Lab 03

CLOUD SECURITY

CS-4105

Course Instructor:

Dr Abid Rauf

Name: Zeeshan Ali

Roll No: 20i-2465

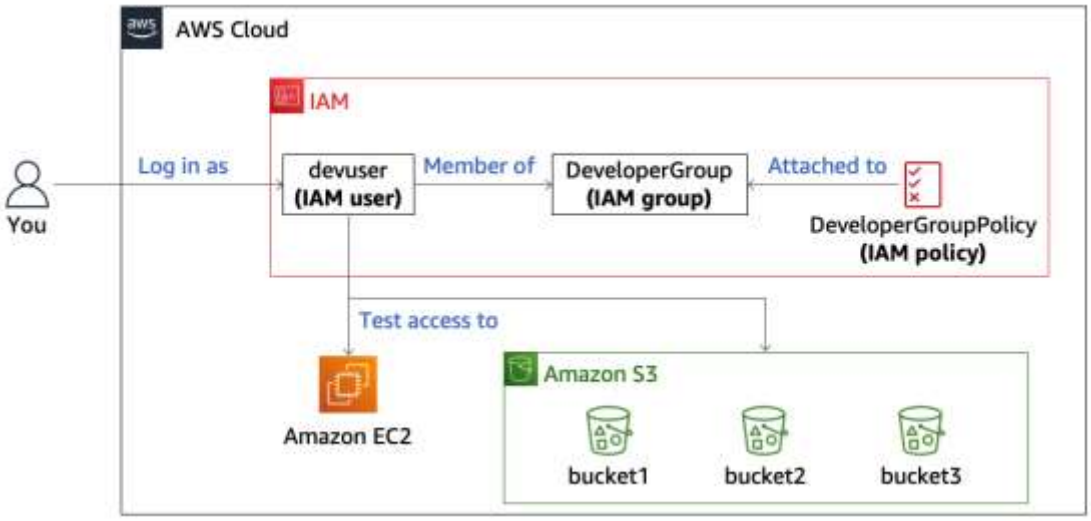
Section: CS - B

Due Date: April 21, 2024

Lab 3.1: Using Resource-Based Policies to Secure an S3 Bucket

Scenario

The following diagram shows the architecture that was created for you in AWS at the *beginning* of the lab.



The lab environment has three preconfigured Amazon S3 buckets: *bucket1*, *bucket2*, and *bucket3*. The environment also has a preconfigured IAM role, which allows access to certain buckets and their objects when the role is assumed. You will analyze different policies to better understand how they control your access level.

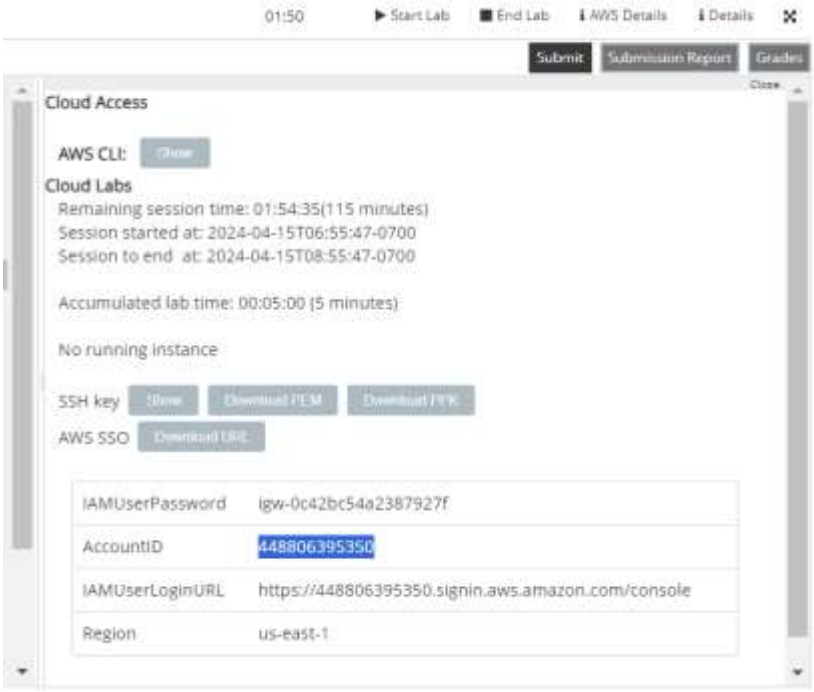
Task 1: Accessing the console as an IAM user

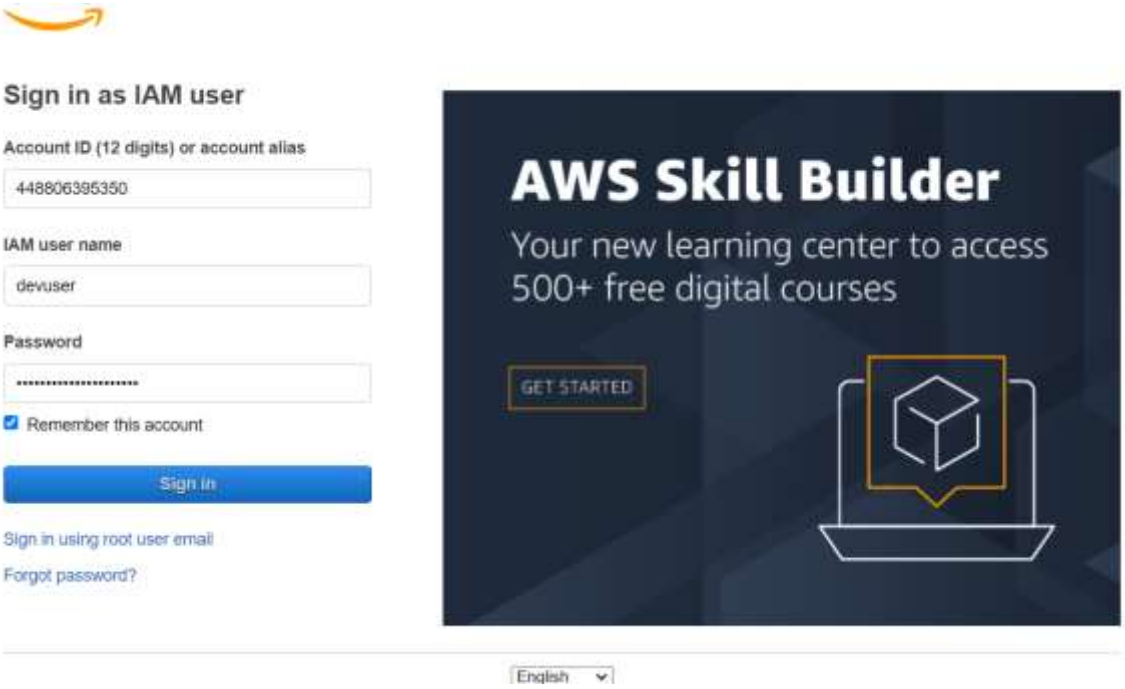
1. At the top of these instructions, choose **Start Lab**. **Tip:** To refresh the session length at any time, choose **Start Lab** again before the timer reaches 00:00.

2. Before you continue, wait until the circle icon to the right of the AWS link in the upper-left corner turns green. When the lab environment is ready, the AWS Details panel will also display.

3. Log in as the IAM user named *devuser*:
    - Choose the **AWS Details** link at the top of the page.
    - Copy the **IAMUserLoginURL** value, and load it in a new browser tab.

- For **IAM user name**, enter devuser
- For **Password**, enter the **IAMUserPassword** value from the AWS Details panel on the lab instructions page.
- Choose **Sign in**. The AWS Management Console displays.

