1T33: Cloud Architecture

Lab- 6A: Implementing and Managing IAM Roles in AWS

Objectives:

- -Understand the importance and functionality of Identity and Access Management (IAM) roles in AWS.
- Learn how to create and configure IAM roles to assign granular permissions for different AWS services and resources.
- Ensure secure and appropriate access control for AWS resources using IAM roles.

Outcomes:

1. Creation of an IAM Role: You will create a fully functional IAM role with specific permissions for AWS services.
2. Role Assignment: You will assign the IAM role to an EC2 instance, Lambda function, or another AWS service to manage permissions securely.
3. Controlled Access: You will implement fine-grained access control using managed policies and custom policies.
4. Enhanced Security: You will ensure security by following the least privilege principle, limiting access to only the resources necessary for tasks.

System Requirements:

- AWS Account: You must have an active AWS account with access to the IAM management console.
- User Permissions: Ensure you have sufficient permissions to create and assign IAM roles within the AWS Management Console.
- Browser: A modern web browser such as Google Chrome, Mozilla Firefox, or Microsoft Edge.
- Internet Connection: A stable internet connection to interact with AWS services.

1T33: Cloud Architecture

Step-by-Step Procedure:

Step 1: Sign in to the AWS Management Console

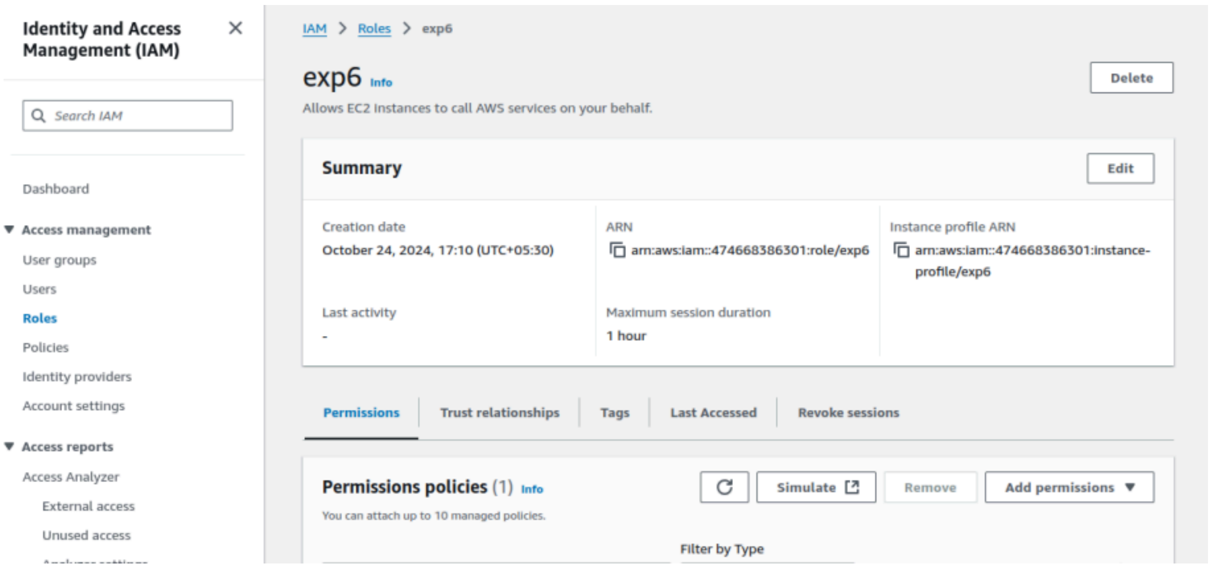
1. Open your web browser and navigate to the [AWS Management Console](https://aws.amazon.com/console/).
2. Sign in with your credentials (AWS root account or IAM user with admin privileges).

Step 2: Access IAM Service

1. From the AWS Management Console, in the search bar, type `IAM` and select IAM from the drop-down list.
2. You will be taken to the IAM dashboard.

