From IAM to Bucket Policies: A Comprehensive Guide to S3 Access Control with Console, CLI, and Terraform

19 min read · Mar 22, 2024

M Mohasina Clt Follow

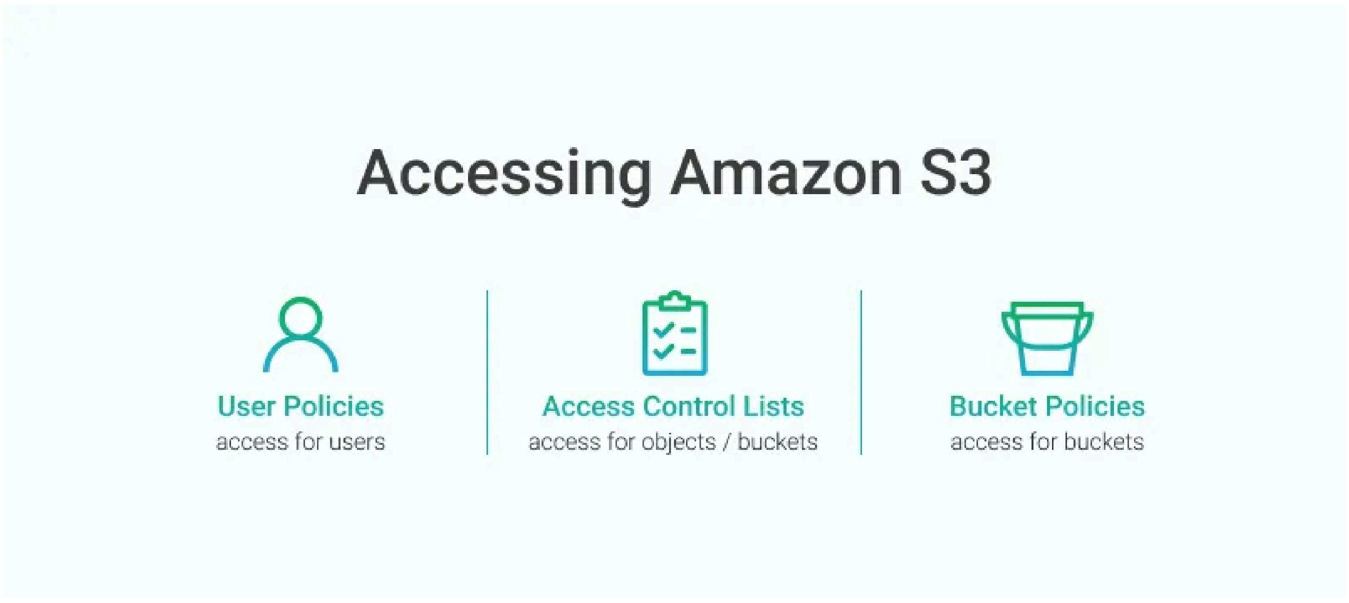
Listen Share More

Table of Contents:

Introduction:

Amazon S3 (Simple Storage Service) is a cornerstone of cloud storage, offering scalable, durable, and secure object storage. A key aspect of managing S3 buckets is controlling access to stored data. Access Control Policies play a pivotal role in defining who can access what in your S3 buckets. In this comprehensive guide, we'll dive into the fundamentals of Access Control Policies in Amazon S3, including IAM policies, bucket policies, configuring access control via the AWS Management Console, AWS CLI (Command Line Interface), and terraform. Additionally, we'll explore best practices for securing S3 buckets effectively.

Understanding IAM Policies and Bucket Policies:



💎 **IAM Policies:**

IAM (Identity and Access Management) policies are JSON documents that define permissions for IAM users, groups, and roles within your AWS account. These policies control access to various AWS services, including Amazon S3. IAM policies allow you to specify granular permissions, such as read, write, and delete actions, for specific resources or actions within those resources.

💎 **Bucket Policies:**

Bucket policies are JSON-based documents that are attached directly to S3 buckets. They define permissions for anyone accessing the bucket, including anonymous users. Bucket policies enable you to manage access controls at the bucket level, allowing you to specify who can perform actions such as reading, writing, or deleting objects within the bucket. These policies are powerful tools for managing access to S3 buckets and can be used in conjunction with IAM policies to enforce comprehensive access controls.

Configuring Access Control for S3 Buckets:

Configuring access control for S3 buckets involves defining permissions to regulate who can access, modify, or delete data stored within them. Through the AWS Management Console, users can set up IAM policies and bucket policies to specify access rights. The AWS CLI offers a command-line interface for scripting access control configurations, providing flexibility and automation. Additionally, Terraform allows for infrastructure as code, enabling users to define access control policies alongside other resources in a reproducible and scalable manner. By leveraging these methods, users can establish robust access control measures to protect their S3 bucket data from unauthorized access or modification.

In my blog, we'll explore configuring access control for S3 buckets using various methods, including the AWS Management Console, AWS CLI, and Terraform. Each approach will be detailed to ensure a comprehensive understanding of how to properly protect S3 buckets.

Getting Started with IAM and Bucket Policies Configuration in AWS Management Console:

Next, we'll guide you through setting up IAM and Bucket Policies in the AWS Management Console. Follow along as we demonstrate each step for configuring secure access control effortlessly.

1. IAM Policies Configuration:

IAM (Identity and Access Management) policies are essential for controlling access to AWS resources, including Amazon S3 buckets. They define the permissions granted to IAM users, groups, and roles within your AWS account. Let's explore how to configure IAM policies to manage access to S3 buckets effectively.

Steps for Configuring IAM Policies:

1. Navigate to the IAM Dashboard:

- Sign in to your AWS Management Console.
- In the top navigation bar, click on "Services" and select "IAM" under the "Security, Identity, & Compliance" section.

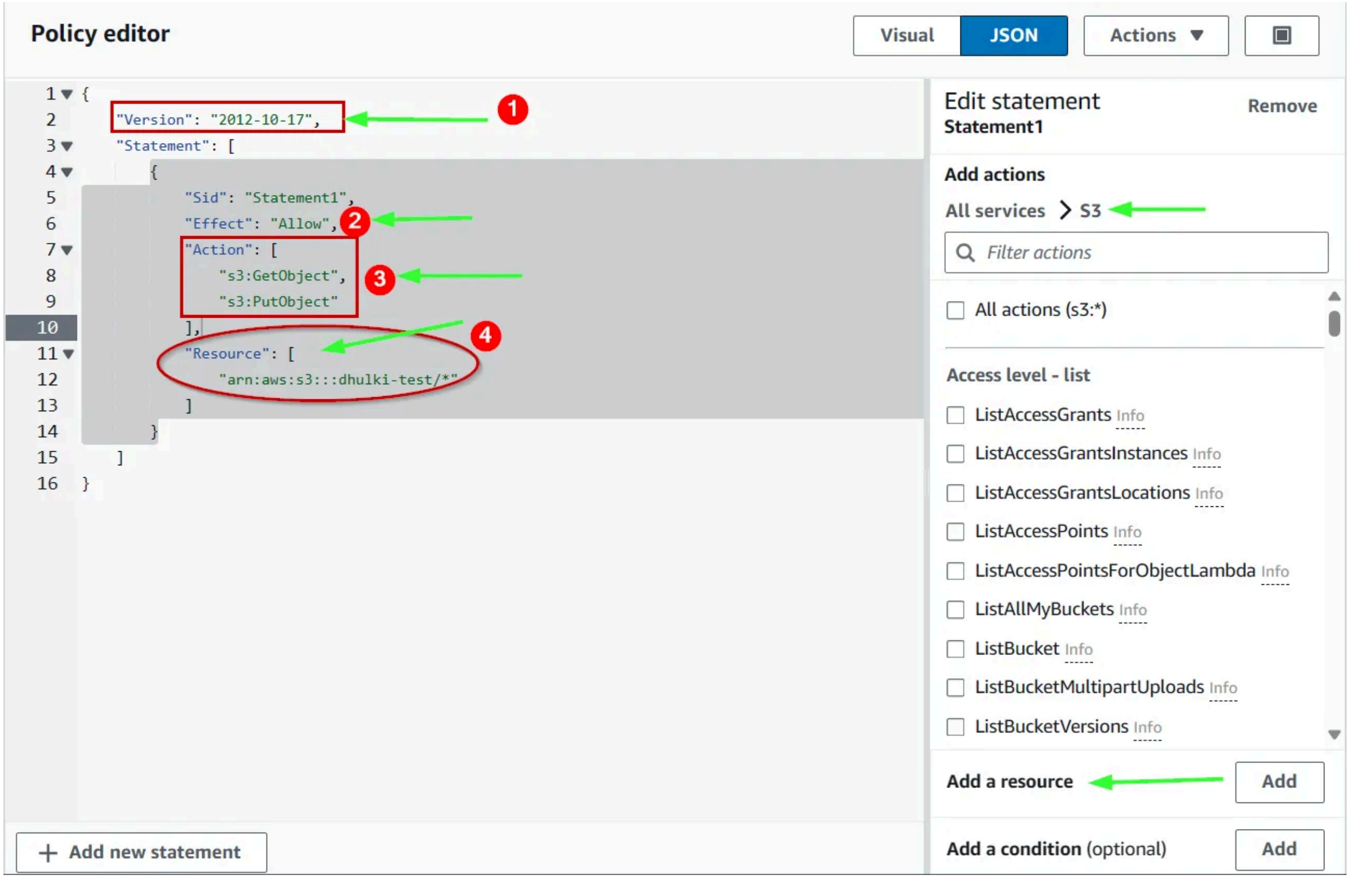
2. Create a New IAM Policy:

- In the IAM dashboard, select "Policies" from the left-hand side menu.
- Click on the "Create policy" button.