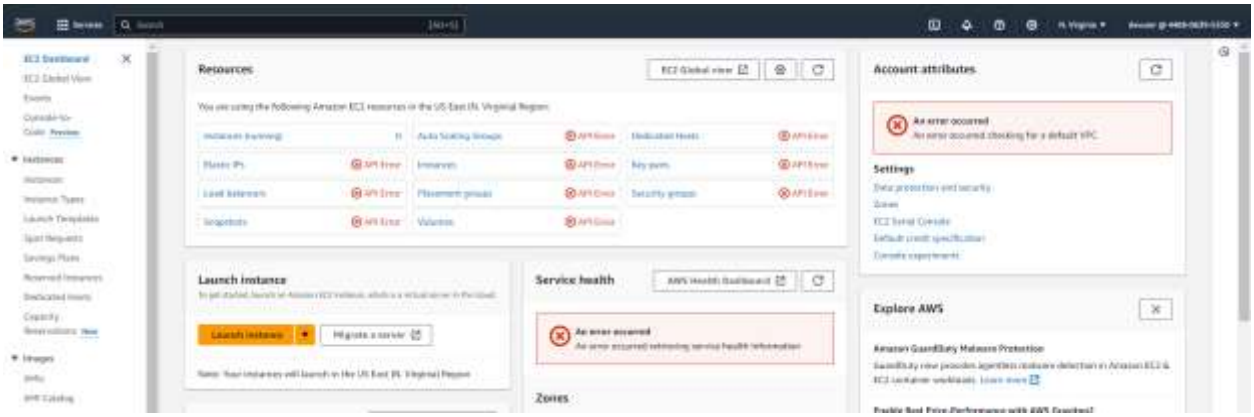




Task 2: Attempting read-level access to AWS services

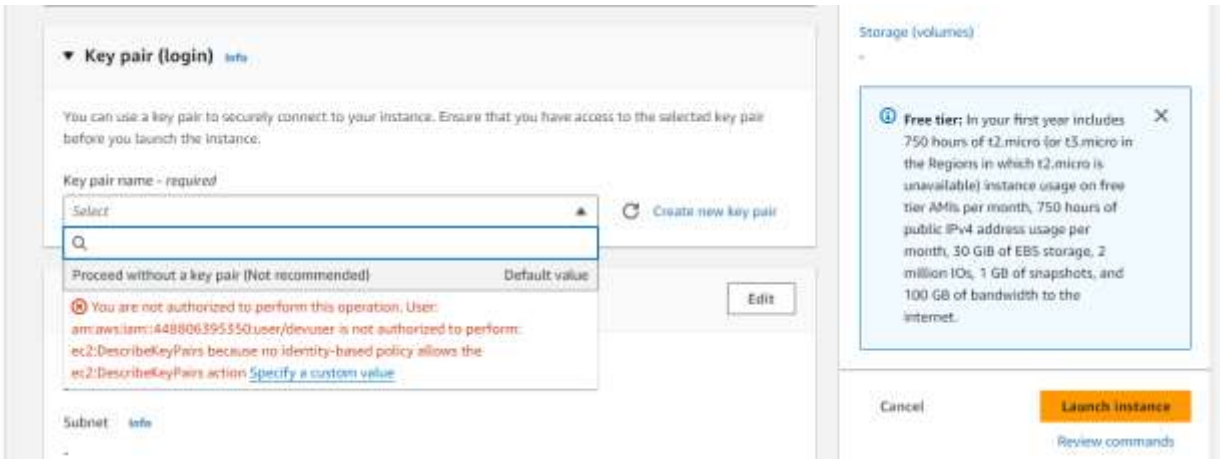
Now that you are logged in to the console as the IAM user named *devuser*, you will explore the level of access that you have to a few AWS services, including Amazon Elastic Compute Cloud (Amazon EC2), Amazon S3, and IAM.

5. Open the Amazon EC2 console:
    - From the **Services** menu, choose **Compute** > **EC2**.
    - In the left navigation pane, choose **EC2 Dashboard**. Many API Error messages display. This is expected.

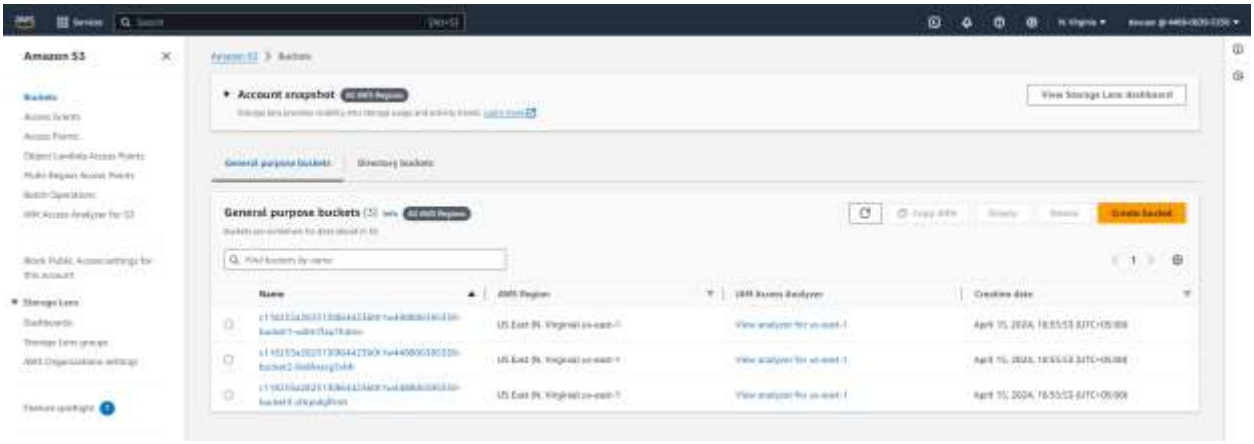


6. Attempt some actions in the Amazon EC2 console:
    - In the left navigation pane, choose **Instances**. In the Instances list, a message displays *You are not authorized to perform this operation*.

    - Choose **Launch instances**
    - Scroll down and choose the Key pair name from the drop down list. A message displays *You are not authorized to perform this operation*. Notice that Key pair name is a *required* setting that must be configured if you want to launch an instance. This is just one of many indications that you will not be able to launch an EC2 instance with the permissions that have been granted to you as the devuser.



7. To explore what you can access in the Amazon S3 console, from the **Services** menu, choose **Storage** > **S3**. Three buckets are listed. The bucket names are unique, but one bucket name contains *bucket1*, another contains *bucket2*, and the third contains *bucket3*. In the list of buckets, notice that the **Access** column displays the message Insufficient permissions for all three buckets. This is expected.

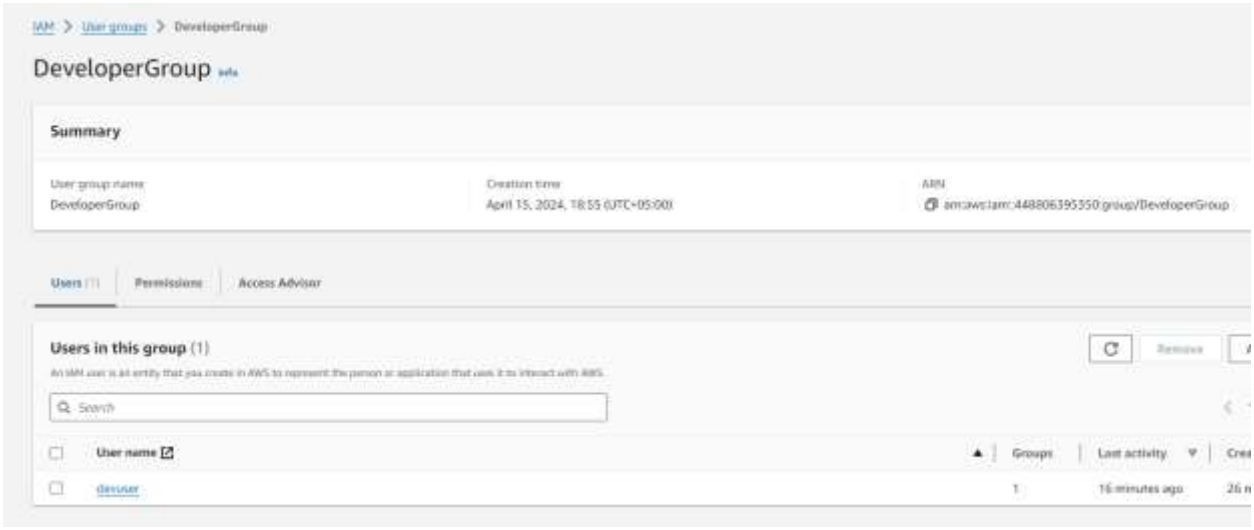


Task 3: Analyzing the identity-based policy applied to the IAM user

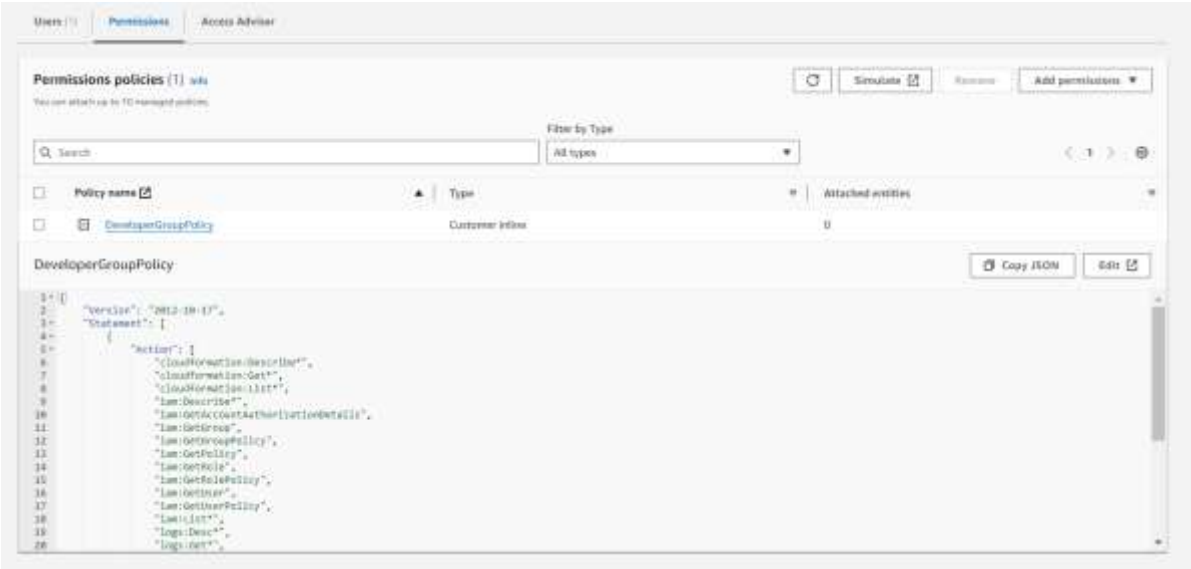
You have observed how the *devuser* IAM user is unable to access certain information and actions in both the Amazon S3 console and Amazon EC2 console. In this task, you will look at the IAM policy details that apply to *devuser* to understand why you can't perform these actions.