Step 3: Create an IAM Role

1. In the IAM dashboard, on the left-hand menu, click on Roles.
2. Click the Create role button.
3. Select trusted entity type:
   - Choose the service that will use this role (e.g., EC2, Lambda, etc.).
   - Alternatively, select Another AWS account if you're creating a cross-account role.
4. Click Next to proceed to permission settings.



exp6 Info

Allows EC2 instances to call AWS services on your behalf.

Summary

Creation date: October 24, 2024, 17:10 (UTC+05:30)

ARN: arn:aws:iam::474668386301:role/exp6

Instance profile ARN: arn:aws:iam::474668386301:instance-profile/exp6

Last activity: -

Maximum session duration: 1 hour

Permissions policies (1) Info

You can attach up to 10 managed policies.

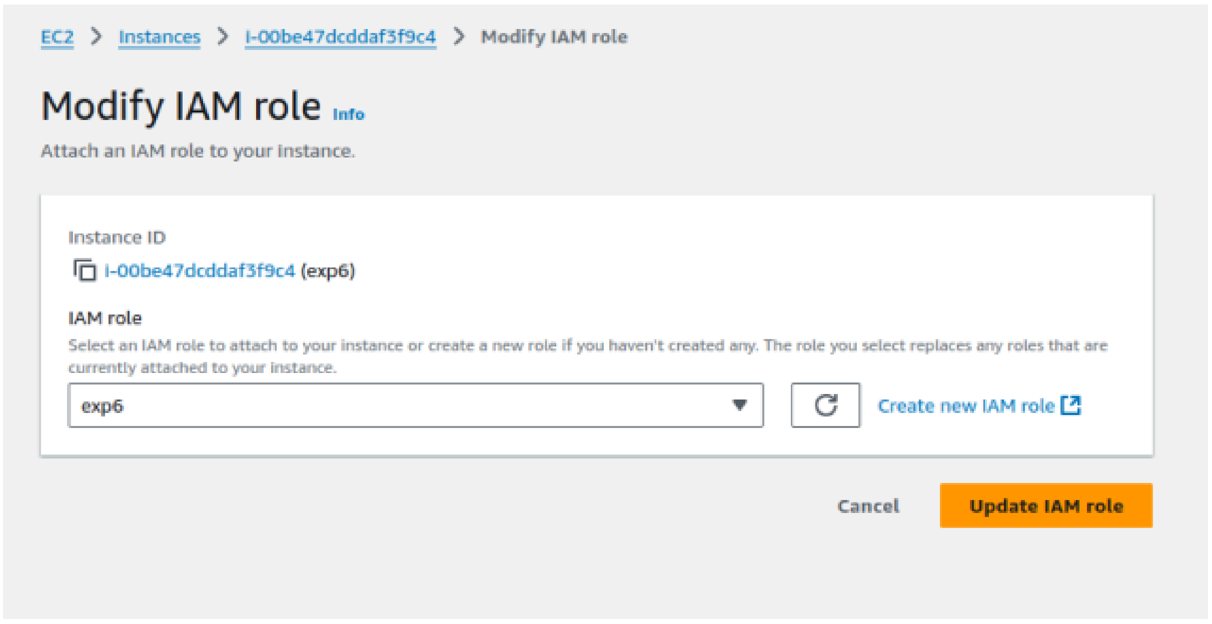
Step 4: Attach Permissions

1. Attach a policy to the role by selecting from AWS-managed policies or custom policies.

 - For example, for an EC2 role with S3 access, choose the AmazonS3ReadOnlyAccess policy.

2. You can create and attach a custom policy if your use case requires fine-grained control.

3. Click Next to add tags (optional), then click Next again to review the role.



Modify IAM role Info

Attach an IAM role to your instance.