3. Define Policy Details:

- Choose the "JSON" tab to create a custom policy.
- Write or paste the JSON code defining the permissions for S3. For example:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::your-bucket-name/*"
        }
    ]
}
```



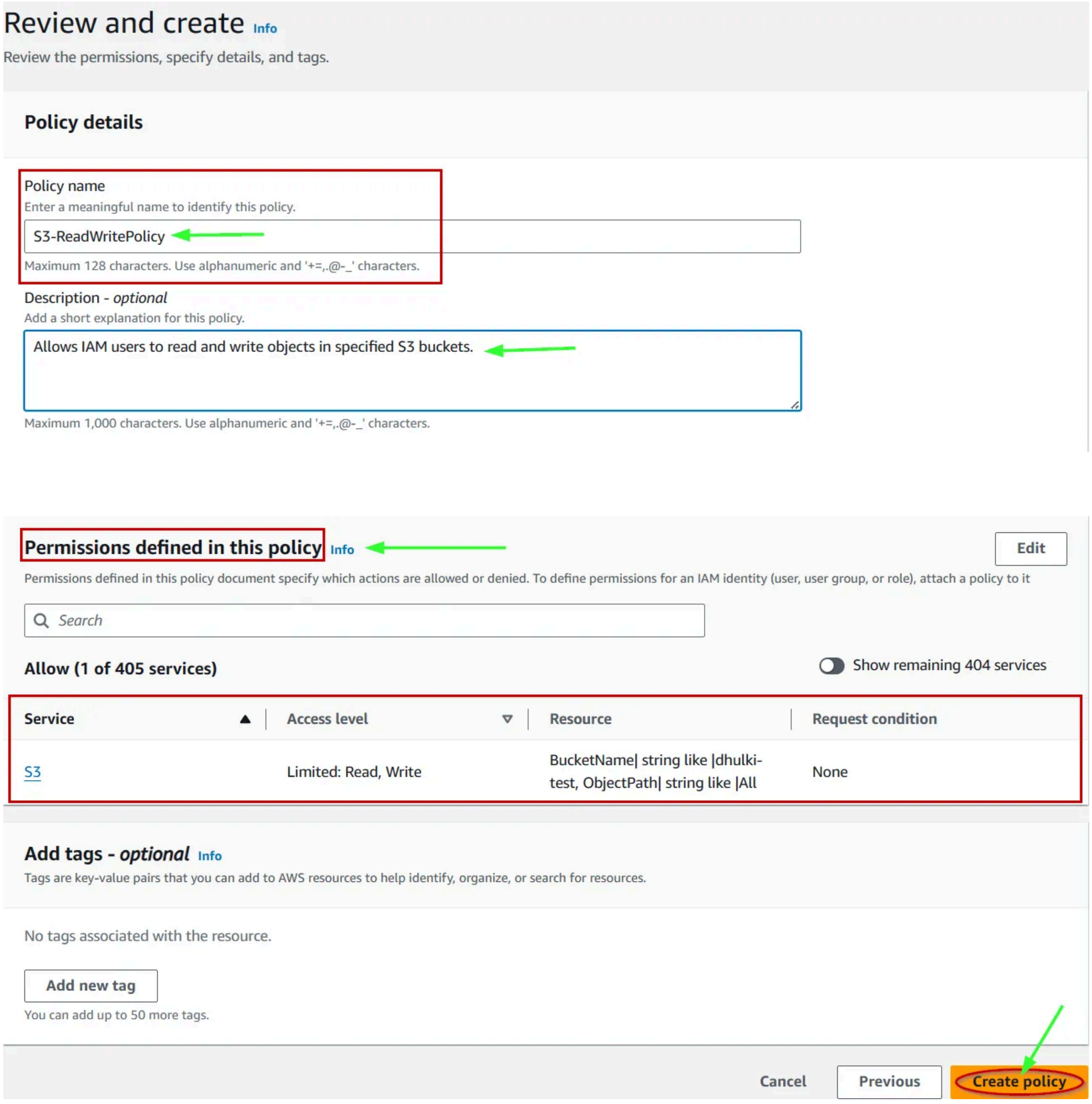
```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Statement1",
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:PutObject"
            ],
            "Resource": [
                "arn:aws:s3:::dhulki-test/*"
            ]
        }
    ]
}
```

> ### Additional Information
>
> *Version: The policy language version used is "2012–10–17".*
>
> *Statement: This section contains an array of statements, each describing a set of permissions.*
>
> *Effect: The policy allows access ("Allow") to the specified actions.*
>
> *Action: Specifies the allowed actions, which include* `s3:GetObject` *and* `s3:PutObject` *.*
>
> *Resource: Defines the AWS resource to which the policy applies, in this case, all objects (* `*` *) in the S3 bucket named "dhulki-test".*

You can customize this JSON code according to your specific requirements, such as changing the actions, resources, or adding additional statements as needed.

4. Review and Create Policy:

- Click on the "Review policy" button, provide a name and description for the policy, and then click "Create policy."


Review and create

Review the permissions, specify details, and tags.

Policy details

Policy name
Enter a meaningful name to identify this policy.
S3-ReadWritePolicy
Maximum 128 characters. Use alphanumeric and '+=,.@-_' characters.

Description - *optional*
Add a short explanation for this policy.
Allows IAM users to read and write objects in specified S3 buckets.
Maximum 1,000 characters. Use alphanumeric and '+=,.@-_' characters.

Permissions defined in this policy

Permissions defined in this policy document specify which actions are allowed or denied. To define permissions for an IAM identity (user, user group, or role), attach a policy to it

Allow (1 of 405 services)

Show remaining 404 services

Service	Access level	Resource	Request condition
S3	Limited: Read, Write	BucketName| string like |dhulki-test, ObjectPath| string like |All	None

Add tags - *optional*

Tags are key-value pairs that you can add to AWS resources to help identify, organize, or search for resources.

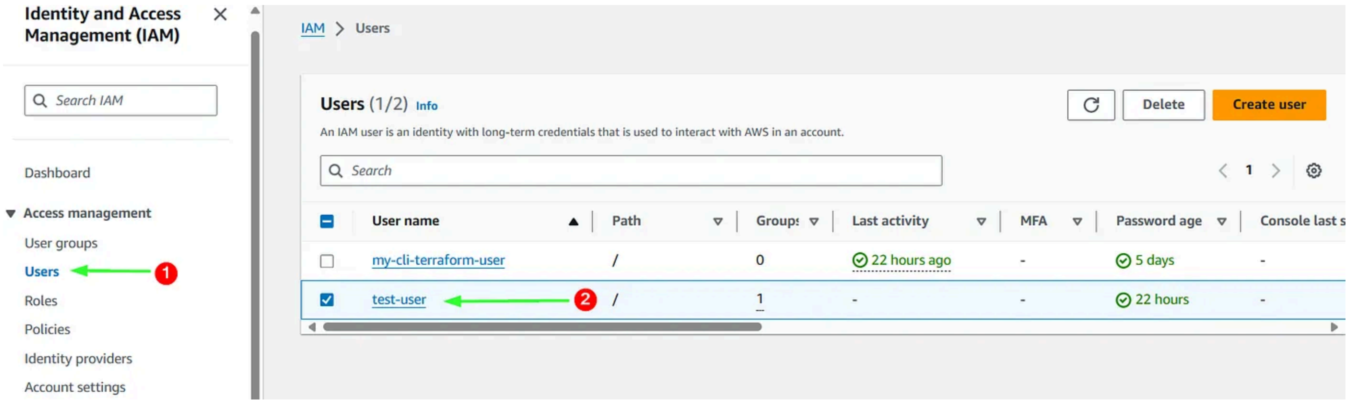
No tags associated with the resource.

Add new tag

You can add up to 50 more tags.

5. Attach Policy to IAM Entities:

- Once the policy is created, navigate to the IAM dashboard.
- Select the user, group, or role to which you want to attach the policy.

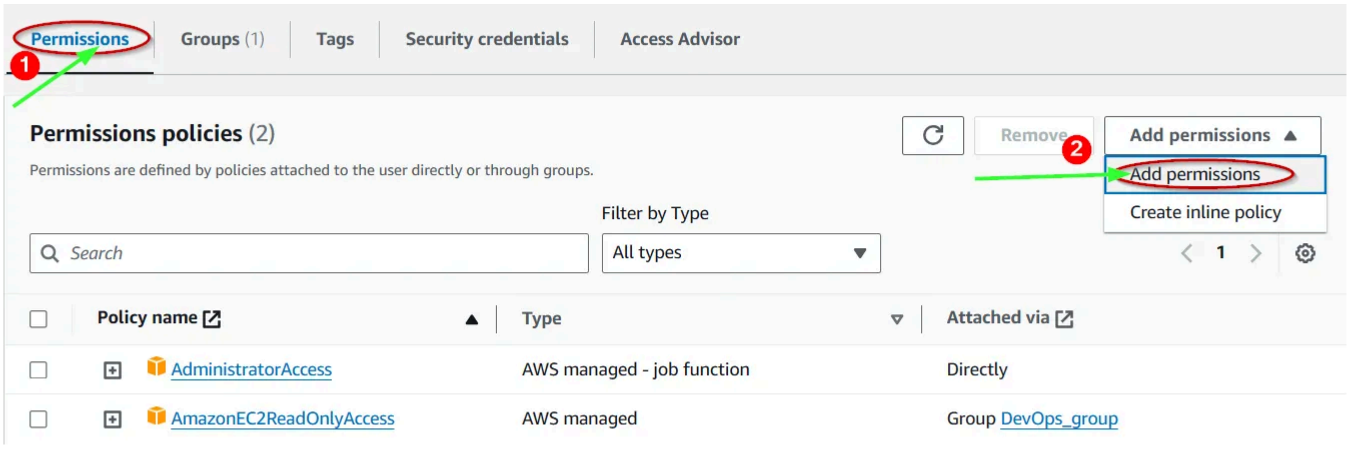


Users (1/2) Info

An IAM user is an identity with long-term credentials that is used to interact with AWS in an account.

User name	Path	Groups	Last activity	MFA	Password age	Console last s
my-cli-terraform-user	/	0	22 hours ago	-	5 days	-
test-user	/	1	-	-	22 hours	-

- In the permissions tab of the selected entity, click on “Attach policies.”

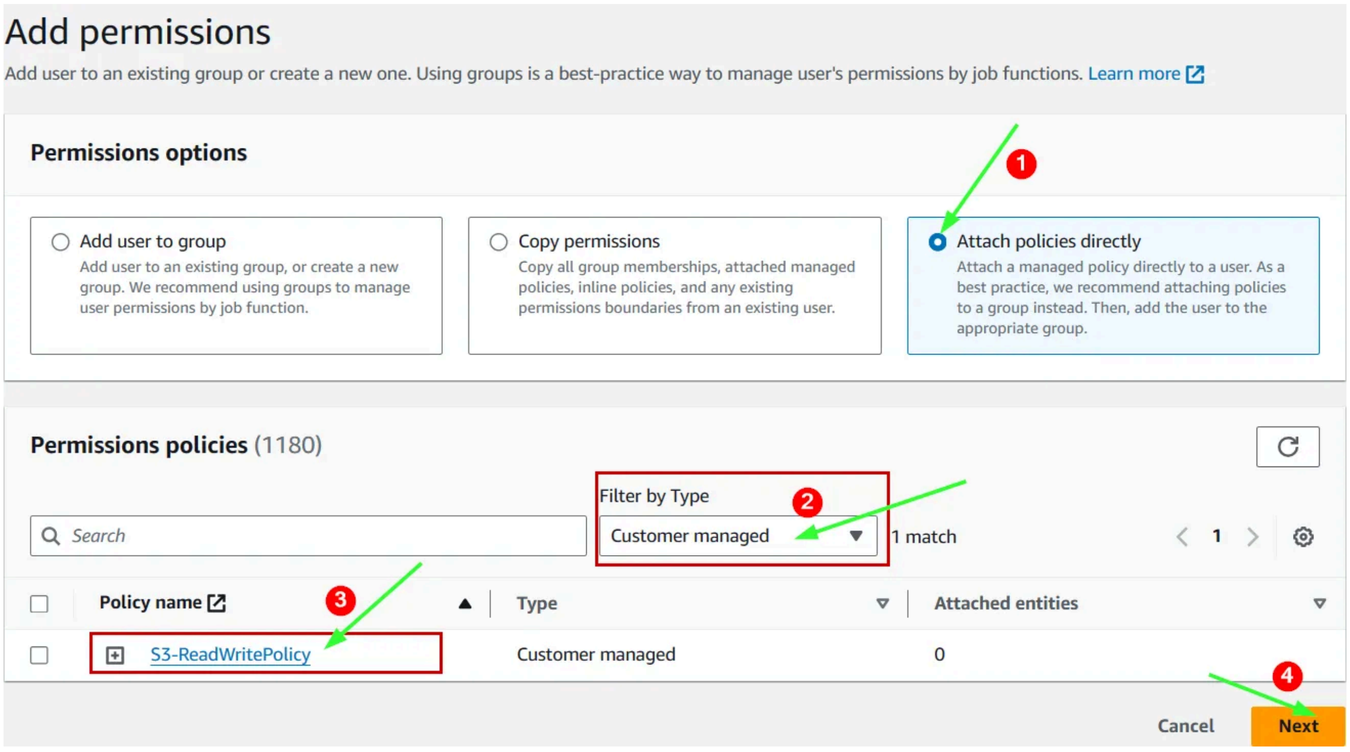


Permissions policies (2)

Permissions are defined by policies attached to the user directly or through groups.

Policy name	Type	Attached via
AdministratorAccess	AWS managed - job function	Directly
AmazonEC2ReadOnlyAccess	AWS managed	Group DevOps_group

- Search for and select the policy you created earlier, and then click on the “Attach policy” button.


Add permissions

Add user to an existing group or create a new one. Using groups is a best-practice way to manage user's permissions by job functions. Learn more

Permissions options

- Add user to group: Add user to an existing group, or create a new group. We recommend using groups to manage user permissions by job function.
- Copy permissions: Copy all group memberships, attached managed policies, inline policies, and any existing permissions boundaries from an existing user.
- [✓ Attach policies directly]: Attach a managed policy directly to a user. As a best practice, we recommend attaching policies to a group instead. Then, add the user to the group.

Permissions policies (1180)

Filter by Type: Customer managed — 1 match

Policy name	Type	Attached entities
S3-ReadWritePolicy	Customer managed	0

By following these steps, you can create and attach IAM policies to manage access to your S3 buckets efficiently.

Troubleshooting IAM Policy Editing Issues:

Users may encounter errors when attempting to save bucket policy changes if they lack the required "s3:PutBucketPolicy" permission. Follow these steps to resolve the issue:

1. Check IAM Policies: Navigate to the IAM console and select the user or role experiencing the issue.
2. Review Attached Policies: Check the attached policies for the selected user or role to ensure they include the necessary permissions.
3. Add Missing Permission: If the "s3:PutBucketPolicy" permission is missing, edit the existing policy or attach a new policy with the required permission.
4. Policy Example: Below is an example of how to add the "s3:PutBucketPolicy" permission to an existing IAM policy:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:PutBucketPolicy",
            "Resource": "*"
        }
    ]
}
```

Save Changes: Once the policy is updated to include the necessary permission, save the changes and retry editing the bucket policy in the S3 Management Console.

By following these steps, users can ensure they have the required permissions to edit bucket policies successfully, resolving any encountered issues effectively.

2. Bucket Policies Configuration:

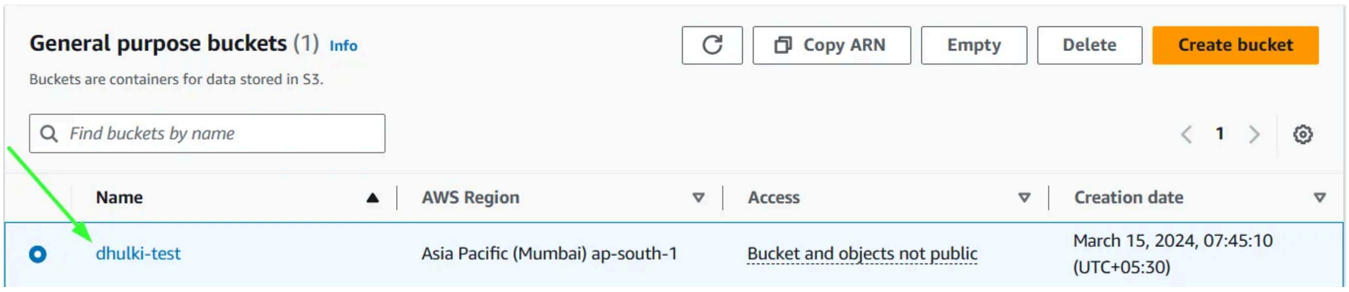
Bucket policies are crucial for controlling access to S3 buckets, defining permissions for users and applications. By configuring these policies, you enforce strict security measures, ensuring only authorized entities can interact with the bucket's contents. Effective implementation of bucket policies is essential for maintaining data integrity and compliance within AWS environments. Follow these steps to set up bucket policies effectively:

1. Navigate to the AWS Management Console:

- Sign in to your AWS account.
- In the top navigation bar, click on "Services" and select "S3" under the "Storage" section.

2. Select the Target Bucket:

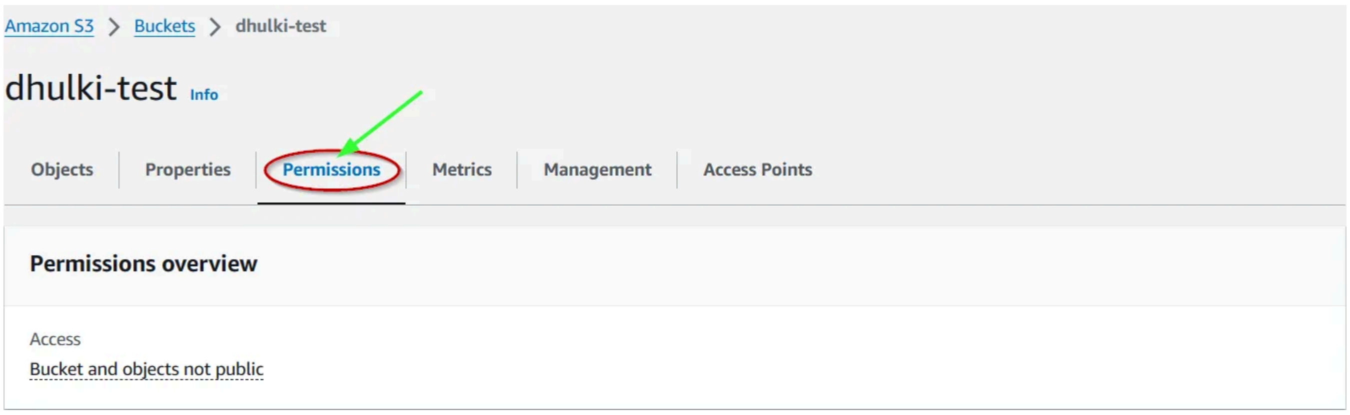
- In the S3 dashboard, locate and click on the name of the bucket for which you want to configure access control.