8. Access the IAM console, and observe user and group membership settings:
    - From the **Services** menu, choose **Security, Identity, & Compliance** > **IAM**. On the IAM dashboard page, notice that you do not have permissions to view certain parts of the page. Both messages state *User: arn:aws:iam:::user/devuser is not authorized to perform: iam:GetAccountSummary on resource: **. This is expected.
    - In the left navigation pane, choose **User groups**.

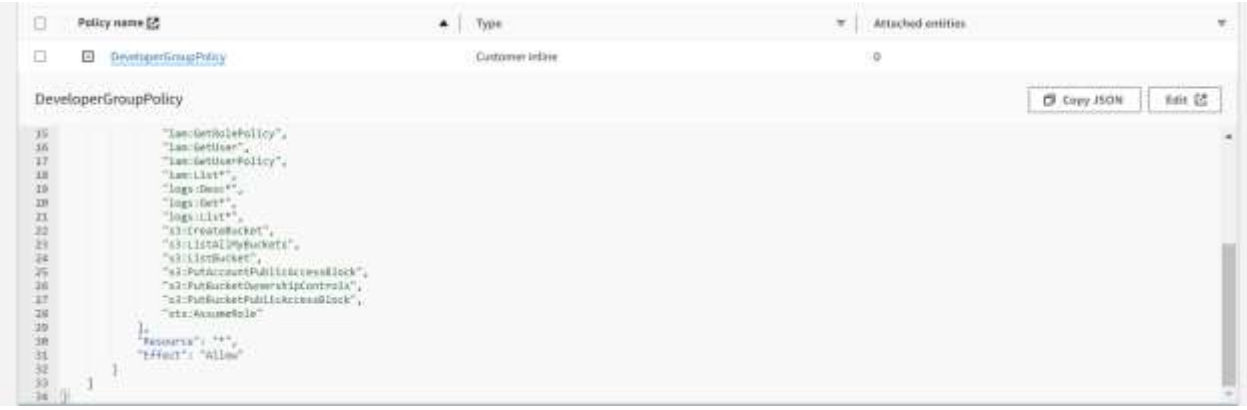
Choose the **DeveloperGroup** group name. On the **Users** tab, notice that *devuser* is a member of this IAM group.



Choose the **Permissions** tab. Notice that an IAM policy named DeveloperGroupPolicy is attached to this IAM group. **Note:** When a policy is attached to a group, the policy applies to any IAM users who are members of the group. Therefore, this policy currently governs your access to the console, because you are logged in as *devuser*, who is a member of this IAM group.



9. Review the IAM policy details:
    - On the lower portion of the page, choose the plus icon to the left of **DeveloperGroupPolicy** to display the policy details.
    - Review the JSON policy details, and recall the level of access that you had for Amazon EC2 and Amazon S3 in the previous task.
        - Notice that the policy does not allow any Amazon EC2 actions.
        - Notice the IAM actions that the policy allows. When you accessed the IAM dashboard, you saw a message that stated that you did not have *iam:GetAccountSummary* authorization. That action is not permitted in this policy document. However, many read-level IAM permissions are granted. For example, you are able to review the details for this policy.
        - Notice the Amazon S3 actions that the policy allows. No object-related actions are granted, but some actions related to buckets are allowed.



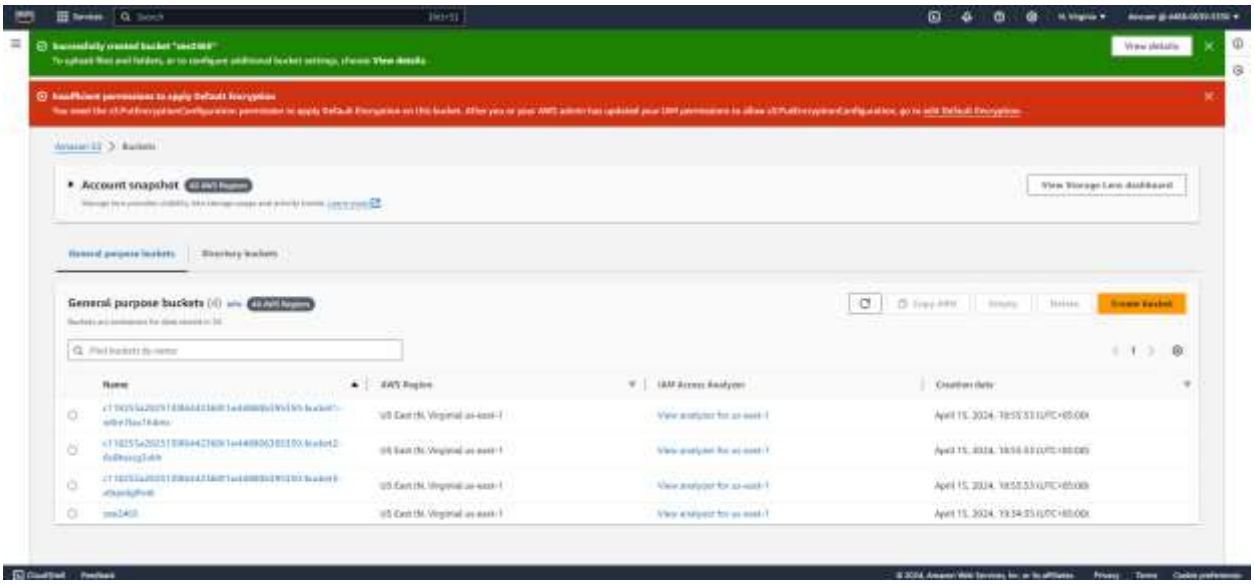
10. Save the policy to a file on your computer:
    - To copy the JSON-formatted policy to your clipboard, choose **Copy**.
    - Open a text editor on your local computer, and paste the policy that you just copied.
    - Save the policy document as `DeveloperGroupPolicy.json` to a location on your computer that you will remember.

Task 4: Attempting write-level access to AWS services

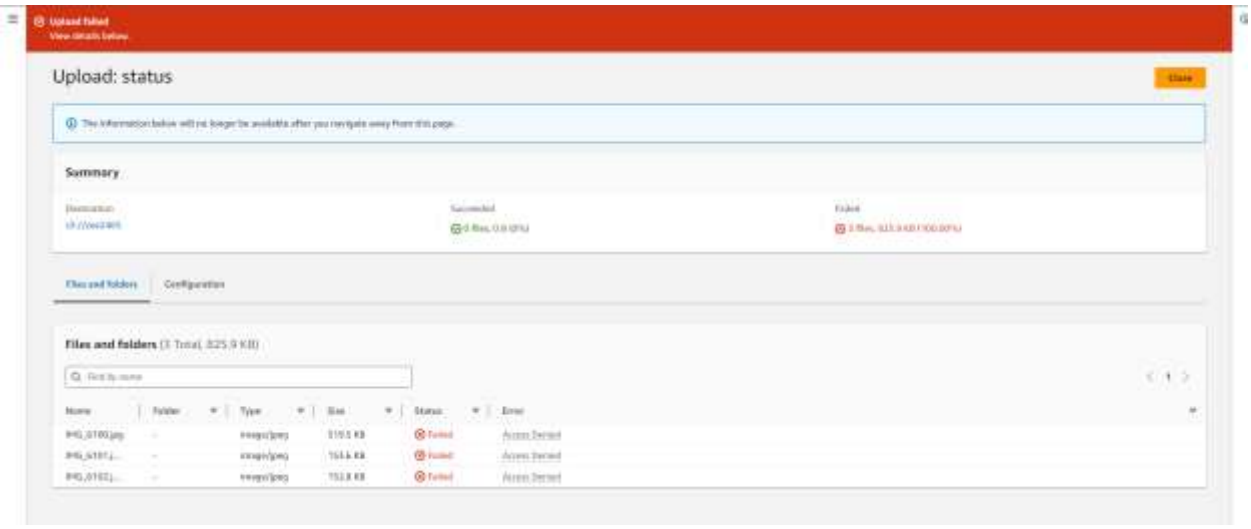
Any action that you attempt when you interact with an AWS service is an API call, whether you are using the console, AWS Command Line Interface (AWS CLI), or AWS software development kits (SDKs). All attempted API calls are recorded in the AWS CloudTrail event logs.