Instance ID

i-00be47dcddaf3f9c4 (exp6)

IAM role

Select an IAM role to attach to your instance or create a new role if you haven't created any. The role you select replaces any roles that are currently attached to your instance.

exp6

Step 5: Define Role Details

1T33: Cloud Architecture

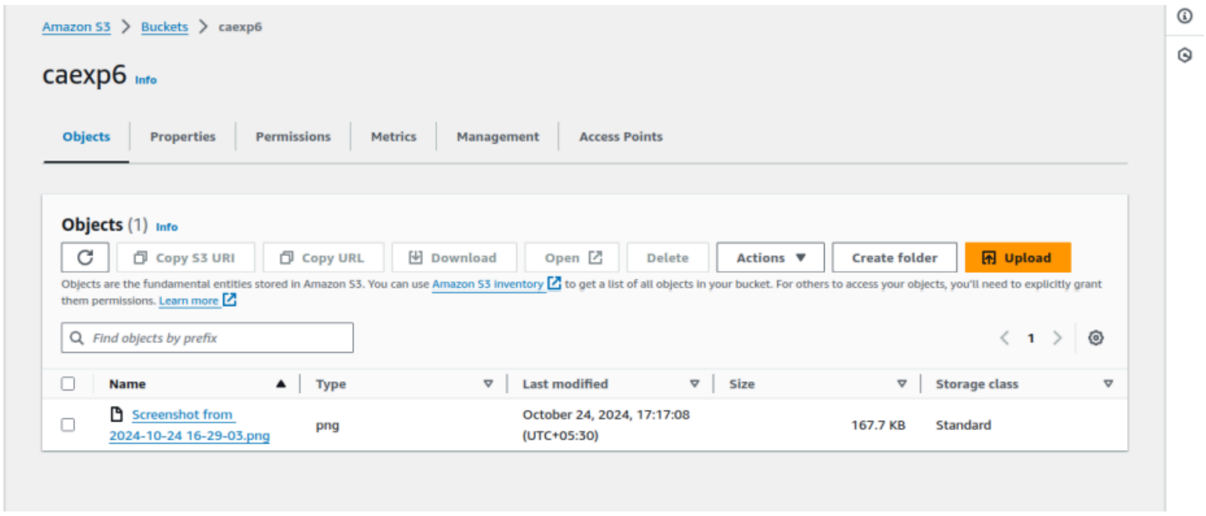
1. Give the role a name that reflects its use case, e.g., EC2-S3-Access-Role.
2. (Optional) Add a description to explain the role's purpose.
3. Review your selections and click Create role.

Step 6: Assign the IAM Role to an AWS Service

1. For example, to assign the role to an EC2 instance:
   - Navigate to the EC2 Dashboard.
   - Select the instance to which you want to assign the role.
   - Click Actions > Security > Modify IAM Role.
   - Select the IAM role you created from the drop-down and click Update IAM role.

Step 7: Validate the Role

1. Test the role by attempting to perform an action using the assigned permissions (e.g., accessing S3 from the EC2 instance).
2. Ensure that actions outside the role's permissions result in access being denied, following the least privilege principle.