General purpose buckets (1) Info
Buckets are containers for data stored in S3.
Copy ARN
Empty
Delete
Create bucket
Find buckets by name
1

Name	AWS Region	Access	Creation date
dhulki-test	Asia Pacific (Mumbai) ap-south-1	Bucket and objects not public	March 15, 2024, 07:45:10 (UTC+05:30)

3. Access the Permissions Tab:

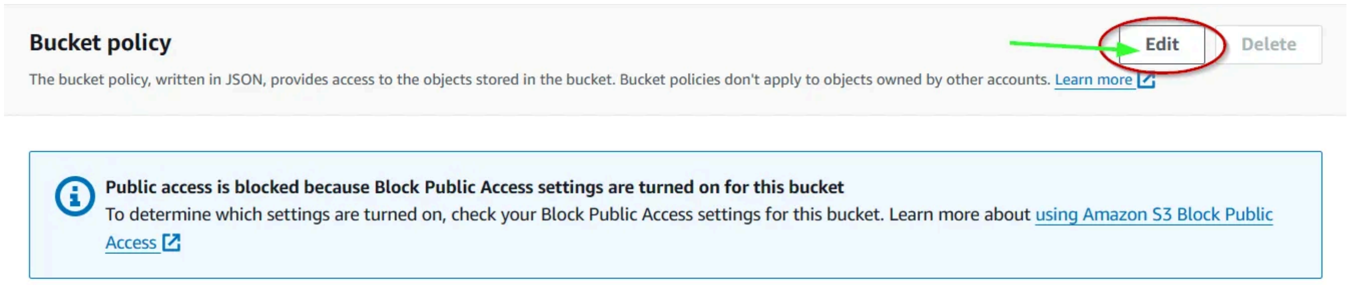
- Once inside the bucket, navigate to the "Permissions" tab.



Access
Bucket and objects not public

4. Edit Bucket Policy:

- Under the "Bucket Policy" section, click on the "Edit" button to modify the existing policy or create a new one.


Bucket policy

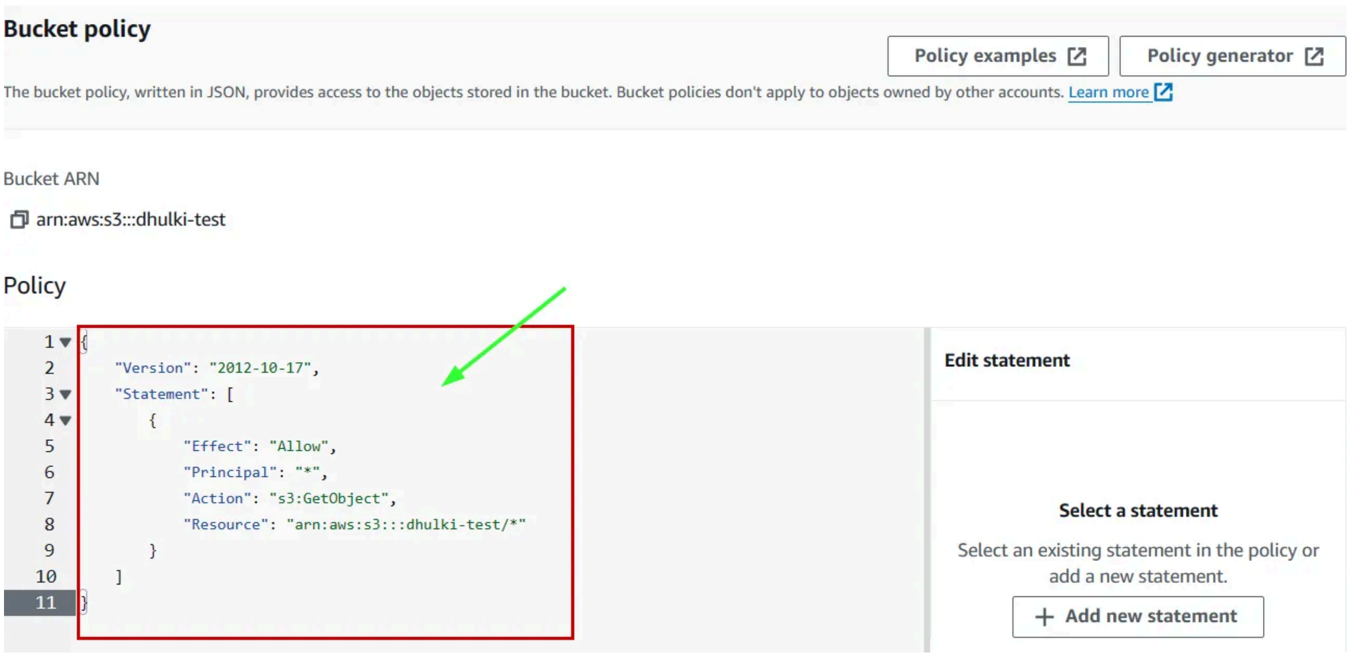
The bucket policy, written in JSON, provides access to the objects stored in the bucket. Bucket policies don't apply to objects owned by other accounts. Learn more

Public access is blocked because Block Public Access settings are turned on for this bucket

To determine which settings are turned on, check your Block Public Access settings for this bucket. Learn more about using Amazon S3 Block Public Access

5. Define the Bucket Policy:

- Write or paste the JSON code for the bucket policy in the editor.


Bucket policy

The bucket policy, written in JSON, provides access to the objects stored in the bucket. Bucket policies don't apply to objects owned by other accounts. Learn more

Bucket ARN

arn:aws:s3:::dhulki-test

Policy

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": "*",
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::dhulki-test/*"
        }
    ]
}
```

Edit statement

Select a statement

Select an existing statement in the policy or add a new statement.

- The JSON code should specify the permissions you want to grant or restrict for the selected bucket.

For example, here's a sample JSON code for a bucket policy:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": "*",
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::your-bucket-name/*"
        }
        ]
    }
```

> ***Additional Details:***
>
> ***Permission: It defines permissions for AWS resources.***
>
> ***Action: Specifically, "s3:GetObject" allows retrieving objects from an S3 bucket.***
>
> ***Effect: Set to "Allow," indicating permission is granted for the specified action.***
>
> ***Principal: "*" represents any entity, meaning the policy applies to all users, groups, and roles.***
>
> ***Resource: The ARN ("arn:aws:s3:::your-bucket-name/*") specifies the S3 bucket and objects within it.***

6. Save Changes:

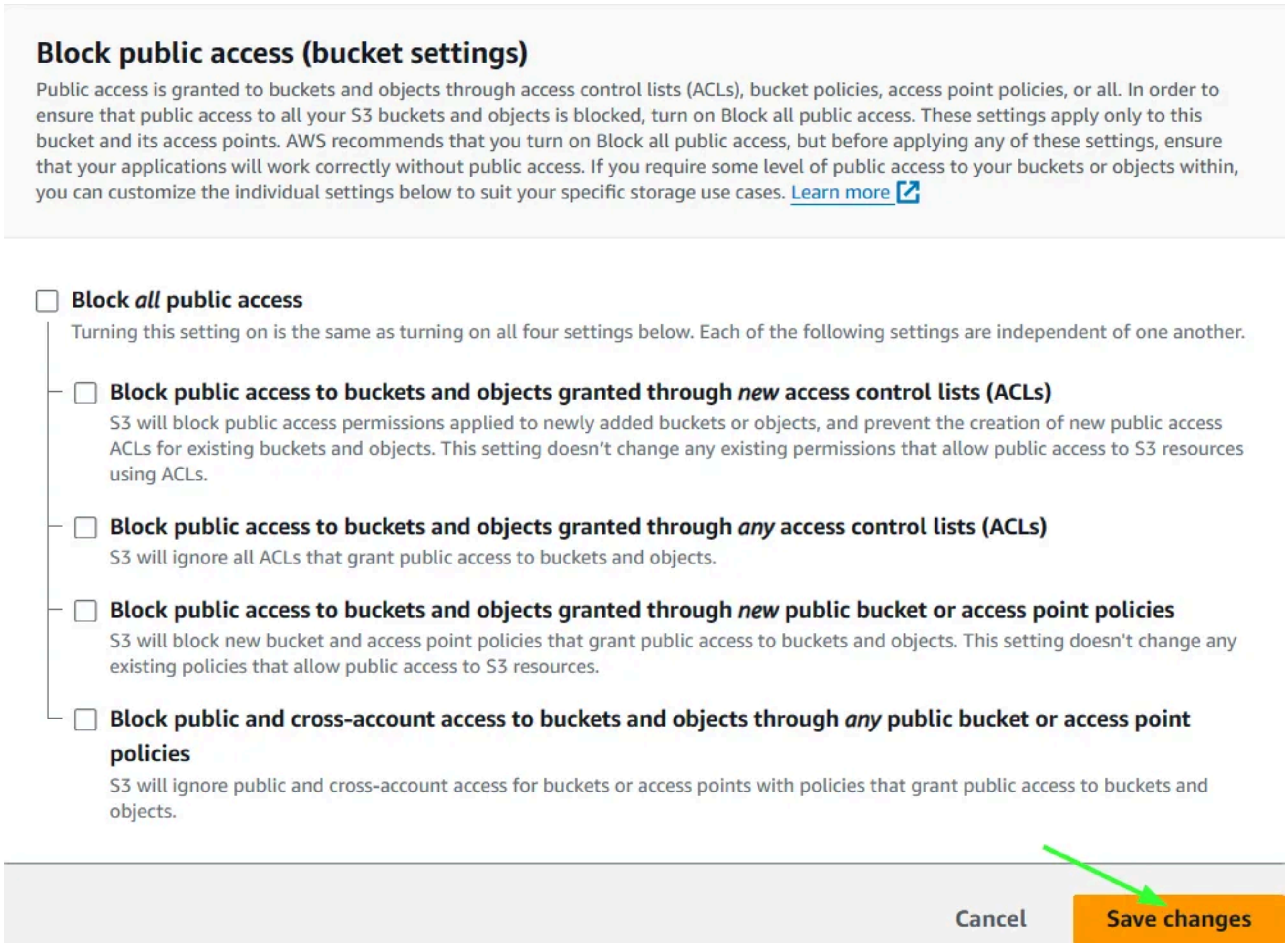
- After defining the bucket policy, click on the "Save changes" button to apply the policy to the bucket.

That's it! You have now successfully configured access control for your S3 bucket using IAM policies and bucket policies via the AWS Management Console.

Resolving Bucket Policy Editing Errors:

Users may encounter errors when attempting to save bucket policy changes due to conflicts with Block Public Access settings or insufficient permissions. Follow these steps to resolve the issue:

1. Review Block Public Access Settings: Navigate to the Block Public Access settings for your S3 bucket in the AWS Management Console.



Block public access (bucket settings)

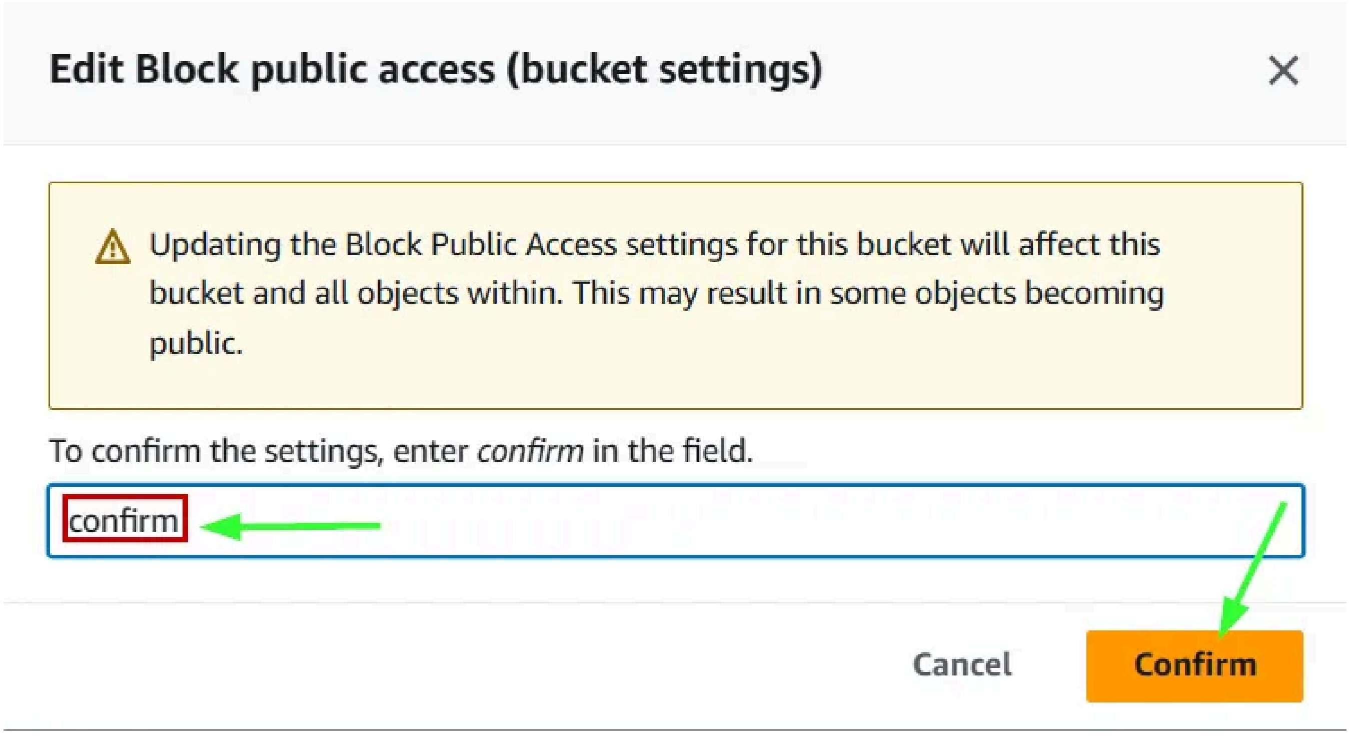
Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to all your S3 buckets and objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to your buckets or objects within, you can customize the individual settings below to suit your specific storage use cases. Learn more

☐ **Block *all* public access**
Turning this setting on is the same as turning on all four settings below. Each of the following settings are independent of one another.

