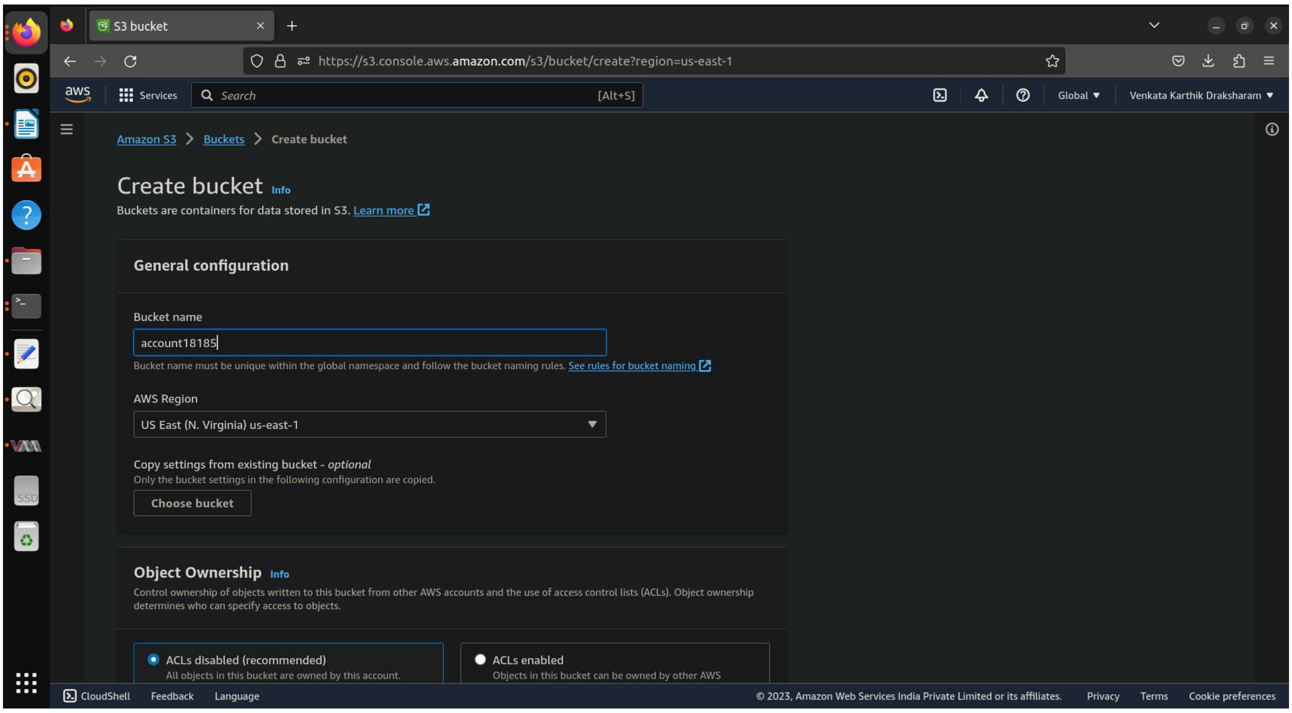
Before starting to showcase my assignment, I would like to extend my sincere gratitude to OpeninApp for affording me the invaluable opportunity to demonstrate my skills through the recent assignment I undertook as part of the internship application process. The experience has been immensely enriching, and it has allowed me to delve deeper into the nuances of AWS resources while also gaining a deeper insight into the dynamic and innovative culture that OpeninApp fosters. This assignment not only challenged me to push my boundaries but also highlighted the alignment of my aspirations with the values and objectives of your esteemed organization. I genuinely appreciate the chance to participate in this process, and I am eager to continue pursuing opportunities to contribute my skills and dedication towards OpeninApp's mission. Once again, thank you for this exceptional opportunity, and I look forward to the possibility of further collaboration in the future.

Assignment

Step 1: Create S3 buckets in account1 and account2, there are two ways of doing this, one is programmatic approach using AWS CLI while another way is using AWS management console, For convenience, I am going to perform using AWS management console.



https://s3.console.aws.amazon.com/s3/bucket/create?region=us-east-1

Create bucket

Buckets are containers for data stored in S3. Learn more

General configuration

Bucket name: account18185

Bucket name must be unique within the global namespace and follow the bucket naming rules. See rules for bucket naming

AWS Region: US East (N. Virginia) us-east-1

Copy settings from existing bucket - *optional*

Only the bucket settings in the following configuration are copied.