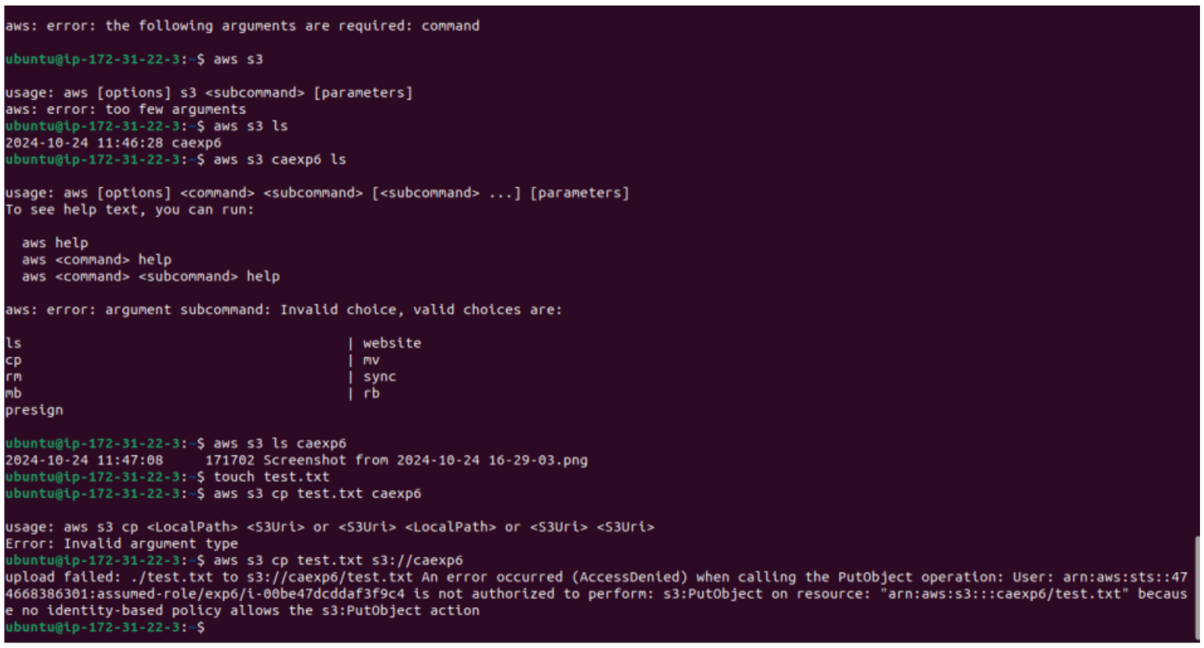


Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

Name	Type	Last modified	Size	Storage class
Screenshot from 2024-10-24 16-29-03.png	png	October 24, 2024, 17:17:08 (UTC+05:30)	167.7 KB	Standard

Step 8: Monitor and Review IAM Role Usage

1. Regularly review the role's activity in CloudTrail to ensure it is being used securely.
2. Update policies as needed to reflect changing access requirements.



```
aws: error: the following arguments are required: command

ubuntu@ip-172-31-22-3:~$ aws s3

usage: aws [options] s3 <subcommand> [parameters]
aws: error: too few arguments
ubuntu@ip-172-31-22-3:~$ aws s3 ls
2024-10-24 11:46:28 caexp6
ubuntu@ip-172-31-22-3:~$ aws s3 caexp6 ls

usage: aws [options] <command> <subcommand> [<subcommand> ...] [parameters]
To see help text, you can run:

  aws help
  aws <command> help
  aws <command> <subcommand> help

aws: error: argument subcommand: Invalid choice, valid choices are:

ls                                       | website
cp                                       | mv
rm                                       | sync
mb                                       | rb
presign

ubuntu@ip-172-31-22-3:~$ aws s3 ls caexp6
2024-10-24 11:47:08     171702 Screenshot from 2024-10-24 16-29-03.png
ubuntu@ip-172-31-22-3:~$ touch test.txt
ubuntu@ip-172-31-22-3:~$ aws s3 cp test.txt caexp6

usage: aws s3 cp <LocalPath> <S3Uri> or <S3Uri> <LocalPath> or <S3Uri> <S3Uri>
Error: Invalid argument type
ubuntu@ip-172-31-22-3:~$ aws s3 cp test.txt s3://caexp6
upload failed: ./test.txt to s3://caexp6/test.txt An error occurred (AccessDenied) when calling the PutObject operation: User: arn:aws:sts::47
4668386301:assumed-role/exp6/i-00be47dcddaf3f9c4 is not authorized to perform: s3:PutObject on resource: "arn:aws:s3:::caexp6/test.txt" becaus
e no identity-based policy allows the s3:PutObject action
ubuntu@ip-172-31-22-3:~$
```

This procedure ensures a secure and properly managed IAM role, contributing to overall AWS environment security and operational efficiency.

Conclusion:

This experiment highlighted how crucial Identity and Access Management (IAM) is for securing AWS resources. By setting up IAM roles with specific managed and custom policies, we were able to ensure that access was limited to only what was absolutely necessary for each task. Assigning these roles to services like EC2 and Lambda allowed for secure and controlled interactions with AWS resources while sticking to the principle of least privilege. This approach not only made the infrastructure more secure but also gave me hands-on experience in managing permissions effectively in a cloud environment, which is essential for building a secure and compliant AWS setup.