- ☐ **Block public access to buckets and objects granted through *new* access control lists (ACLs)**
  S3 will block public access permissions applied to newly added buckets or objects, and prevent the creation of new public access ACLs for existing buckets and objects. This setting doesn't change any existing permissions that allow public access to S3 resources using ACLs.
- ☐ **Block public access to buckets and objects granted through *any* access control lists (ACLs)**
  S3 will ignore all ACLs that grant public access to buckets and objects.
- ☐ **Block public access to buckets and objects granted through *new* public bucket or access point policies**
  S3 will block new bucket and access point policies that grant public access to buckets and objects. This setting doesn't change any existing policies that allow public access to S3 resources.
- ☐ **Block public and cross-account access to buckets and objects through *any* public bucket or access point policies**
  S3 will ignore public and cross-account access for buckets or access points with policies that grant public access to buckets and objects.

2. *Check for Conflicts:* Review the Block Public Access settings to ensure they align with the intended access controls specified in the bucket policy.

3. *Adjust Settings if Necessary*: If the Block Public Access settings conflict with the permissions specified in the bucket policy, consider adjusting the settings to allow the desired level of access.


Edit Block public access (bucket settings)

Updating the Block Public Access settings for this bucket will affect this bucket and all objects within. This may result in some objects becoming public.

To confirm the settings, enter *confirm* in the field.

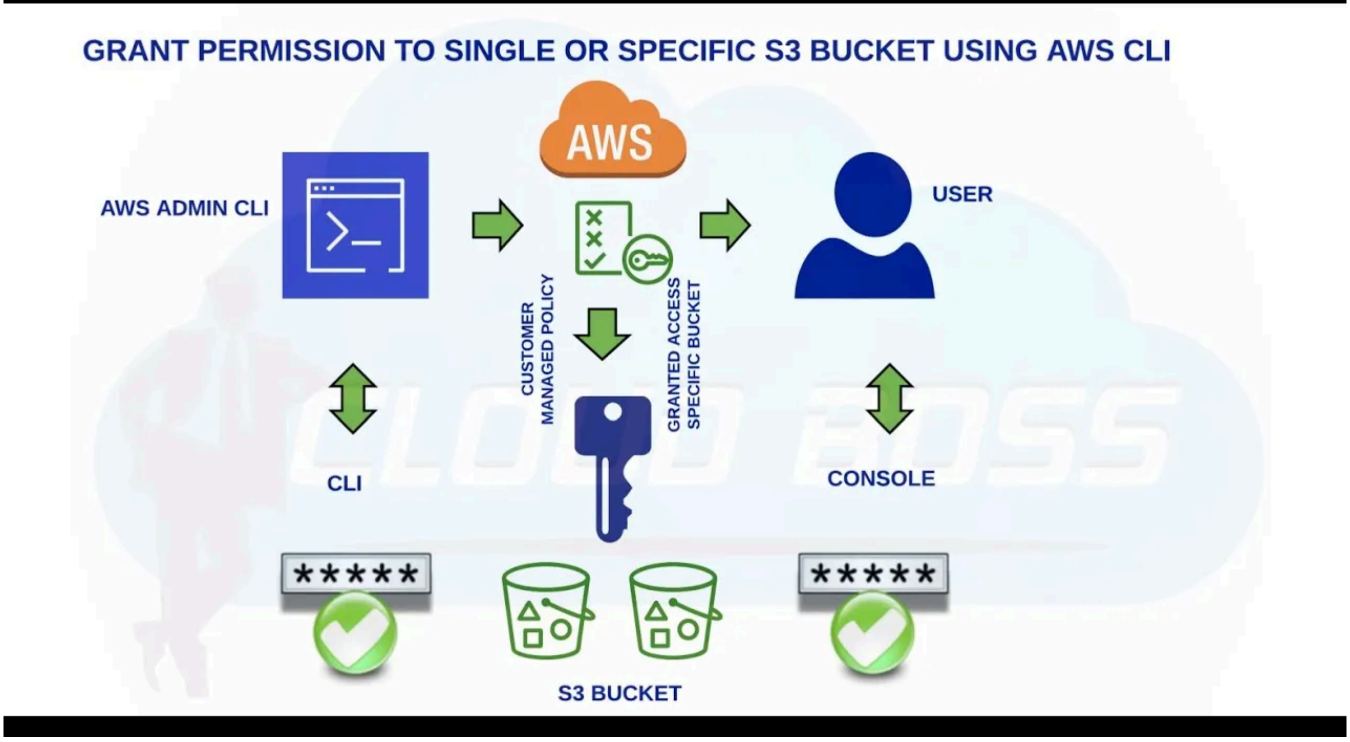
confirm

4. Modify Bucket Policy: Edit the bucket policy to ensure it aligns with any adjustments made to the Block Public Access settings and meets the security requirements of your use case.

Getting Started with IAM and Bucket Policies Configuration via Command Line Interface

Exploring IAM and Bucket Policies Configuration via Command Line Interface: Learn how to efficiently manage access control directly from your terminal with our step-by-step guide.

1. IAM Policies Configuration:


GRANT PERMISSION TO SINGLE OR SPECIFIC S3 BUCKET USING AWS CLI
AWS
AWS ADMIN CLI
USER
CUSTOMER MANAGED POLICY
GRANTED ACCESS SPECIFIC BUCKET
CLI
CONSOLE
S3 BUCKET

1. Creating a JSON Policy Document:

To create a JSON file containing the policy document, you can use a text editor such as Notepad, Visual Studio Code, or any other text editor of your choice. Follow these steps:

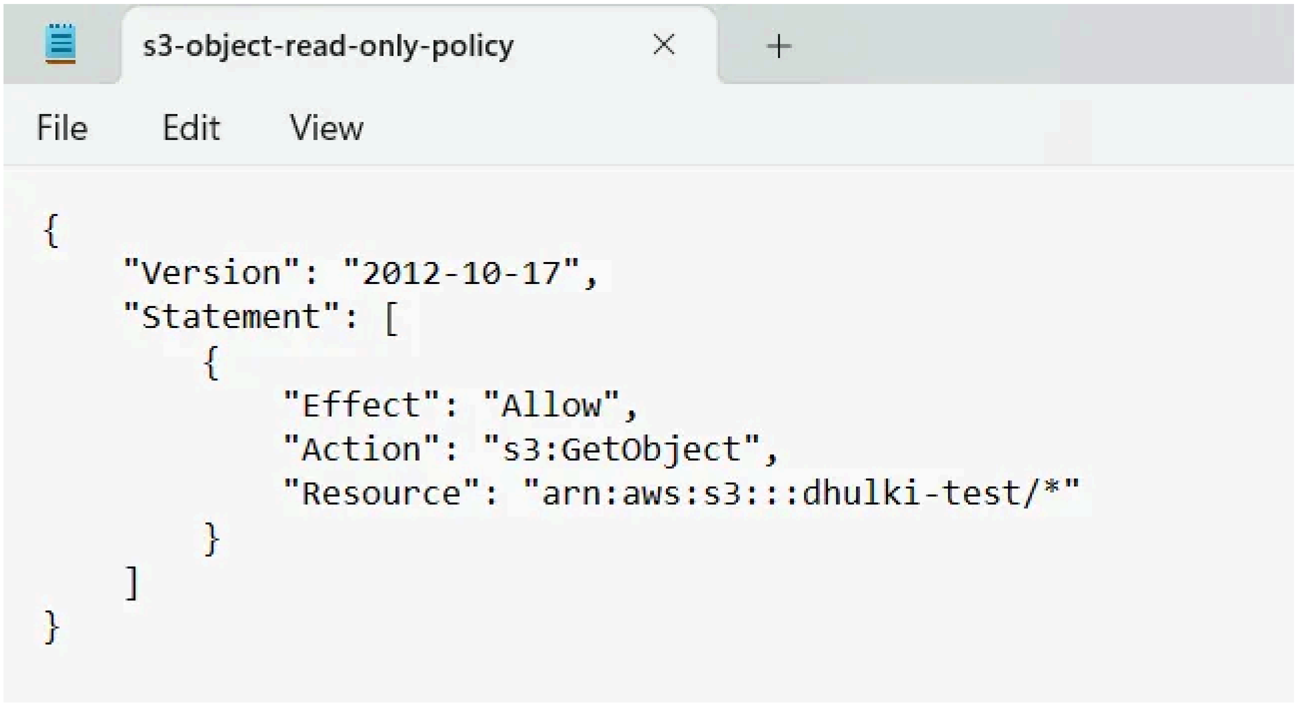
1. Open your preferred text editor.

2. Create a file containing the policy document and include the following content:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::dhulki-test/*"
        }
    ]
}
```

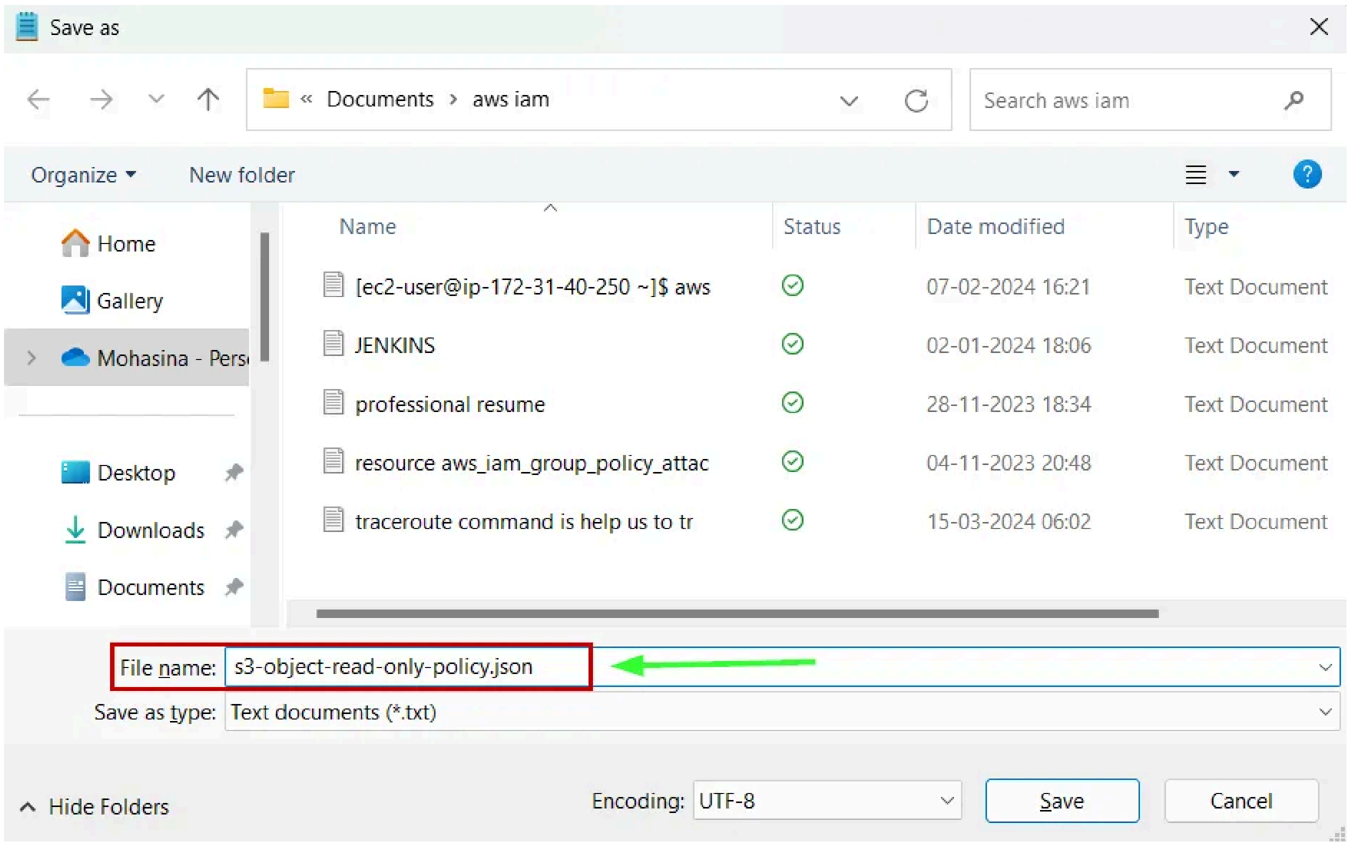
Replace `"dhulki-test"` with the name of your S3 bucket.

3. Paste the copied content into a new file in your text editor.