Object Ownership

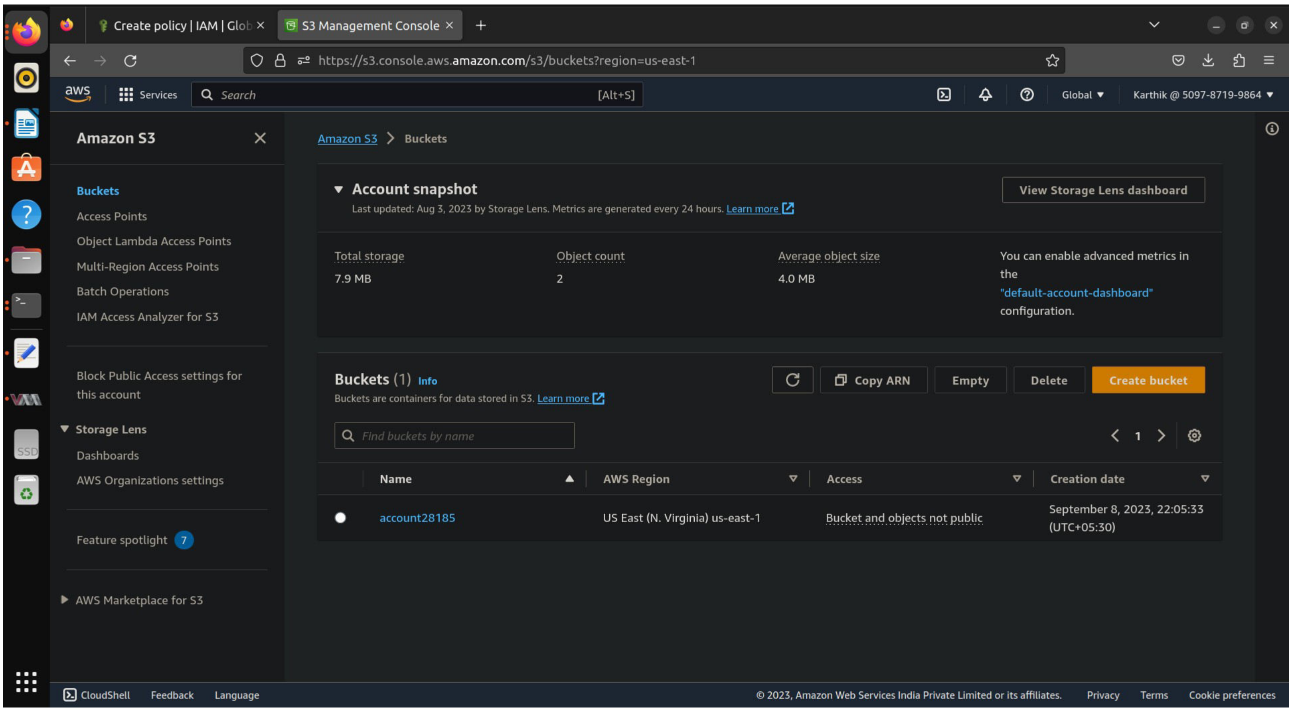
Control ownership of objects written to this bucket from other AWS accounts and the use of access control lists (ACLs). Object ownership determines who can specify access to objects.

● ACLs disabled (recommended)
All objects in this bucket are owned by this account.

○ ACLs enabled
Objects in this bucket can be owned by other AWS

© 2023, Amazon Web Services India Private Limited or its affiliates.

Similarly, in Account-2 as well,


Create policy | IAM | Glob
S3 Management Console
https://s3.console.aws.amazon.com/s3/buckets?region=us-east-1
Services
Search
[Alt+S]
Global
Karthik @ 5097-8719-9864
Amazon S3
Buckets
Access Points
Object Lambda Access Points
Multi-Region Access Points
Batch Operations
IAM Access Analyzer for S3
Block Public Access settings for this account
Storage Lens
Dashboards
AWS Organizations settings
Feature spotlight 7
AWS Marketplace for S3
Amazon S3 > Buckets
View Storage Lens dashboard
Copy ARN
Empty
Delete
Create bucket
Find buckets by name
CloudShell
Feedback
Language
© 2023, Amazon Web Services India Private Limited or its affiliates.
Privacy
Terms
Cookie preferences