In this task, you will attempt to make two API calls that require *write-level* access within Amazon S3. The first action is to create an S3 bucket, and the second action is to upload an object to that bucket. After you attempt the two tasks, you will again analyze the policy attached to the IAM group to analyze why you could or could not perform the specific API calls.

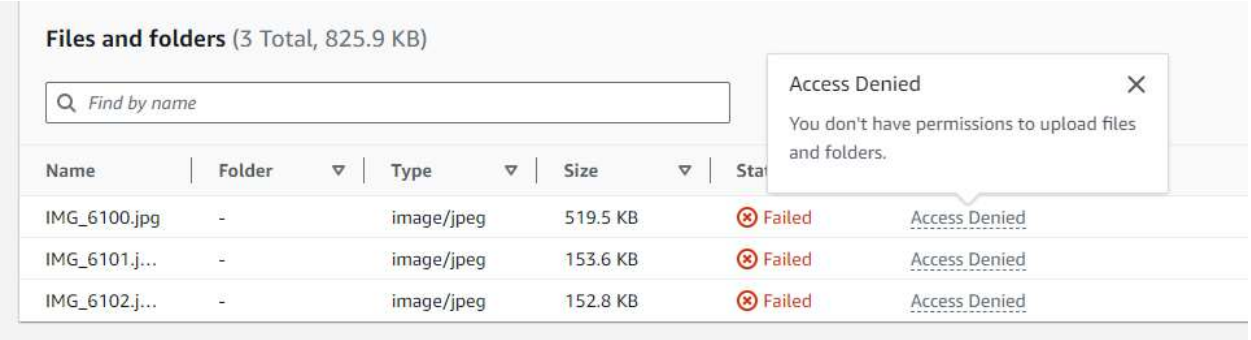
11. Attempt to create an S3 bucket:
    - Navigate to the Amazon S3 console. **Tip:** Use the **Services** menu, or search for `S3` in the search box to the right of the menu.
    - Choose **Create bucket**
    - For **Bucket name**, enter your initials followed by a random four-digit number; for example, *zba1234*. **Note:** By default, new buckets, access points, and objects don't allow public access. Diving deeper into this goes beyond the scope of this lab, but it's important to note.
    - For **AWS Region**, choose **US East (N. Virginia) us-east-1**.
    - Review the settings, and then choose **Create bucket** at the bottom of the page. You successfully created an S3 bucket.



12. Access the bucket, and attempt to upload an object:
    - Choose the name of the bucket that you just created.
    - Choose **Upload**, and then choose **Add files**.
    - Browse to and choose the **DeveloperGroupPolicy.json** file that you saved earlier.
    - Choose **Upload**. A message displays *Upload failed*.



- On the **Files and folders** tab on the lower part of the page, in the **Error** column, choose the **Access Denied** link. The message states *You don't have permission to upload files and folders*.
- Choose **Close**.
- From the breadcrumbs in the upper-left corner of the page, choose **Amazon S3**.

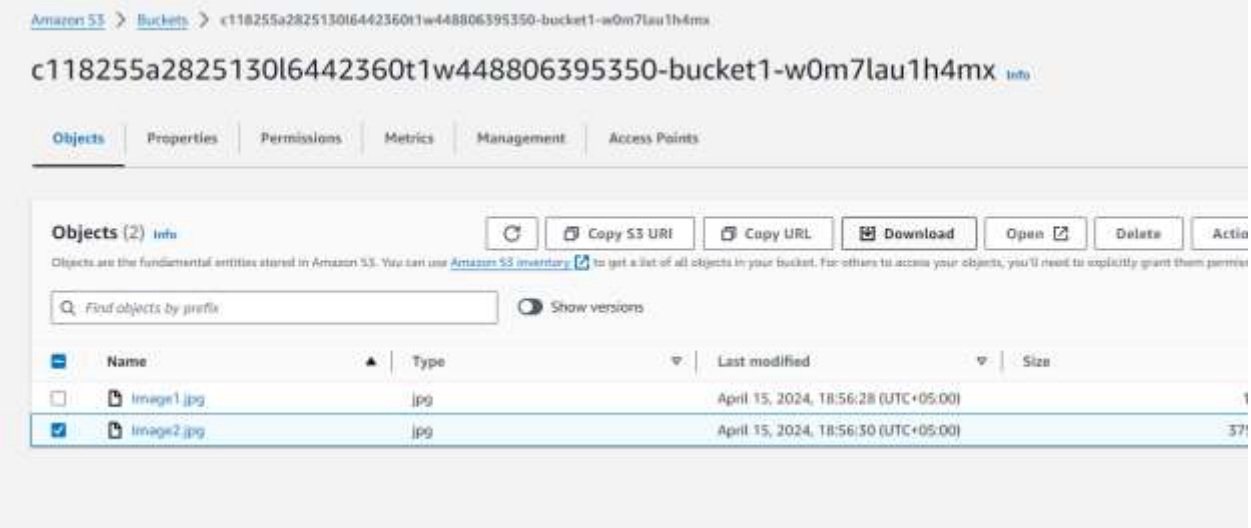


13. Review the policy details for Amazon S3 access:
    - Return to the text editor where you copied the DeveloperGroupPolicy.json document.
    - Review the policy details to understand why you were able to create an S3 bucket but couldn't upload objects to it.

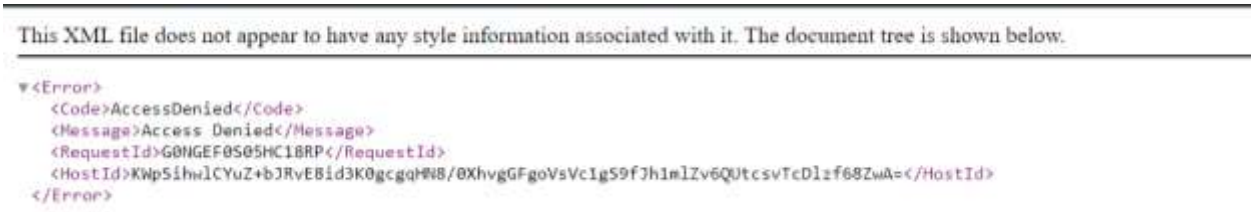
Task 5: Assuming an IAM role and reviewing a resource-based policy

In this task, you will try to access *bucket1* and *bucket2* while logged in as the *devuser* IAM user. You will also try to access the buckets by using a role that was preconfigured as part of the lab setup.

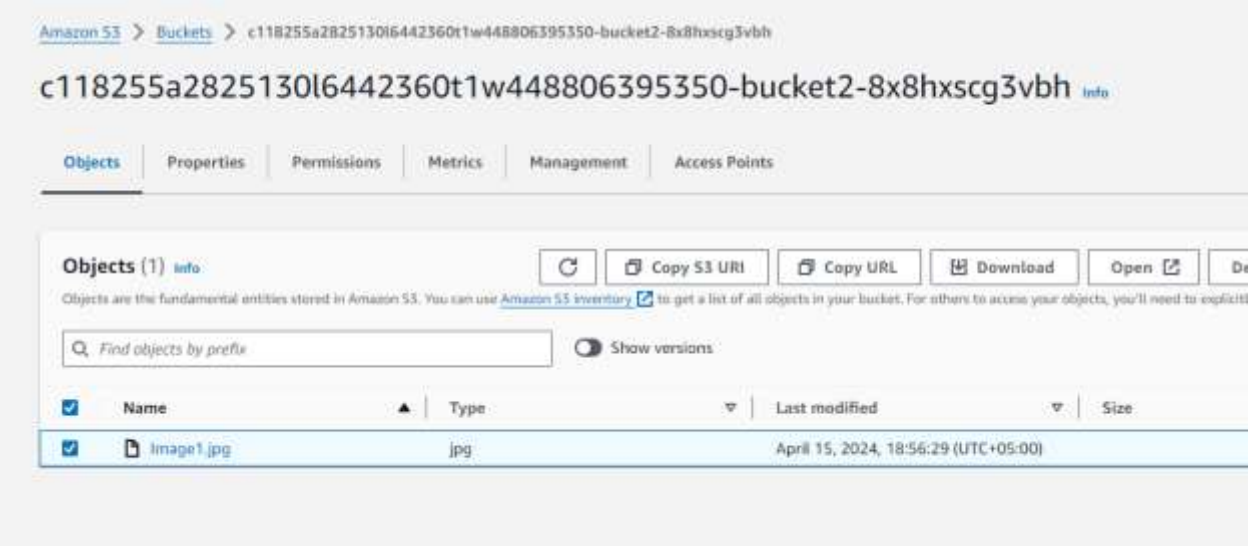
14. Try to download an object from the buckets that were created during lab setup:
    - In the Amazon S3 console, choose the bucket name that contains **bucket1**.
    - Select **Image2.jpg**, and then choose **Download**.