```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::dhulki-test/*"
        }
    ]
}
```

4. Save the file with the name `s3-object-read-only-policy.json`. Ensure that you save it with the `.json` extension and that the file format is set to plain text or JSON.



Name	Status	Date modified	Type
[ec2-user@ip-172-31-40-250 ~]$ aws	✓	07-02-2024 16:21	Text Document
JENKINS	✓	02-01-2024 18:06	Text Document
professional resume	✓	28-11-2023 18:34	Text Document
resource aws_iam_group_policy_attac	✓	04-11-2023 20:48	Text Document
traceroute command is help us to tr	✓	15-03-2024 06:02	Text Document

File name: s3-object-read-only-policy.json
Save as type: Text documents (*.txt)
Encoding: UTF-8

Once you've saved the JSON file, you can proceed with the next steps to use the AWS CLI to create the IAM policy using the `aws iam create-policy` command.

2. Using AWS CLI to Create IAM Policy:

To create an IAM policy using the AWS CLI, execute the following command:

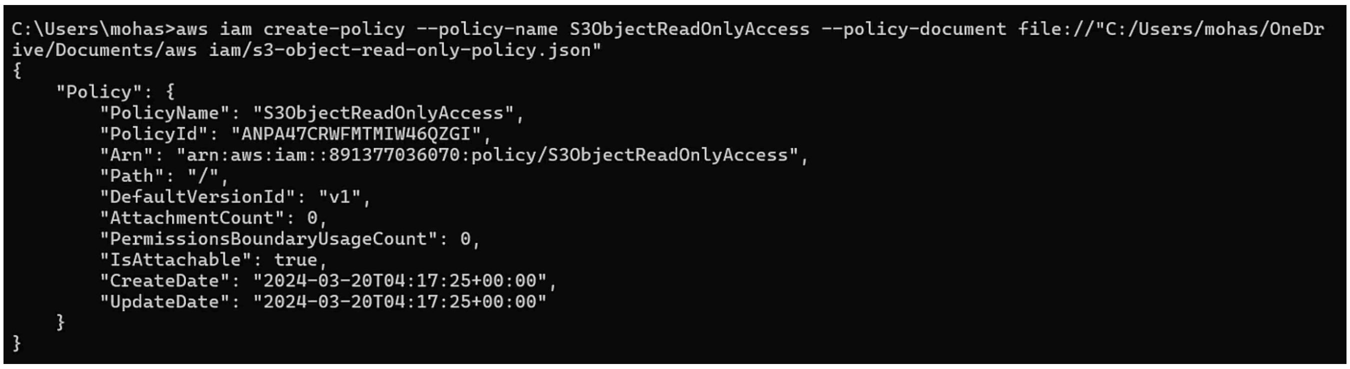
```
aws iam create-policy --policy-name MyPolicy --policy-document file://path/to/p
```

- Provide the file path to the policy document JSON file using `file://path/to/policy.json` prefix.

For Example:

```
aws iam create-policy --policy-name S3ObjectReadOnlyAccess --policy-document fi
```

This command creates an IAM policy named "S3ObjectReadOnlyAccess" using the policy document from the JSON file.



```
C:\Users\mohas>aws iam create-policy --policy-name S3ObjectReadOnlyAccess --policy-document file://"C:/Users/mohas/OneDr
ive/Documents/aws iam/s3-object-read-only-policy.json"
{
    "Policy": {
        "PolicyName": "S3ObjectReadOnlyAccess",
        "PolicyId": "ANPA47CRWFMTMIW46QZGI",
        "Arn": "arn:aws:iam::891377036070:policy/S3ObjectReadOnlyAccess",
        "Path": "/",
        "DefaultVersionId": "v1",
        "AttachmentCount": 0,
        "PermissionsBoundaryUsageCount": 0,
        "IsAttachable": true,
        "CreateDate": "2024-03-20T04:17:25+00:00",
        "UpdateDate": "2024-03-20T04:17:25+00:00"
    }
}
```

3. Attaching Policy to IAM User:

Once the policy is created, you can attach it to IAM users, groups, or roles as needed using the `aws iam attach-user-policy`, `aws iam attach-group-policy`, or `aws iam attach-role-policy` command, respectively.

1. To attach the newly created policy to an IAM user, you can use the `attach-user-policy` command. Here's the general syntax:

```
aws iam attach-user-policy --policy-arn <policy-arn> --user-name <user-name>
```

- Replace `<policy-arn>` with the Amazon Resource Name (ARN) of the policy you just created and `<user-name>` with the name of the IAM user you want to attach the policy to.

For example, If the policy ARN is `arn:aws:iam::ACCOUNT_ID_HERE:policy/S3ObjectReadOnlyAccess`, and you want to attach it to a specific IAM user,let's say `mohasina`, you can use the following command:

```
aws iam attach-user-policy --policy-arn arn:aws:iam::ACCOUNT_ID_HERE:policy/S3C
```

Replace `ACCOUNT_ID_HERE` with your actual AWS account ID. This command will attach the `S3ObjectReadOnlyAccess` policy to the IAM user `mohasina`.

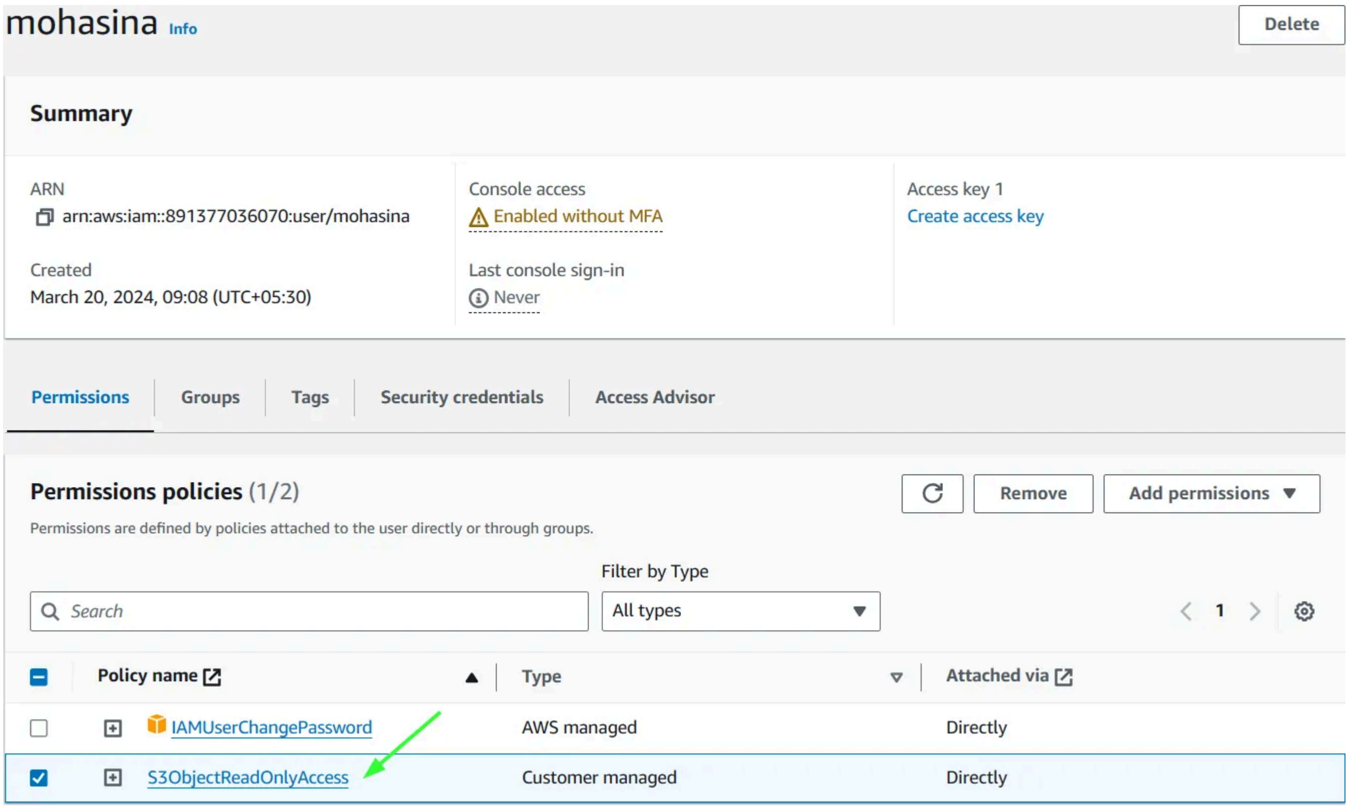
That's it! You have successfully created a custom IAM policy for S3 object read-only access using the AWS CLI.

4. Verify Policy Attachment:

Confirm that the IAM policy has been successfully attached to the specified IAM user by checking the AWS Management Console or running additional AWS CLI commands.
To ensure the IAM policy has been successfully attached to the specified IAM user, follow these steps:

1. AWS Management Console:

- Navigate to the IAM dashboard in the AWS Management Console.
- Select "Users" from the sidebar menu.
- Choose the IAM user you attached the policy to.
- Under the "Permissions" tab, verify that the policy is listed.


mohasina Info

Delete

Summary

ARN	Console access	Access key 1
arn:aws:iam::891377036070:user/mohasina	Enabled without MFA	Create access key
Created	Last console sign-in	
March 20, 2024, 09:08 (UTC+05:30)	Never	

Permissions | Groups | Tags | Security credentials | Access Advisor

Permissions policies (1/2)

Permissions are defined by policies attached to the user directly or through groups.

Filter by Type: All types

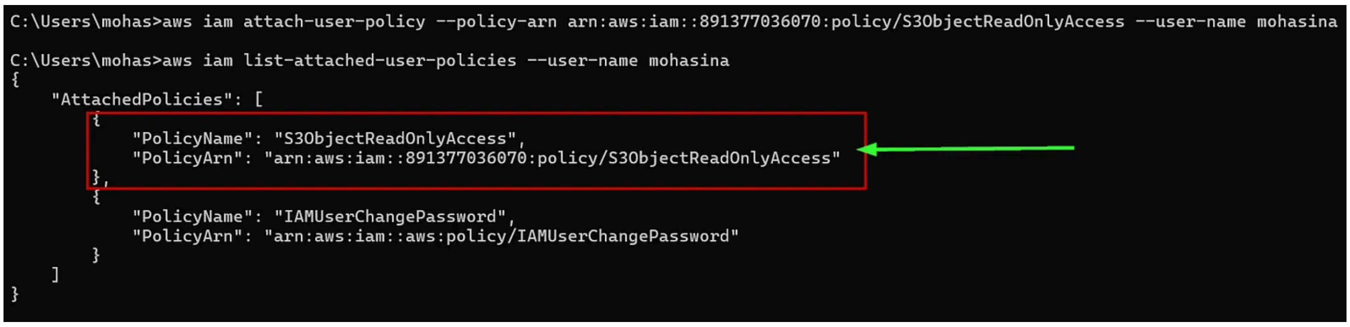
	Policy name	Type	Attached via
☐	IAMUserChangePassword	AWS managed	Directly
☑	S3ObjectReadOnlyAccess	Customer managed	Directly

2. AWS CLI:

- Run the following command to list the attached policies for the IAM user:

```
aws iam list-attached-user-policies --user-name mohasina
```

- Verify that the output includes the newly attached policy.


```
C:\Users\mohas>aws iam attach-user-policy --policy-arn arn:aws:iam::891377036070:policy/S3ObjectReadOnlyAccess --user-name mohasina

C:\Users\mohas>aws iam list-attached-user-policies --user-name mohasina
{
    "AttachedPolicies": [
        {
            "PolicyName": "S3ObjectReadOnlyAccess",
            "PolicyArn": "arn:aws:iam::891377036070:policy/S3ObjectReadOnlyAccess"
        },
        {
            "PolicyName": "IAMUserChangePassword",
            "PolicyArn": "arn:aws:iam::aws:policy/IAMUserChangePassword"
        }
    ]
}
```

By performing these checks, you can confirm that the IAM policy has been successfully attached to the specified IAM user.

2. Bucket Policies Configuration:

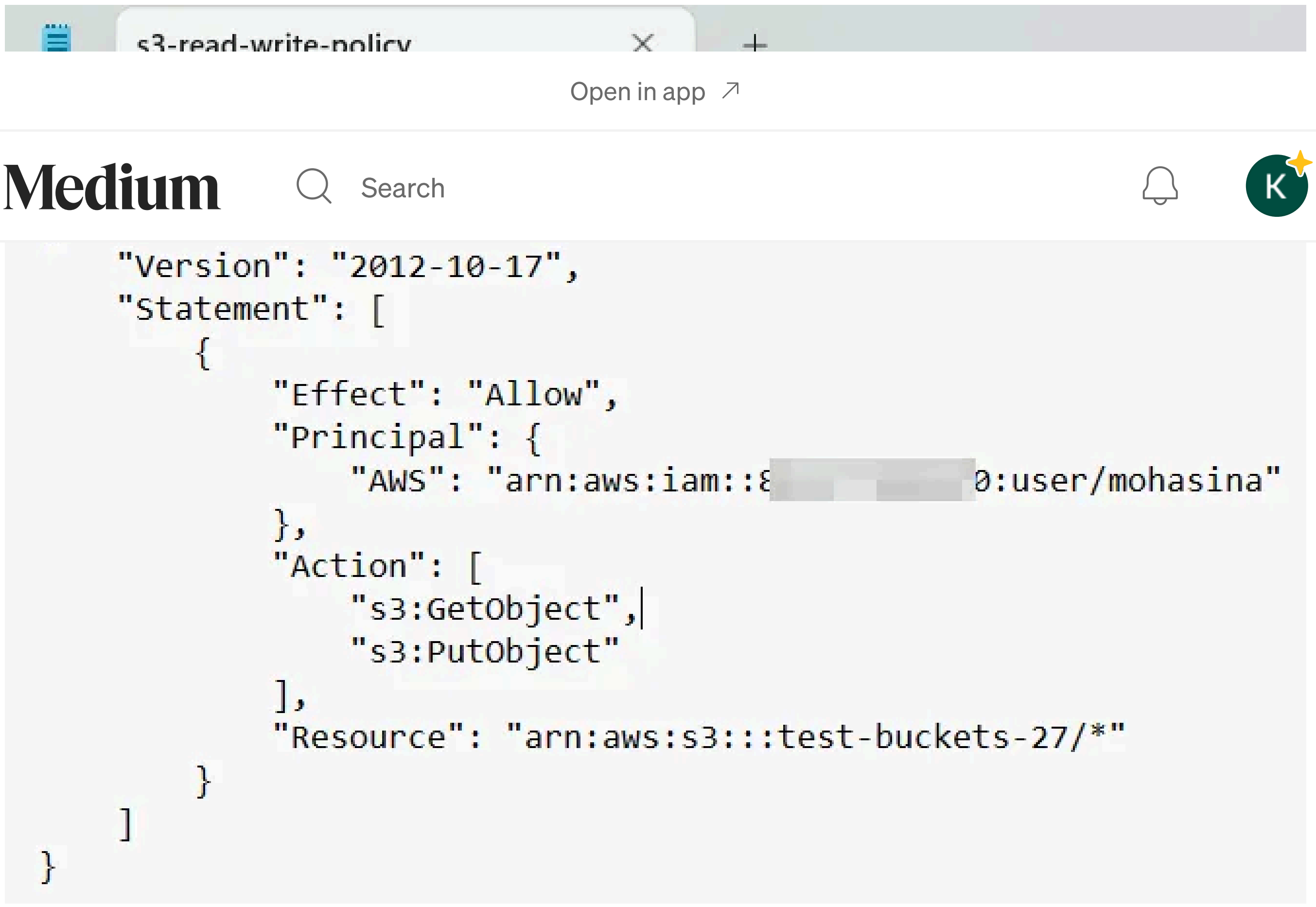
To configure bucket policies for your S3 buckets using the AWS CLI, follow these steps:

1. Creating a JSON Policy Document:

1. Create a file containing the policy document and include the following content:

Here's an example of an S3 bucket policy that allows a specific IAM user to perform both `GetObject` and `PutObject` actions on objects within the bucket:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::USER_ID:user/your-username"
            },
            "Action": [
                "s3:GetObject",
                "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::your-bucket-name/*"
        }
    ]
}
```



```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::8[redacted]0:user/mohasina"
            },
            "Action": [
                "s3:GetObject",
                "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::test-buckets-27/*"
        }
    ]
}
```

