias took me less than a minute. Now, my new AWS console sign-in URL is easier to remember and share, using the alias instead of the long account ID.



Seth Sekyere

IAM Users and User Groups

Users

IAM users are individual accounts created in AWS to allow specific people or applications to access resources in your AWS account. Each user has unique credentials and permissions, enabling secure, controlled access to AWS services and resources.

User Groups

IAM user groups are collections of IAM users. They let you manage permissions for multiple users at once by attaching policies to the group, making user management easier and ensuring consistent access control across similar users.

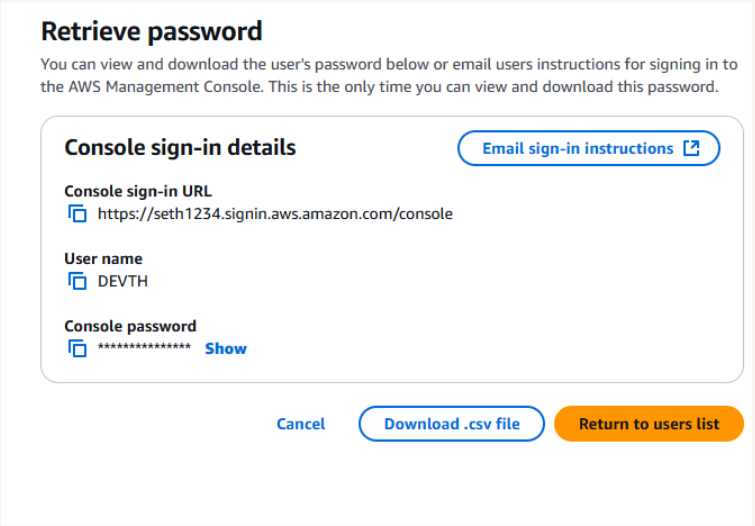
I attached the policy I created to this user group, which means all users in the group, including the intern, will have the permissions defined in the policy, granting them access to the development EC2 instance but not the production instance.

Seth Sekyere

Logging in as an IAM User

The first way is by sending the sign-in details directly to the user via email. The second way is by sharing the user's sign-in URL along with their username and password through a secure communication channel.

Once I logged in as my IAM user, I noticed that some dashboard panels showed "Access denied." This was because the user only has permissions for EC2 instances tagged with "Env = development" and can't access broader AWS services.



Seth Sekyere

Testing IAM Policies

I tested my JSON IAM policy by trying to stop both EC2 instances—access was allowed for the development instance and denied for the production instance, confirming the policy worked as expected.

Stopping the production instance

When I tried to stop the production instance, I saw "Access denied" because my IAM policy only allows actions on instances tagged with Env = development.

Seth Sekyere

Testing IAM Policies

Stopping the development instance

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