An AccessDenied error page appears.



- To return to the Amazon S3 console, choose your browser's back button.
- From the breadcrumbs in the upper-left corner of the page, choose **Amazon S3**.
- Try to download the **Image1.jpg** file from *bucket2*.



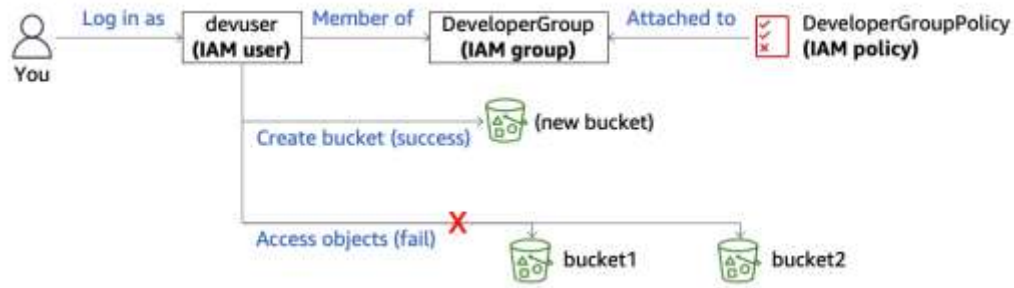
You receive the same error.

This XML file does not appear to have any style information associated with it. The document tree is shown below.


```
▼<Error>
   <Code>AccessDenied</Code>
   <Message>Access Denied</Message>
   <RequestId>17H074ZS49TQVK9T</RequestId>
   <HostId>mimGtjkNbIiqy3SowiM9jdR0Icq2F96UtEF8sPxYpC/sNKONUONU4vGIaEgEK0pPHMJ2+n4EWpQ=</HostId>
 </Error>
```

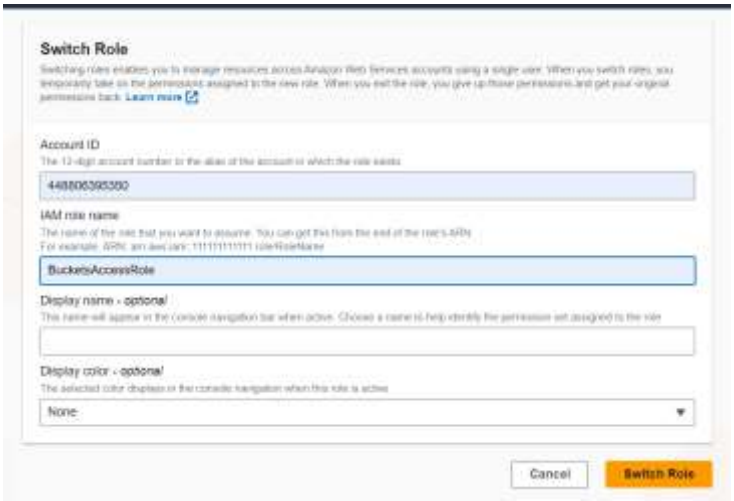

- To return to the Amazon S3 console, choose your browser's back button.

Analysis: As shown in the following diagram, with the permissions that are granted through membership in the *DeveloperGroup*, you were able to create a new bucket. However, you cannot access objects in *bucket1* or *bucket2*.

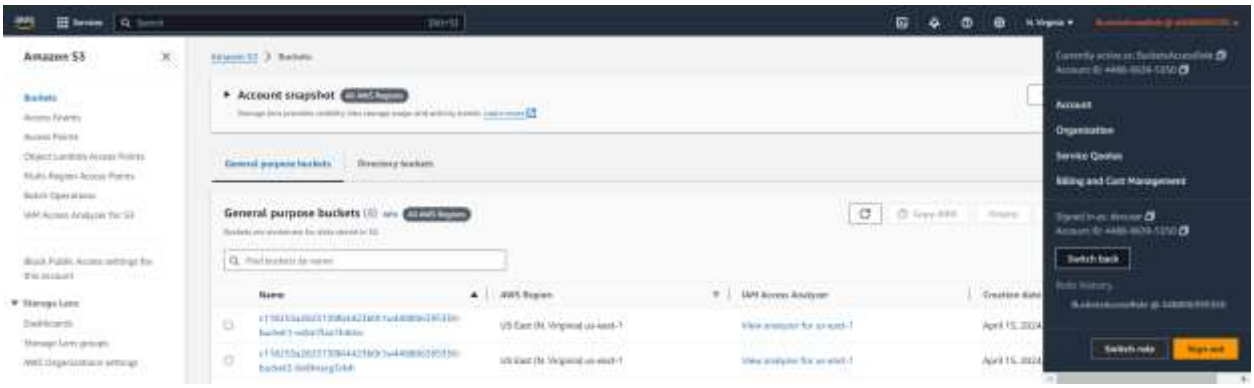


- From the breadcrumbs in the upper-left corner of the page, choose **Amazon S3**.

15. Assume the *BucketsAccessRole* IAM role in the console:
    - In the upper-right corner of the page, choose **devuser**, and then choose **Switch role**.
    - If the Switch role page appears, choose **Switch Role.**
    - Configure the following:
        - **Account:** Enter the **AccountID** value from the AWS Details panel on the lab instructions page.
        - **Role:** Enter `BucketsAccessRole`
        - **Display Name:** Leave this field blank.
        - Choose **Switch Role**

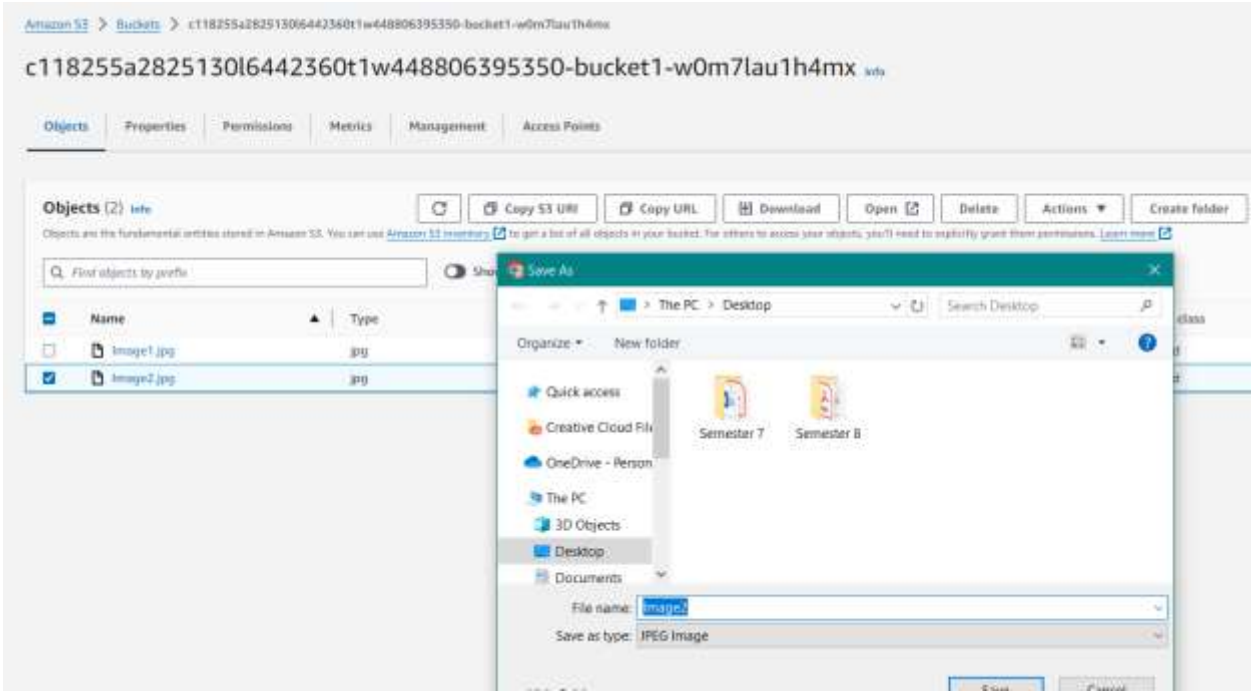


You successfully assumed the IAM role named *BucketsAccessRole*, which was preconfigured for this lab. **Tip:** You can tell that you switched into the role by looking at the upper-right corner of the console. Notice that **BucketsAccessRole** is displayed where **devuser** was previously displayed.