> *Explanation:*
>
> `"Effect": "Allow"` *: Specifies that the defined actions are allowed.*
>
> `"Principal": {"AWS": "arn:aws:iam::USER_ID:user/your-username"}` *: Restricts the permission to a specific IAM user identified by their ARN. Replace* `"USER_ID"` *with the actual AWS account ID and* `"your-username"` *with the username of the IAM user.*
>
> `"Action": ["s3:GetObject", "s3:PutObject"]` *: Specifies the actions allowed for the user, which are* `GetObject` *(read) and* `PutObject` *(write).*
>
> `"Resource": "arn:aws:s3:::your-bucket-name/*"` *: Defines the ARN of the bucket and objects to which the policy applies. Replace* `"your-bucket-name"` *with your actual bucket name. The* `/*` *at the end allows actions on any object within the bucket.*

2. Using AWS CLI to Create Bucket Policy:

To create a bucket policy using the AWS Command Line Interface (CLI), follow these steps:

1. Open your terminal or command prompt.

2. Execute the following command, replacing the placeholders with actual values:

```
aws s3api put-bucket-policy --bucket YOUR_BUCKET_NAME --policy file://path/to/b
```

Replace `YOUR_BUCKET_NAME` with the name of your S3 bucket.

Provide the file path to the bucket policy JSON file using `file://` prefix.

3. Uploading Bucket Policy:

For example:

```
aws s3api put-bucket-policy --bucket my-bucket --policy file://path/to/bucket-p
```

This command uploads the bucket policy specified in the JSON file to the S3 bucket.

Once the command is executed successfully, the bucket policy will be applied to the specified S3 bucket.

4. Verify Policy Application:

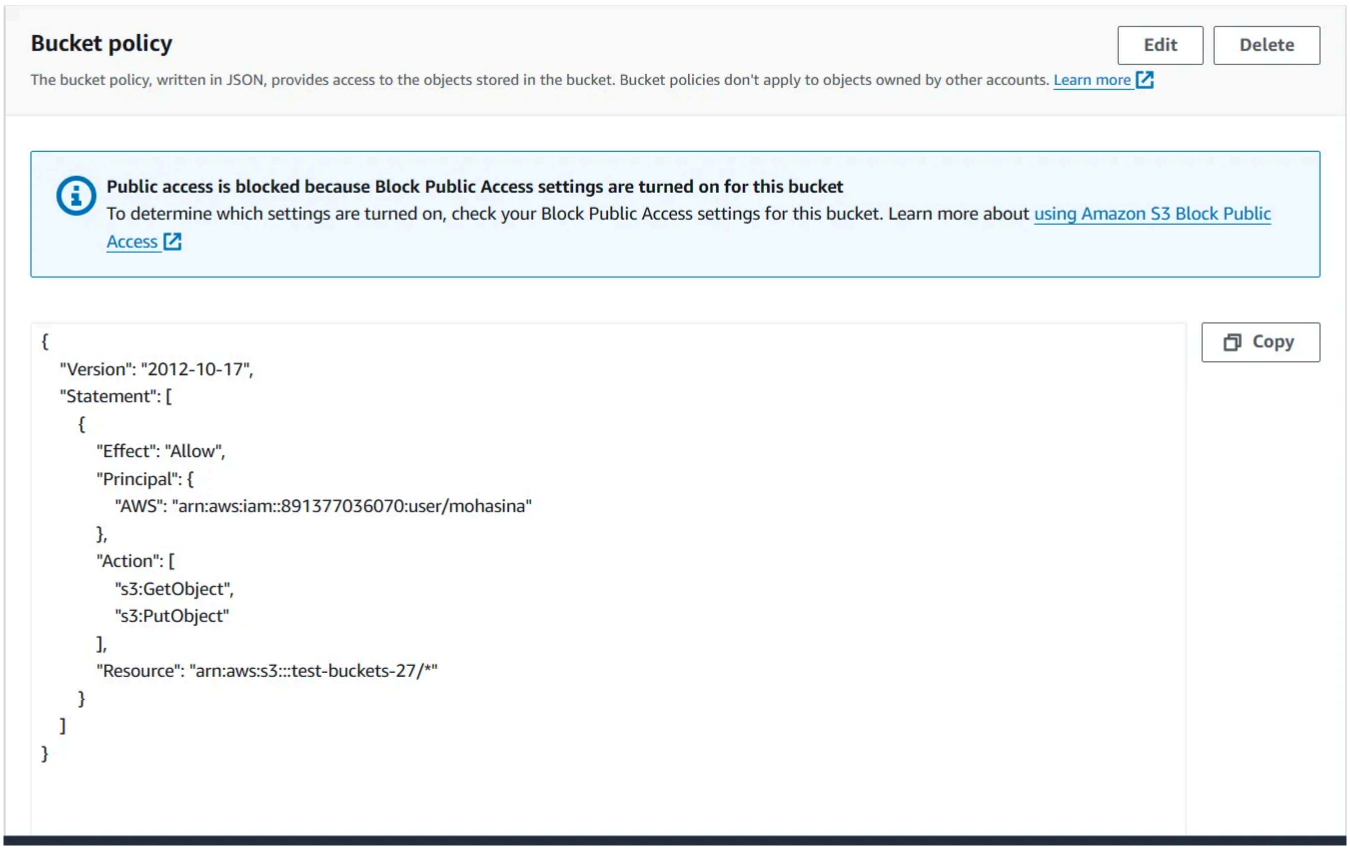
You can confirm that the bucket policy has been successfully applied to the specified S3 bucket by checking the AWS Management Console or running additional AWS CLI commands.

To verify that the bucket policy has been successfully applied to the specified S3 bucket, follow these steps:

1. Check AWS Management Console:

- Log in to the AWS Management Console.
- Navigate to the S3 service.
- Select the bucket for which you applied the policy.
- Navigate to the "Permissions" tab.

- Review the bucket policy to ensure it reflects the desired permissions.



Bucket policy

The bucket policy, written in JSON, provides access to the objects stored in the bucket. Bucket policies don't apply to objects owned by other accounts. Learn more

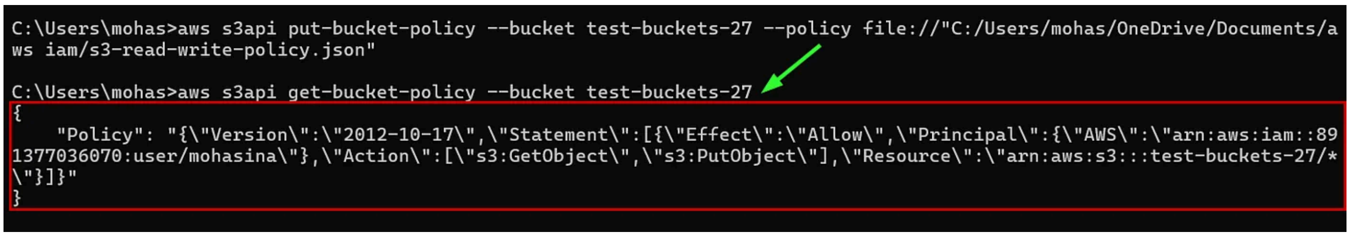
Public access is blocked because Block Public Access settings are turned on for this bucket
To determine which settings are turned on, check your Block Public Access settings for this bucket. Learn more about using Amazon S3 Block Public Access

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::891377036070:user/mohasina"
      },
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": "arn:aws:s3:::test-buckets-27/*"
    }
  ]
}
```

2. Run Additional AWS CLI Commands:

- Open your terminal or command prompt.
- Execute AWS CLI commands to list bucket policies or retrieve policy details:

```
aws s3api get-bucket-policy --bucket YOUR_BUCKET_NAME
```



```
C:\Users\mohas>aws s3api put-bucket-policy --bucket test-buckets-27 --policy file://"C:/Users/mohas/OneDrive/Documents/a
ws iam/s3-read-write-policy.json"

C:\Users\mohas>aws s3api get-bucket-policy --bucket test-buckets-27
{
    "Policy": "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect\":\"Allow\",\"Principal\":{\"AWS\":\"arn:aws:iam::89
1377036070:user/mohasina\"},\"Action\":[\"s3:GetObject\",\"s3:PutObject\"],\"Resource\":\"arn:aws:s3:::test-buckets-27/*
\"}]}"
}
```

- Ensure that the command output displays the applied bucket policy without errors.
- Verify that the policy matches the intended permissions and configurations.

By following these step-by-step instructions, users can configure IAM policies and bucket policies using the AWS CLI, ensuring proper access control and security configurations for their AWS resources.

> ***Note:***
>
> *IAM Policies Configuration: Use the* `aws iam put-policy` *command to attach IAM policies to users/groups/roles.*
>
> *Bucket Policies Configuration: Utilize the* `aws s3api put-bucket-policy` *command to apply bucket policies to S3 buckets.*

Getting Started: IAM and Bucket Policies Configuration with Terraform

Embarking on IAM and Bucket Policies Configuration with Terraform: Discover the streamlined process of setting up access control using Terraform in this comprehensive guide. Let's dive in!

1. IAM Policies Configuration with Terraform:

To define an IAM policy allowing read and write access to S3 buckets using Terraform's `aws_iam_policy` resource, follow these steps:

1. Declare an IAM policy using the `aws_iam_policy` resource in your Terraform configuration.
2. Assign a name and description to your IAM policy. The `name` attribute provides a unique identifier for the policy, while `description` offers a brief explanation of its purpose.
3. Define the policy document using the `policy` attribute. The policy document is written in JSON format and specifies the permissions granted by the policy. In this example, we're allowing the `s3:GetObject` and `s3:PutObject` actions on objects within a specific S3 bucket.
4. Use the `jsonencode` function to convert the policy document from HCL (HashiCorp Configuration Language) to JSON format, as required by Terraform.
5. Complete the IAM policy definition by specifying the version and statements within the policy document. Each statement describes a set of permissions,

including the effect (allow or deny), actions, resources, and optionally, the principal entities (users, roles, or accounts) to which the policy applies.

Here's an example of how the Terraform configuration might look:

```
resource "aws_iam_policy" "s3_read_write_policy" {
  name        = "s3-read-write-access"
  path        = "/"
  description = "Allows read and write access to S3 buckets"
  policy = jsonencode({
    Version = "2012-10-17"
    Statement = [
      {
        Effect = "Allow"
        Action = [
          "S3:GetObject",
          "S3:PutObject"
        ]
        Resource = "*"
      },
    ]
  })