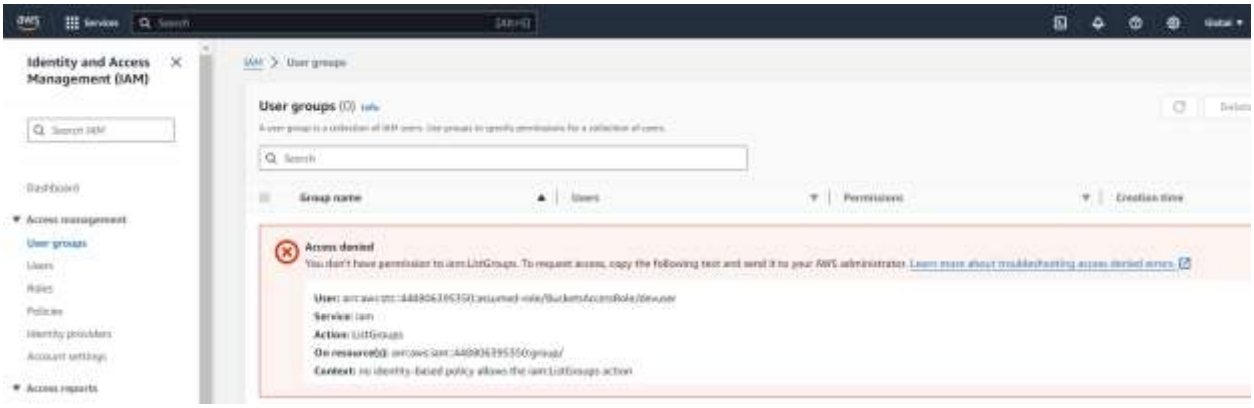


16. Try to download an object from Amazon S3 again:
    - In the Amazon S3 console, choose the bucket name that contains **bucket1**.
    - Select **Image2.jpg**, and then choose **Download**.
    - Open the file to verify that the file downloaded.

Analysis: The download was successful, which means that the policy or policies applied to the *BucketsAccessRole* allow the *s3:GetObject* action on *bucket1*.

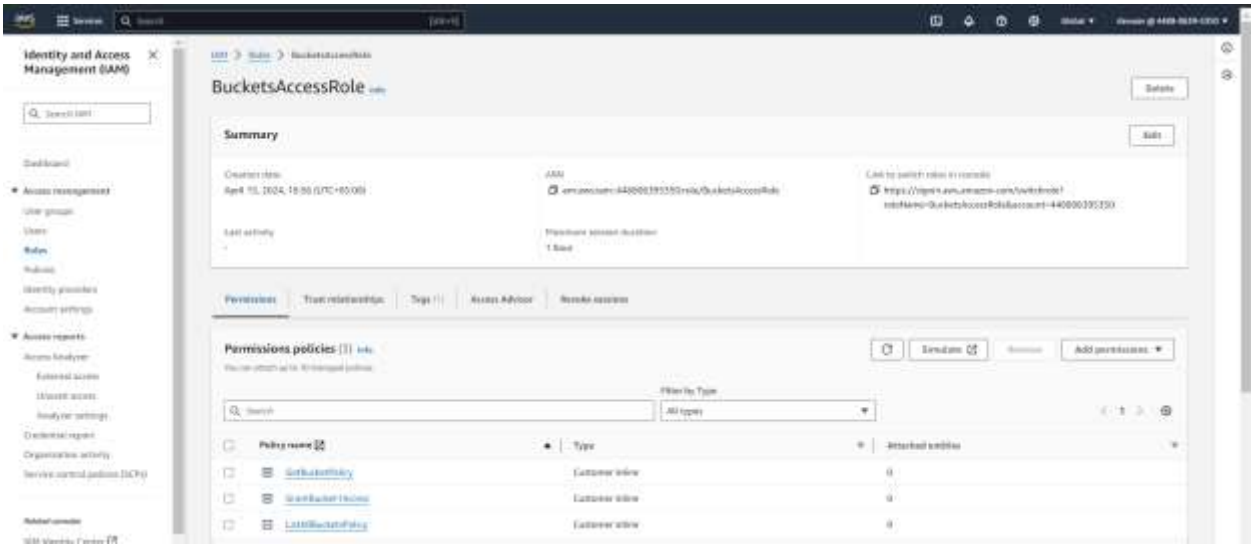


17. Test IAM access with the *BucketsAccessRole*:
    - Navigate to the IAM console. **Note:** By changing roles, the permissions that you have to interact with different AWS services have changed. As you navigate the IAM console, you will see new error messages that state that you are not authorized.
    - In the left navigation pane, choose **User groups**. **Analysis:** An error message displays. You no longer have permissions to view the IAM user groups page because *BucketsAccessRole* does not have the *iam:ListGroups* action applied to it.

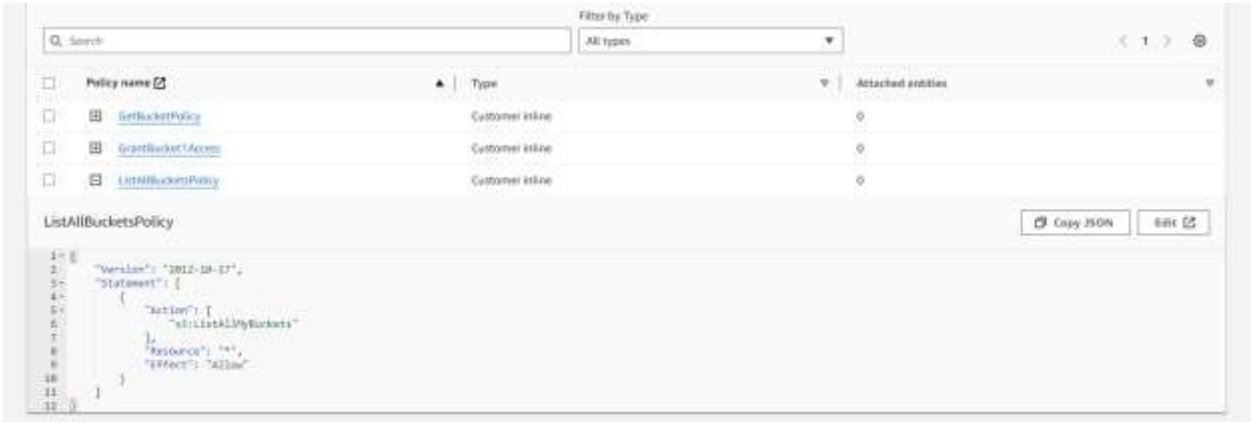


18. Assume the *devuser* role again, and test access to the user groups page:
    - In the upper-right corner of the page, choose **BucketsAccessRole**, and then choose **Switch back**.
    - In the left navigation pane, choose **User groups** again. **Analysis:** Now that you unassumed the *BucketsAccessRole*, you have the permissions that are assigned to the *devuser* IAM user (through this user's membership in the *DeveloperGroup*). You are able to view the user groups page again.

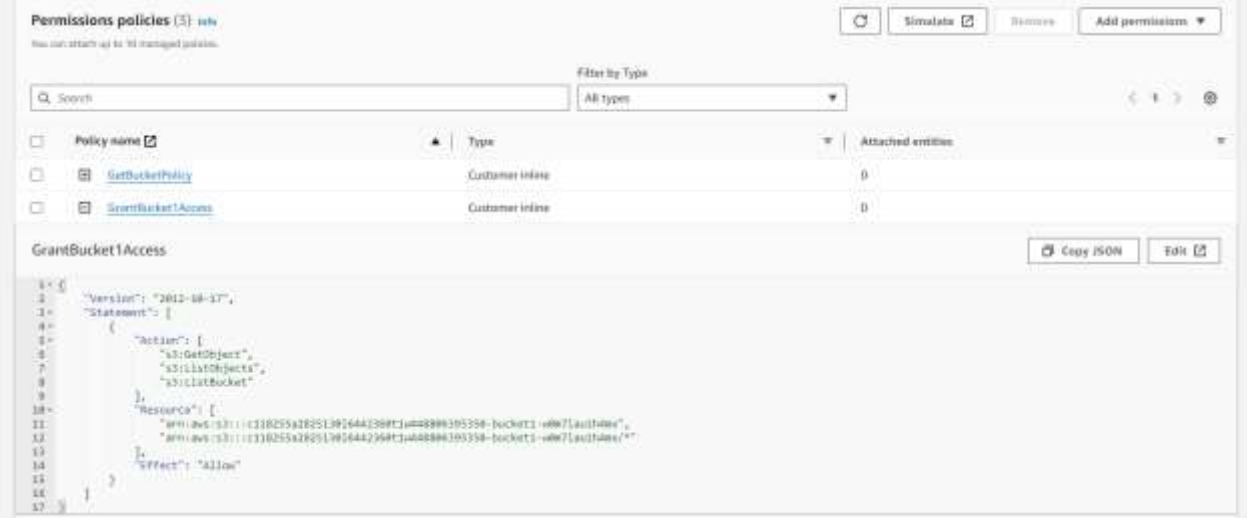
19. Analyze the IAM policy that is associated with the *BucketsAccessRole*:
    - In the left navigation pane, choose **Roles**.
    - Search for `BucketsAccessRole` and choose the role name when it appears.