}
```

In this example, the IAM policy named `s3-read-write-access` grants permissions to perform `s3:GetObject` (read) and `s3:PutObject` (write) actions on any S3 bucket.

Once you've defined the IAM policy in your Terraform configuration, you can proceed to apply the changes using the Terraform CLI.

2. Verify the Changes:

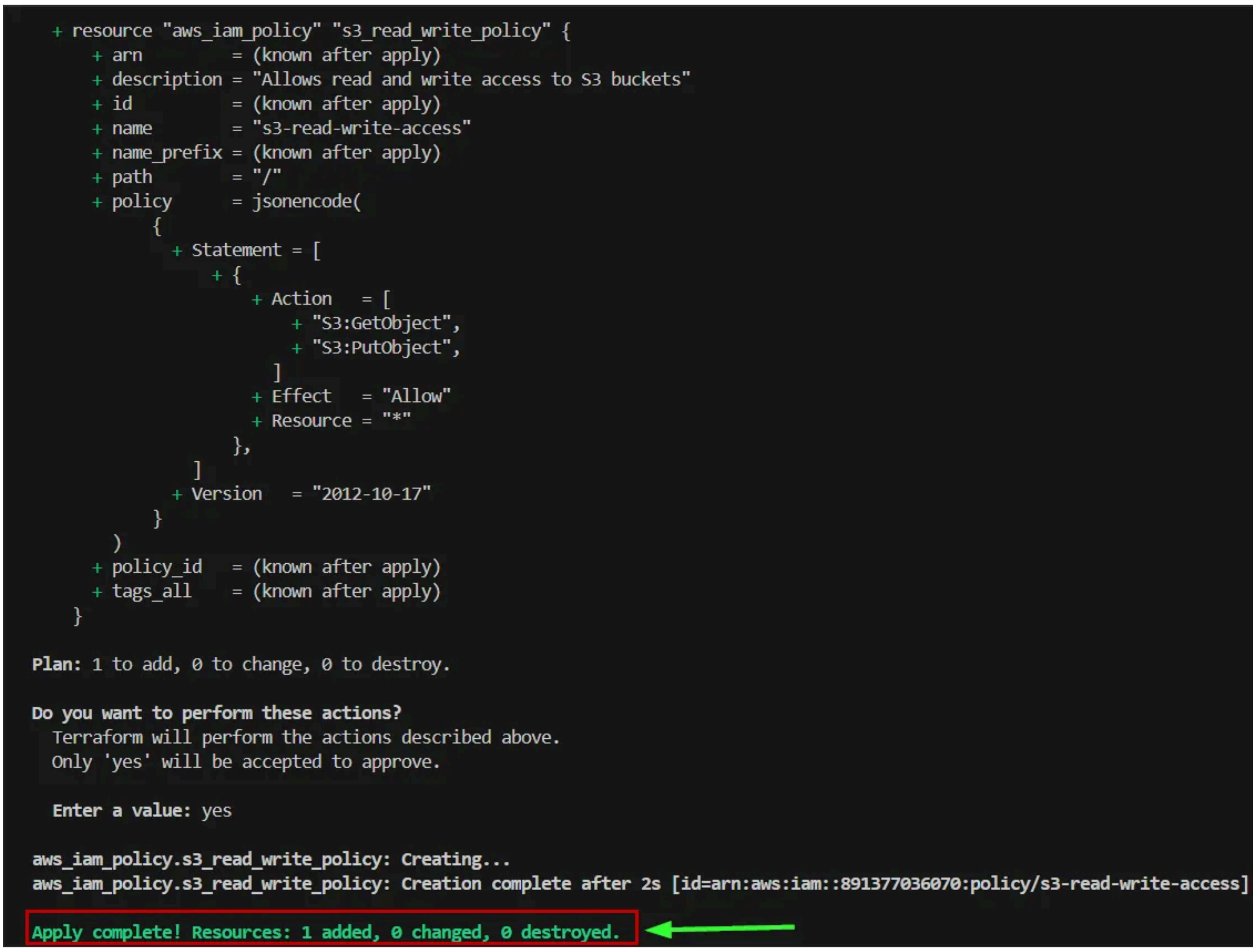
After applying the Terraform configuration, it's important to verify that the IAM policy has been created or updated as intended. Here's how you can do it:

1. Check Terraform Output:

Upon running `terraform apply`, review the output provided by Terraform. Look for messages related to the IAM policy resource to confirm if it was created, updated, or destroyed.

```
PS C:\Users\mohas\my_terraform_project\terraform-iam> terraform apply
aws_iam_group.my_user_group: Refreshing state... [id=DevOps_group]
aws_iam_user_login_profile.consol_access_profile: Refreshing state... [id=test-
aws_iam_user.my_iam_user: Refreshing state... [id=test-user]
aws_iam_role.my_assume_role: Refreshing state... [id=ec2-assume-role]
aws_iam_user_policy_attachment.my_iam_user_policy_attachment: Refreshing state.
aws_iam_group_policy_attachment.my_iam_user_group_policy_attachment: Refreshing
aws_iam_user_group_membership.my_iam_user_group_membership: Refreshing state...
aws_iam_role_policy_attachment.my_assume_role_policy_attachment: Refreshing sta

Terraform used the selected providers to generate the following execution plan.
following symbols:
  + create

Terraform will perform the following actions:

  # aws_iam_policy.s3_read_write_policy will be created
  + resource "aws_iam_policy" "s3_read_write_policy" {
      + arn         = (known after apply)
      + description = "Allows read and write access to S3 buckets"
      + id          = (known after apply)
      + name        = "s3-read-write-access"
      + name_prefix = (known after apply)
      + path        = "/"
      + policy      = jsonencode(
            {
              + Statement = [
                  + {
                      + Action   = [
                          + "S3:GetObject",
                          + "S3:PutObject",
                        ]
                      + Effect   = "Allow"
                      + Resource = "*"
                    },
                ]
              + Version   = "2012-10-17"
            }
        )
      + policy_id   = (known after apply)
      + tags_all    = (known after apply)
    }

Plan: 1 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes
```

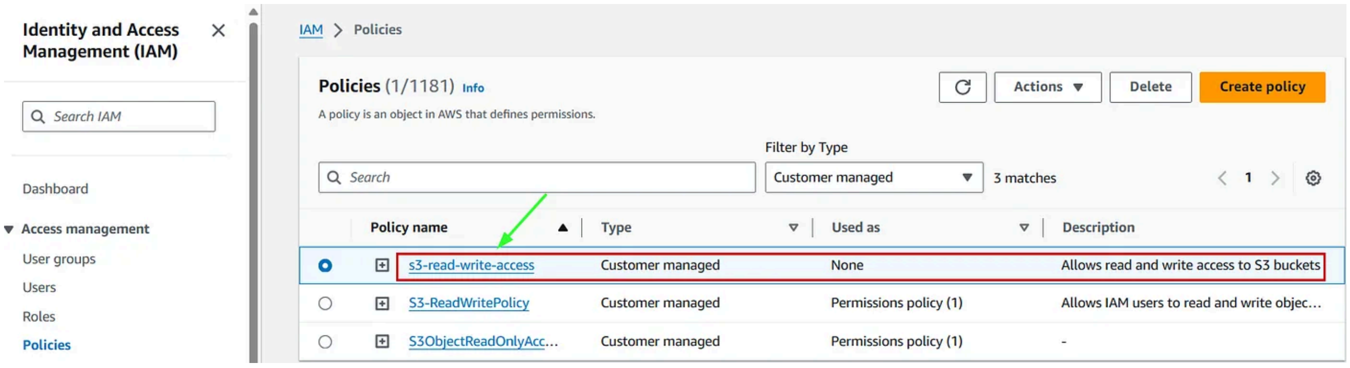
```
aws_iam_policy.s3_read_write_policy: Creating...
aws_iam_policy.s3_read_write_policy: Creation complete after 2s [id=arn:aws:iam

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
```



```
  + resource "aws_iam_policy" "s3_read_write_policy" {
      + arn         = (known after apply)
      + description = "Allows read and write access to S3 buckets"
      + id          = (known after apply)
      + name        = "s3-read-write-access"
      + name_prefix = (known after apply)
      + path        = "/"
      + policy      = jsonencode(
            {
              + Statement = [
                  + {
                      + Action   = [
                          + "S3:GetObject",
                          + "S3:PutObject",
                        ]
                      + Effect   = "Allow"
                      + Resource = "*"
                    },
                ]
              + Version   = "2012-10-17"
            }
        )
      + policy_id   = (known after apply)
      + tags_all    = (known after apply)
    }

Plan: 1 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes

aws_iam_policy.s3_read_write_policy: Creating...
aws_iam_policy.s3_read_write_policy: Creation complete after 2s [id=arn:aws:iam::891377036070:policy/s3-read-write-access]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
```

2. Verify in AWS Console:

Go to the AWS Management Console and navigate to the IAM service. Locate the IAM policy with the specified name (e.g., “s3-read-write-access”). Ensure that it exists and matches the configuration you provided in Terraform, including the description and policy document.


Identity and Access Management (IAM)

IAM > Policies

Policies (1/1181) Info

A policy is an object in AWS that defines permissions.

Filter by Type: Customer managed — 3 matches

Policy name	Type	Used as	Description
s3-read-write-access	Customer managed	None	Allows read and write access to S3 buckets
S3-ReadWritePolicy	Customer managed	Permissions policy (1)	Allows IAM users to read and write objec…
S3ObjectReadOnlyAcc…	Customer managed	Permissions policy (1)	-

3. *Review Terraform State:*

Inspect the Terraform state file (`terraform.tfstate` or `terraform.tfstate.d`) to see the current status of resources. Search for the IAM policy resource to confirm its presence and verify that it has the correct attributes.

```
{
  "version": 4,
  "terraform_version": "1.6.0",
  "serial": 53,
  "lineage": "415c56e8-89fd-a5ba-14ae-0d47cfe8db52",
  "outputs": {},
  "resources": [
    {
      "mode": "managed",
      "type": "aws_iam_policy",
      "name": "s3_read_write_policy",
      "provider": "provider[\"registry.terraform.io/hashicorp/aws\"]",
      "instances": [
        {
          "schema_version": 0,
          "attributes": {
            "arn": "arn:aws:iam::891377036070:policy/s3-read-write-access",
            "description": "Allows read and write access to S3 buckets",
            "id": "arn:aws:iam::891377036070:policy/s3-read-write-access",
            "name": "s3-read-write-access",
            "name_prefix": "",
            "path": "/",
            "policy": "{\"Statement\":[{\"Action\":[\"S3:GetObject\",\"S3:PutOb
            "policy_id": "ANPA47CRWFMTDHD4OVKLM",
            "tags": null,
            "tags_all": {}
          },
          "sensitive_attributes": [],
          "private": "bnVsbA=="
        }
        ]
      },
```

4. Test Permissions:

To ensure that the IAM policy functions as expected, perform tests using IAM users or roles assigned the policy. Attempt actions allowed by the policy and confirmed that they are permitted, while actions not allowed are denied.

By following these steps, you can validate that the IAM policy has been successfully created or updated according to your Terraform configuration, providing confidence in your infrastructure's security and access control settings.

2. Bucket Policies Configuration with Terraform:

Configuring bucket policies with Terraform provides a robust and scalable method to manage access controls for your S3 buckets. With Terraform, you can define policies as code, ensuring consistency and auditability across your infrastructure. This approach simplifies policy management, enabling you to efficiently enforce security requirements and access restrictions for your AWS resources. Let's explore how to leverage Terraform for seamless bucket policy configuration.

When managing bucket policies for your S3 buckets using Terraform, follow these steps:

1. Define Terraform Configuration:

Ensure you have a Terraform configuration file (usually named `main.tf` or similar) in your project directory.

1.1: Declare an S3 Bucket

Define an AWS S3 bucket using the `aws_s3_bucket` resource in your Terraform configuration. Specify the desired bucket name within the `bucket` attribute.

```
resource "aws_s3_bucket" "my_terraform_bucket" {
  bucket = "my-terraform-bucket-278"
}
```

1.2: Define Bucket Policy

Utilize the `aws_s3_bucket_policy` resource to define the bucket policy for the previously created S3 bucket. Set the `bucket` attribute to reference the ID of the S3 bucket, and set the `policy` attribute to reference the JSON policy document.

```
resource "aws_s3_bucket_policy" "s3_getobject_policy" {
  bucket = aws_s3_bucket.my_terraform_bucket.id
  policy = data.aws_iam_policy_document.s3_getobject_policy.json
}
```

1.3: Create IAM Policy Document

Define the IAM policy document using the `aws_iam_policy_document` data source. In this example, we're allowing the `s3:GetObject` action for specific IAM principals (users, groups, or roles) identified by their AWS account IDs.

```
data "aws_iam_policy_document" "s3_getobject_policy" {
  statement {
    principals {
      type        = "AWS"
      identifiers = ["891377036070"]
    }

    actions = [
      "s3:GetObject",
    ]