Choose the arrow to the left of **ListAllBucketsPolicy**. This policy grants the same *s3:ListAllMyBuckets* action to every resource. This permission allows you to see all S3 buckets when you assume *BucketsAccessRole*.

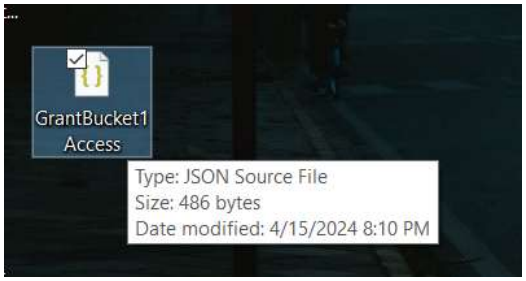


- Choose the arrow to the left of **GrantBucket1Access**.



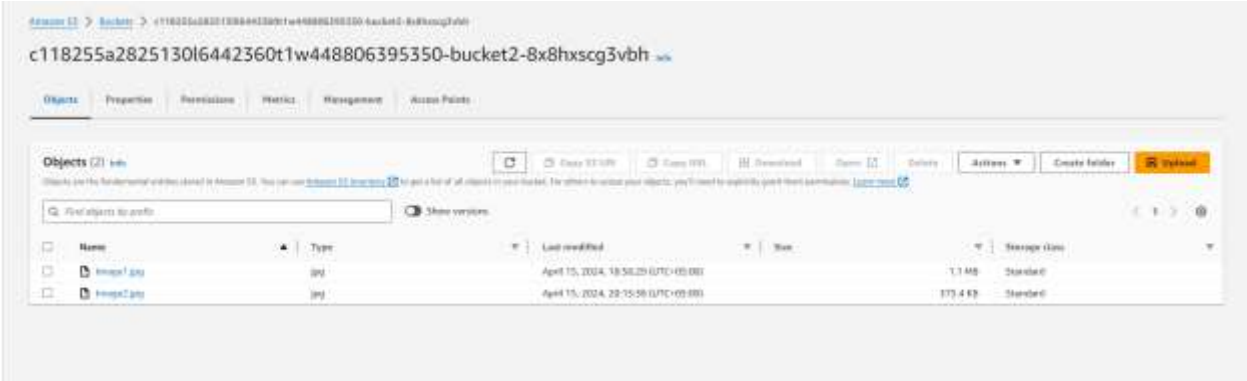
Analysis: This policy allows the *s3:GetObject*, *s3:ListObjects*, and *s3:ListBucket* actions. Notice that this policy does *not* grant *s3:PutObject* access. The allowed actions are only granted for specific resources, *bucket1* and all objects within *bucket1* (as indicated by **/***). The asterisk (*) is a wildcard character, which indicates that this would match any value. Because of this policy, when you assumed the *BucketsAccessRole*, you could see and download objects from *bucket1*.

20. Save a copy of the *GrantBucket1Access* policy to your computer:
    - Place your cursor at the start of line 1 in the policy details, and select all the lines of code (down to line 17).
    - Copy the JSON-formatted policy to your clipboard.
    - Open a new text file on your computer, and paste the policy that you just copied.
    - Save the policy document as `GrantBucket1Access.json` to a location on your computer that you will remember.



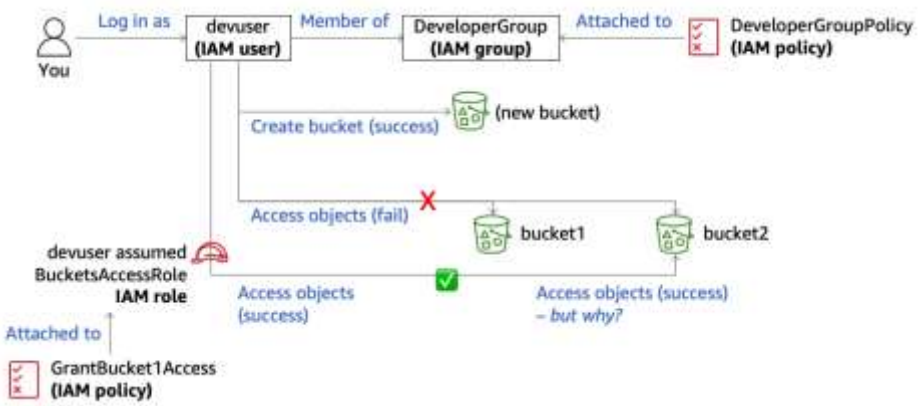
21. Complete your analysis of the *BucketsAccessRole* details:
    - Scroll back up the page, and choose the **Trust relationships** tab. Notice that the *devuser* IAM user in this AWS account is listed as a trusted entity that can assume this role. Notice that the account number that appears in the upper-right corner of the console (after **devuser**) matches the account number in the **Trusted entities** list (without the dashes).

22. Assume the *BucketsAccessRole*, and try to upload an image to *bucket2*:
    - To assume the *BucketsAccessRole* again, in the upper-right corner of the page, choose **devuser**.
    - Under **Role history**, choose **BucketsAccessRole**.
    - Navigate to the Amazon S3 console.
    - Choose the bucket name that contains **bucket2**. Notice that this bucket does not yet have an Image2.jpg file.
    - Choose **Upload**, and then choose **Add files**.
    - Browse to and choose the **Image2.jpg** file that you downloaded earlier from *bucket1*.
    - Choose **Upload**. The file uploads successfully.



    - Choose **Close**.

Analysis: After assuming the *BucketsAccessRole*, you successfully accessed *bucket1* to download an object. You then uploaded the same object to *bucket2*. After inspecting the policies attached to the *BucketsAccessRole*, you know that the Amazon S3 permissions that were granted to that role were limited to *bucket1*, as shown in the following diagram. So, how were you just now able to upload an object to *bucket2*? The reason will become clear in the next task.



Task 6: Understanding resource-based policies

In this task, you will inspect the bucket policy that is associated with *bucket2*.

23. Observe the details of the bucket policy that is applied to *bucket2*:
    - On the details page for *bucket2*, choose the **Permissions** tab.
    - In the **Bucket policy** section, review the policy that is applied to *bucket2*.

The policy has two statements.

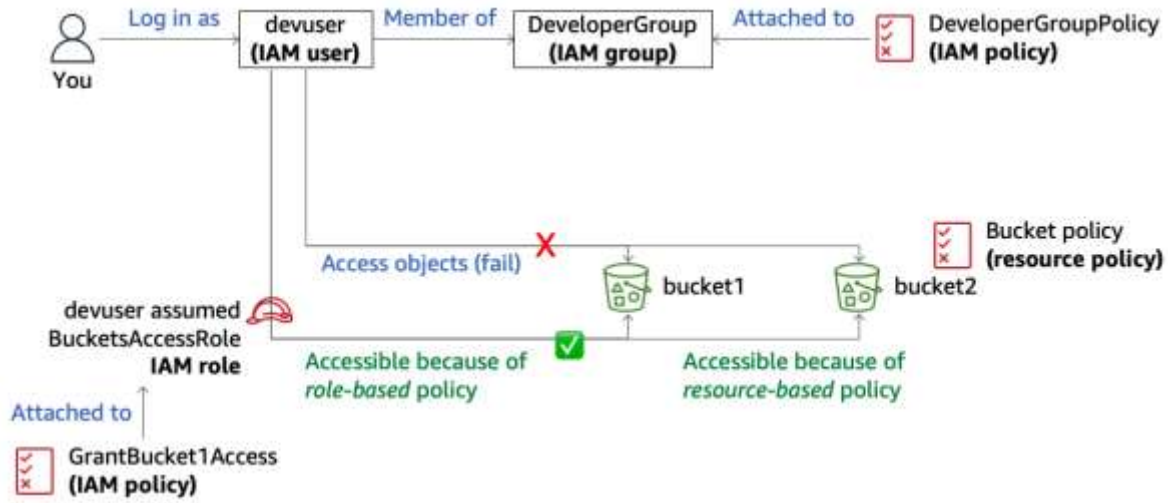
- The first statement ID (SID) is *S3Write*. The principal is the *BucketsAccessRole* IAM role that you assumed. This role is allowed to call the actions *s3:GetObject* and *s3:PutObject* on the resource, which is *bucket2*.
- The second SID is *ListBucket*. The principal is *BucketsAccessRole*. This role is allowed to call the action *s3:ListBucket* on the resource, which is *bucket2*.

Policy:

Analysis: You should now have a better understanding of how resource-based policies (such as S3 bucket policies) and role-based policies (policies associated with IAM roles) can interact and be used together.

In this lab, the *role-based policies* attached to the *BucketsAccessRole* IAM role granted *s3:GetObject* and *s3:ListBucket* access to *bucket1* and the objects in it. These role-based policies did not explicitly allow access to *bucket2*; however, they also did not explicitly deny access.

The following diagram shows how the policies that were applied to the IAM user, IAM role, and bucket determined what actions you were able to perform.