    resources = [
      aws_s3_bucket.my_terraform_bucket.arn,
      "${aws_s3_bucket.my_terraform_bucket.arn}/*",
    ]
  }
}
```

These steps define a bucket policy named `s3-getobject` for the S3 bucket `my-terraform-bucket-278`, allowing all users to perform the `s3:GetObject` action on objects within the bucket. Adjust the policy document as needed to match your specific requirements.

1.4: Review and Apply Changes

Before applying the Terraform configuration, ensure that the defined bucket name and policy actions align with your requirements. After verifying the changes, apply the configuration to create or update the S3 bucket and associated bucket policy accordingly.

1. Initialize Terraform:

Run `terraform init` in your project directory to initialize Terraform and download any necessary plugins.

2. Apply Changes:

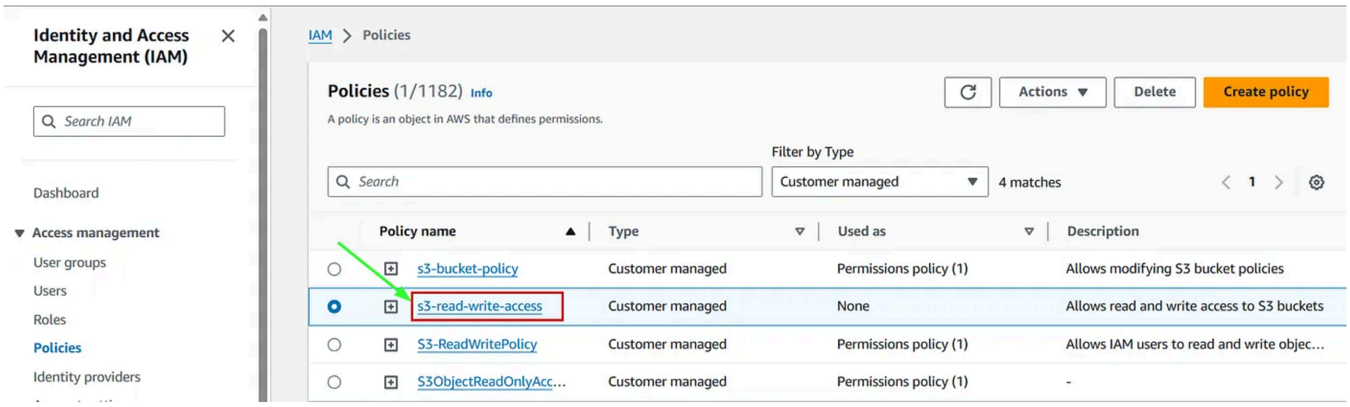
Run `terraform apply` and review the proposed changes. If everything looks correct, confirm by typing `yes` when prompted.

2. Verify in AWS Console:

After applying the Terraform configuration, verify the changes in the AWS Management Console:

2.1 IAM Policy:

Navigate to the IAM service in the AWS Management Console. Locate the IAM policy with the specified name, such as ***"s3-read-write-access"***. Ensure that it exists and matches the configuration provided in Terraform, including the description and policy document.



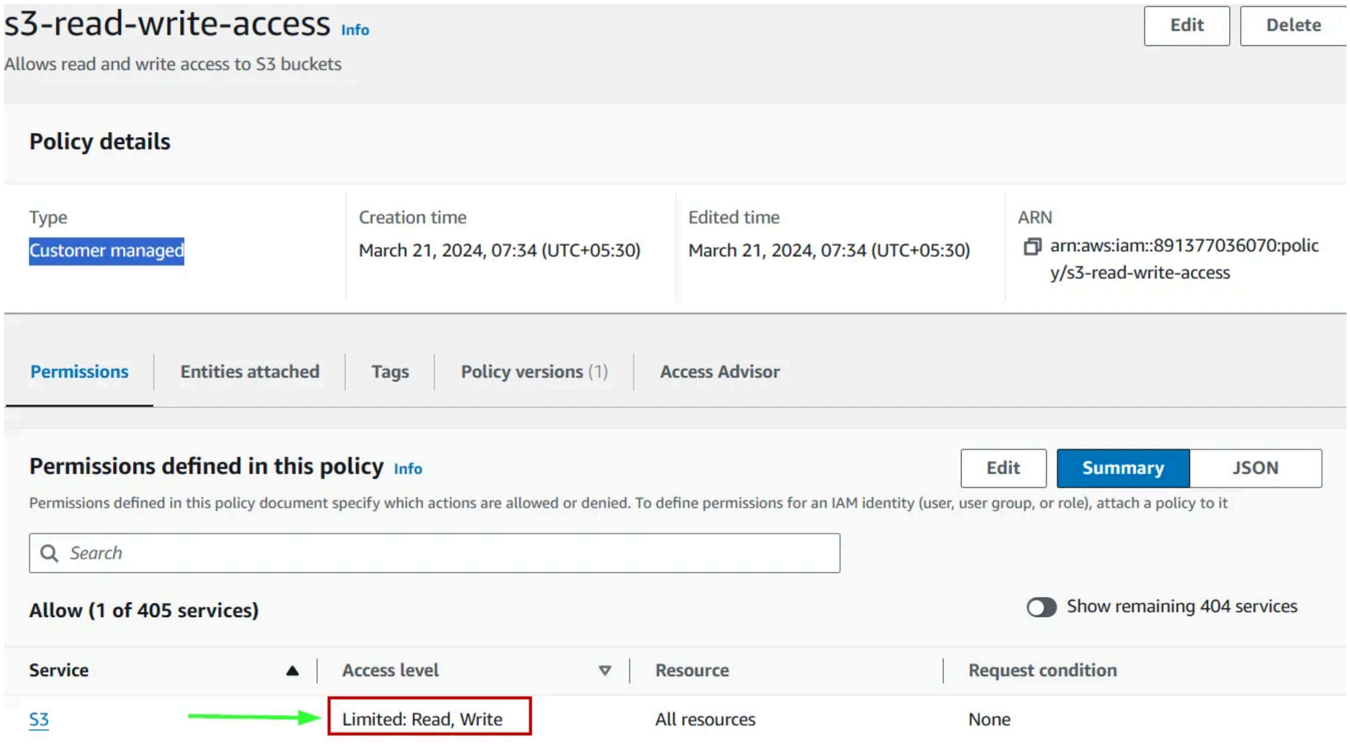
Policies (1/1182) Info

A policy is an object in AWS that defines permissions.

Filter by Type: Customer managed

4 matches

	Policy name	Type	Used as	Description
○	s3-bucket-policy	Customer managed	Permissions policy (1)	Allows modifying S3 bucket policies
●	s3-read-write-access	Customer managed	None	Allows read and write access to S3 buckets
○	S3-ReadWritePolicy	Customer managed	Permissions policy (1)	Allows IAM users to read and write objec...
○	S3ObjectReadOnlyAcc...	Customer managed	Permissions policy (1)	-


s3-read-write-access Info

Allows read and write access to S3 buckets

Edit Delete

Policy details

Type	Creation time	Edited time	ARN
Customer managed	March 21, 2024, 07:34 (UTC+05:30)	March 21, 2024, 07:34 (UTC+05:30)	arn:aws:iam::891377036070:policy/s3-read-write-access

Permissions | Entities attached | Tags | Policy versions (1) | Access Advisor

Permissions defined in this policy Info

Edit Summary JSON

Permissions defined in this policy document specify which actions are allowed or denied. To define permissions for an IAM identity (user, user group, or role), attach a policy to it

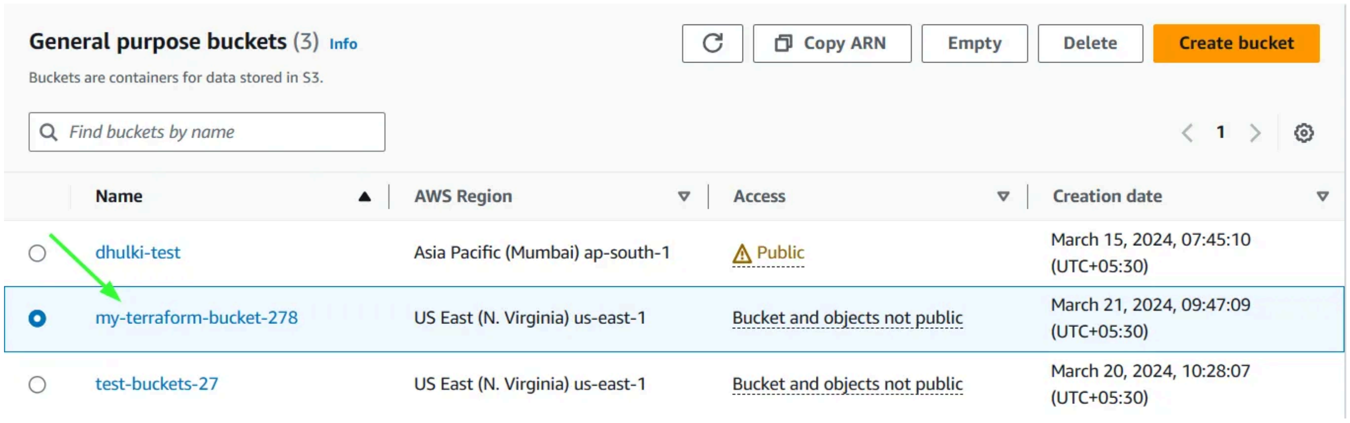
Allow (1 of 405 services)

Show remaining 404 services

Service	Access level	Resource	Request condition
S3	Limited: Read, Write	All resources	None

2.2 Bucket Policy:

Access the Amazon S3 service in the AWS Management Console. Open the bucket for which you defined the bucket policy. Navigate to the *"Permissions"* tab and select *"Bucket Policy"*. Confirm that the bucket policy is applied correctly, allowing the specified actions on the designated resources.

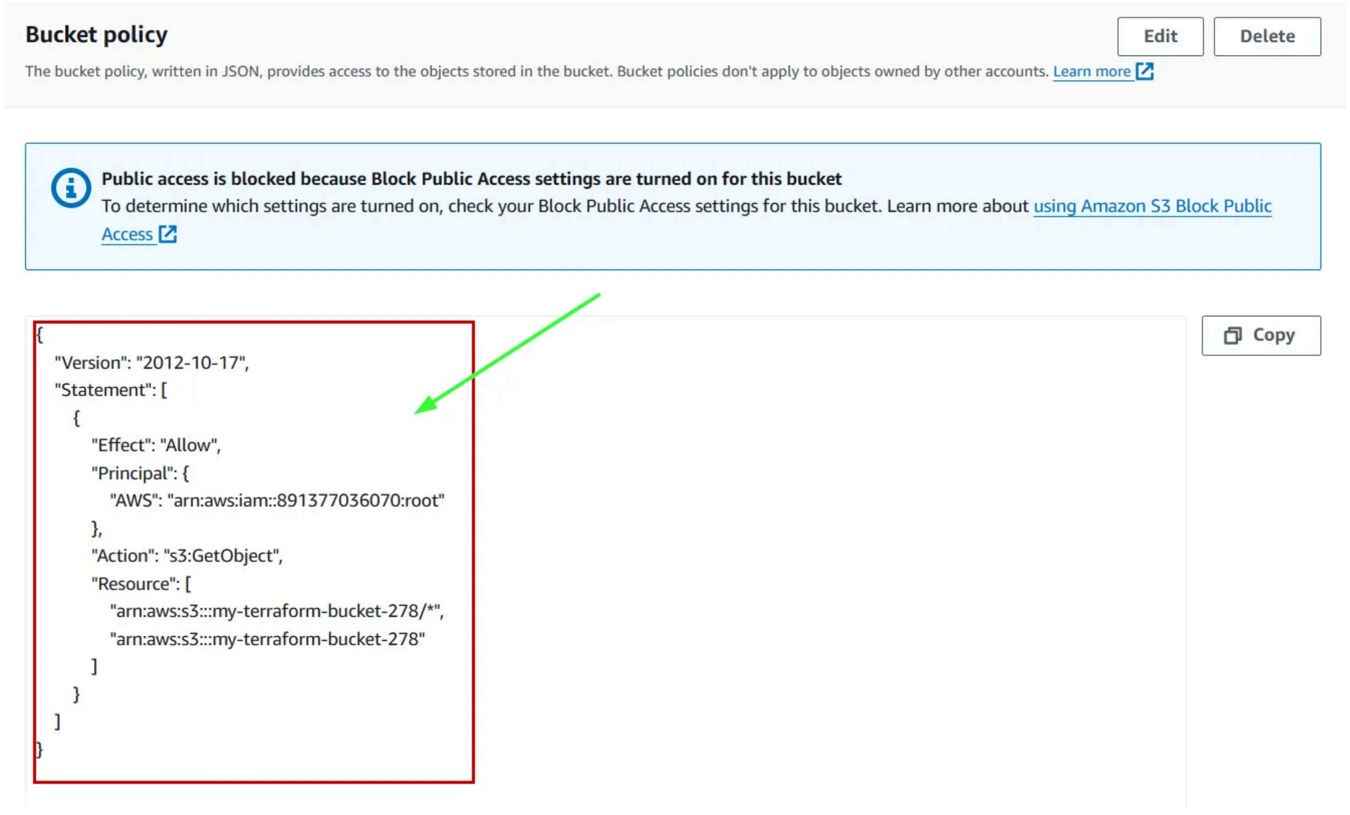

General purpose buckets (3) Info

Buckets are containers for data stored in S3.

Copy ARN Empty Delete Create bucket

Find buckets by name

	Name	AWS Region	Access	Creation date
○	dhulki-test	Asia Pacific (Mumbai) ap-south-1	⚠ Public	March 15, 2024, 07:45:10 (UTC+05:30)
●	my-terraform-bucket-278	US East (N. Virginia) us-east-1	Bucket and objects not public	March 21, 2024, 09:47:09 (UTC+05:30)
○	test-buckets-27	US East (N. Virginia) us-east-1	Bucket and objects not public	March 20, 2024, 10:28:07 (UTC+05:30)



The bucket policy, written in JSON, provides access to the objects stored in the bucket. Bucket policies don't apply to objects owned by other accounts. Learn more

Public access is blocked because Block Public Access settings are turned on for this bucket
To determine which settings are turned on, check your Block Public Access settings for this bucket. Learn more about using Amazon S3 Block Public Access

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::891377036070:root"
      },
      "Action": "s3:GetObject",
      "Resource": [
        "arn:aws:s3:::my-terraform-bucket-278/*",
        "arn:aws:s3:::my-terraform-bucket-278"
      ]
    }
  ]
}
```

By following these steps, you can ensure that the IAM policy and bucket policy defined in Terraform are accurately reflected in the AWS Management Console.

Best Practices for Securing S3 Buckets using Access Control:

1. *Principle of Least Privilege:* Grant only the minimum permissions required for users and applications to accomplish their tasks.
2. *Regular Review and Audit:* Periodically review and audit access control policies to ensure they align with security requirements.
3. *Use IAM Roles for Applications:* Avoid using long-term access keys and instead utilize IAM roles to provide temporary credentials with restricted permissions.
4. *Enable MFA Delete:* Add an extra layer of security by enabling Multi-Factor Authentication (MFA) Delete on S3 buckets.
5. *Monitor Access and Logging:* Utilize S3 server access logging and AWS CloudTrail to track access requests and monitor for unauthorized activity.

Conclusion:

Effective management of access control policies is crucial for securing Amazon S3 buckets and safeguarding sensitive data. Understanding IAM policies, bucket policies, and adhering to best practices for access control configuration are essential steps in ensuring the security of your S3 resources. Whether through the AWS Management Console, AWS CLI, or Terraform, implementing robust access controls helps maintain the confidentiality, integrity, and availability of your data stored in Amazon S3.

Stay tuned for the next installment of this series, where we'll explore another critical aspect of Amazon S3: Versioning. Discover how versioning enhances data protection and recovery strategies within S3 buckets.

> Connect with me on LinkedIn for more insights and discussions on cloud security and AWS best practices.

Aws S3 · S3 Bucket Policies · S3 Bucket Properties · Cloud Storage Security · S3 Bucket Access

M

Follow

Written by Mohasina Clt

41 followers · 40 following

🌟 🎓 📊 ➡️ 💻 Freelance cloud consultant, educator @ Insight for Innovation, & Medium writer. Let's empower, inspire, & innovate! ✨ 🚀 📝