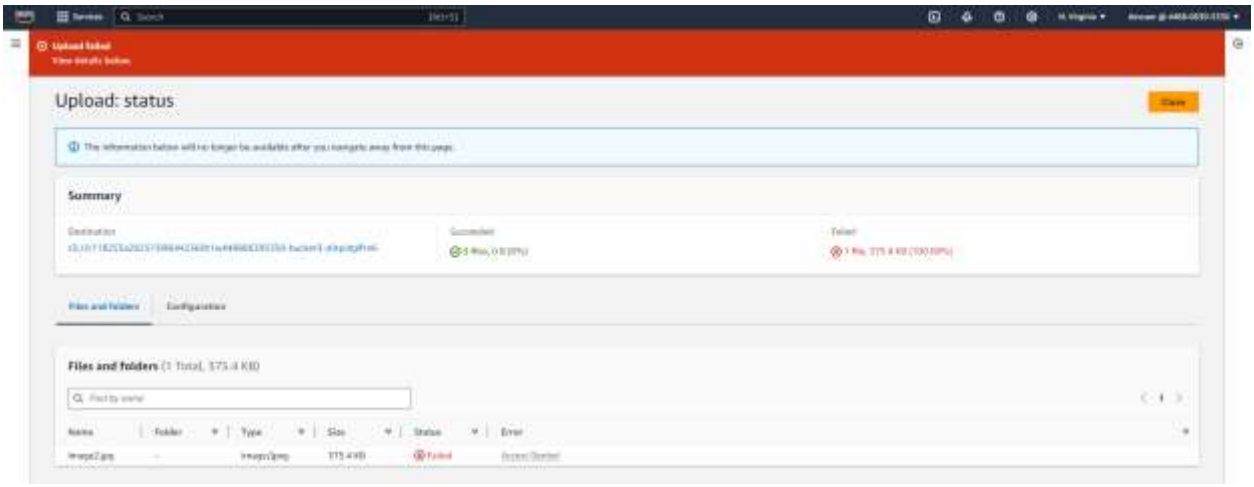


Then, while still assuming the *BucketsAccessRole*, you tried to upload an object to *bucket2*, and you were able to do it. That seemed strange based on the IAM policies that you reviewed. However, after you reviewed the *resource-based policy* (in this case, a bucket policy) that was attached to the bucket, your access made sense. That bucket policy grants access, including the *s3:PutObject* action, to *bucket2* to the *BucketsAccessRole* principal.

Challenge Task:

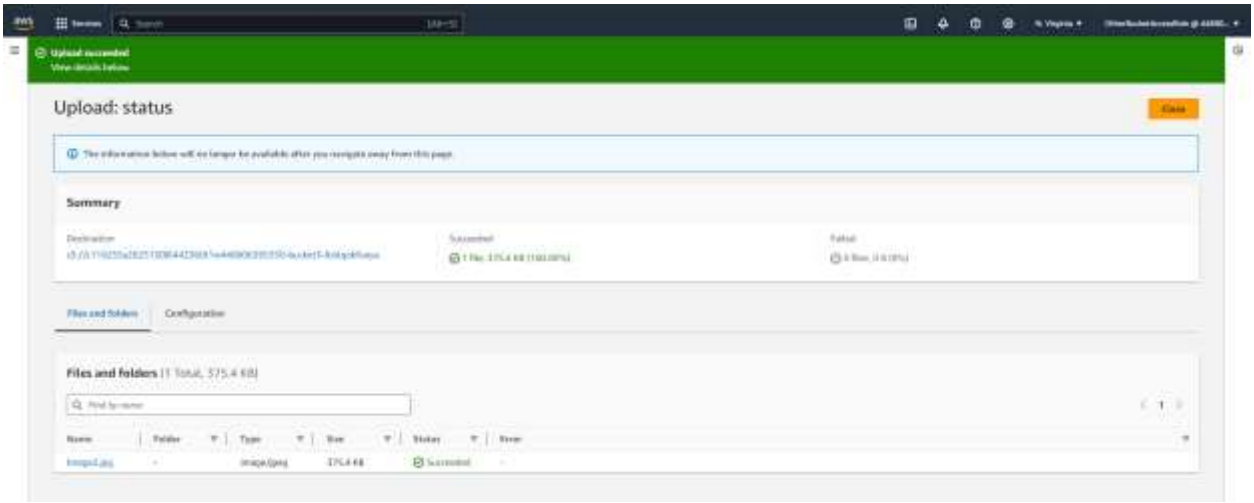
Your objective for this challenge task is to figure out a way to upload the Image2.jpg file to *bucket3*.

24. Try to upload the file as *devuser* with no role assumed:
    - Unassume the *BucketsAccessRole*.
    - Attempt to upload Image2.jpg, which you downloaded from *bucket1* earlier in this lab, to *bucket3*. The upload fails.

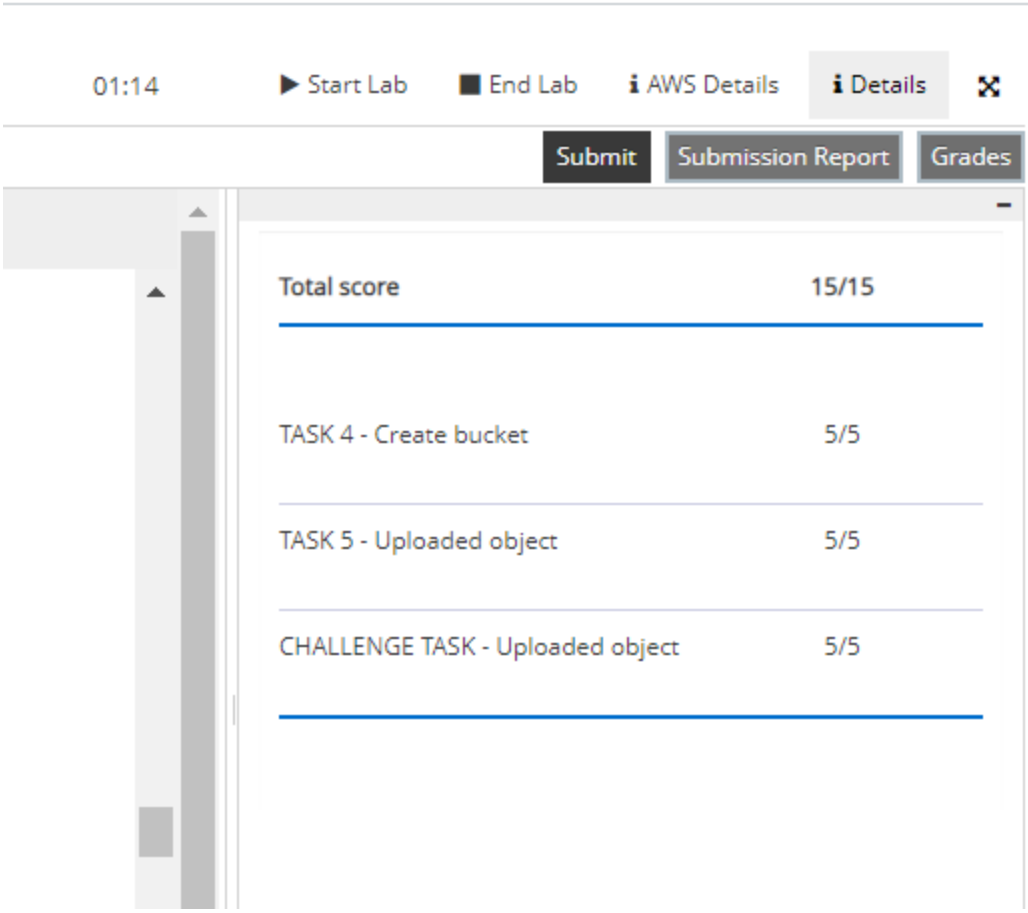


- Check whether a bucket policy is associated with *bucket3*. Maybe that will give you some indication about how to accomplish this task.You can't view the bucket policy.

25. Assume the *BucketsAccessRole*, and try the actions from the previous step:
    - Can you upload a file to *bucket3*?
    - Can you view the bucket policy now? Review the bucket policy details. Do you have an idea for how you can upload Image2.jpg to *bucket3*?
    - Did you figure out how to upload the file? If so, congratulations!



```
{
  "Version": "2008-10-17",
  "Statement": [
    {
      "Sid": "S3Write",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::448806395350:role/OtherBucketAccessRole"
      },
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": "arn:aws:s3:::c118255a2825130l6442360t1w448806395350-bucket3-8xltqokfueya/*"
    },
    {
      "Sid": "ListBucket",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::448806395350:role/OtherBucketAccessRole"
      },
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::c118255a2825130l6442360t1w448806395350-bucket3-8xltqokfueya"
    }
  ]
}
```


01:14
Start Lab
End Lab
AWS Details
Details
Submit
Submission Report
Grades
Total score
15/15
TASK 4 - Create bucket
5/5
TASK 5 - Uploaded object
5/5
CHALLENGE TASK - Uploaded object
5/5