Account snapshot

Last updated: Aug 3, 2023 by Storage Lens. Metrics are generated every 24 hours. Learn more

Total storage	Object count	Average object size	You can enable advanced metrics in the "default-account-dashboard" configuration.
7.9 MB	2	4.0 MB	

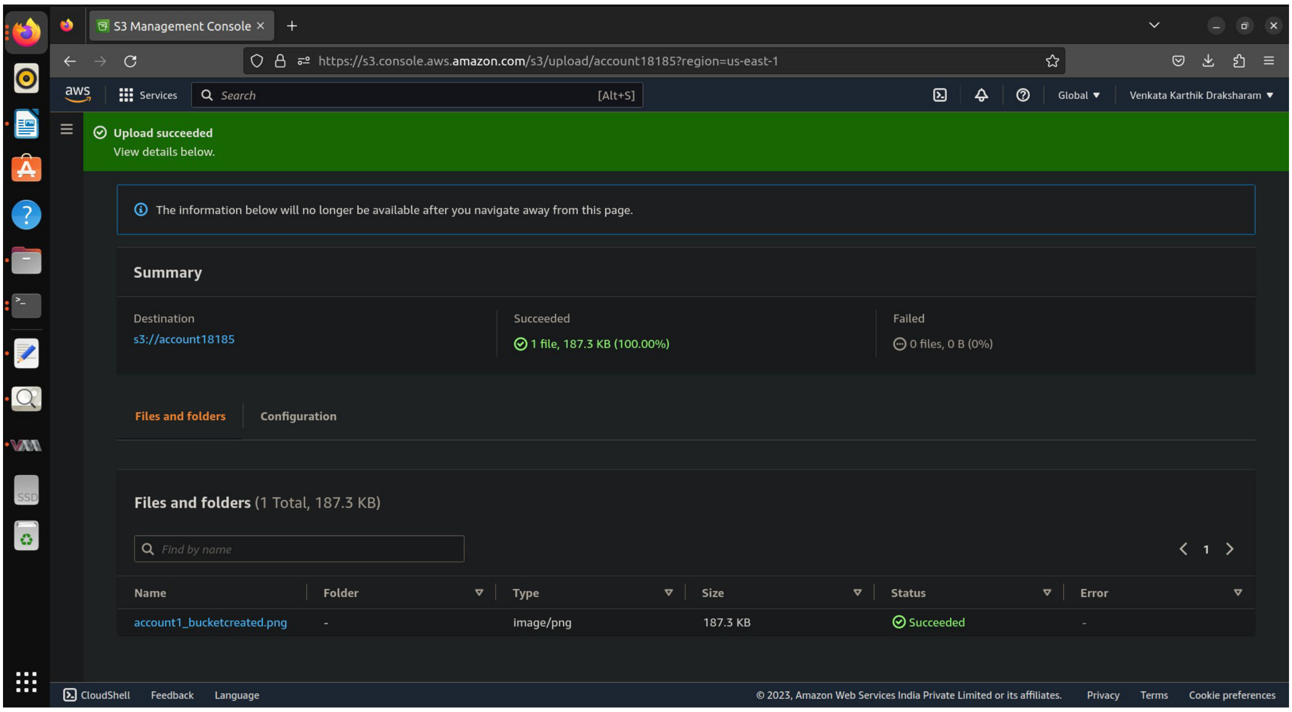
Buckets (1) Info

Buckets are containers for data stored in S3. Learn more

	Name	AWS Region	Access	Creation date
○	account28185	US East (N. Virginia) us-east-1	Bucket and objects not public	September 8, 2023, 22:05:33 (UTC+05:30)

Step 2: Add desired objects in Account1 In my case it is a screenshot of account1_bucket_creation.

Bucket name: account28185


S3 Management Console
https://s3.console.aws.amazon.com/s3/upload/account18185?region=us-east-1
Services
Search
[Alt+S]
Global
Venkata Karthik Draksharam
Files and folders
Configuration
Find by name
CloudShell
Feedback
Language
© 2023, Amazon Web Services India Private Limited or its affiliates.
Privacy
Terms
Cookie preferences

Upload succeeded
View details below.

The information below will no longer be available after you navigate away from this page.

Summary

Destination	Succeeded	Failed
s3://account18185	1 file, 187.3 KB (100.00%)	0 files, 0 B (0%)

Files and folders (1 Total, 187.3 KB)

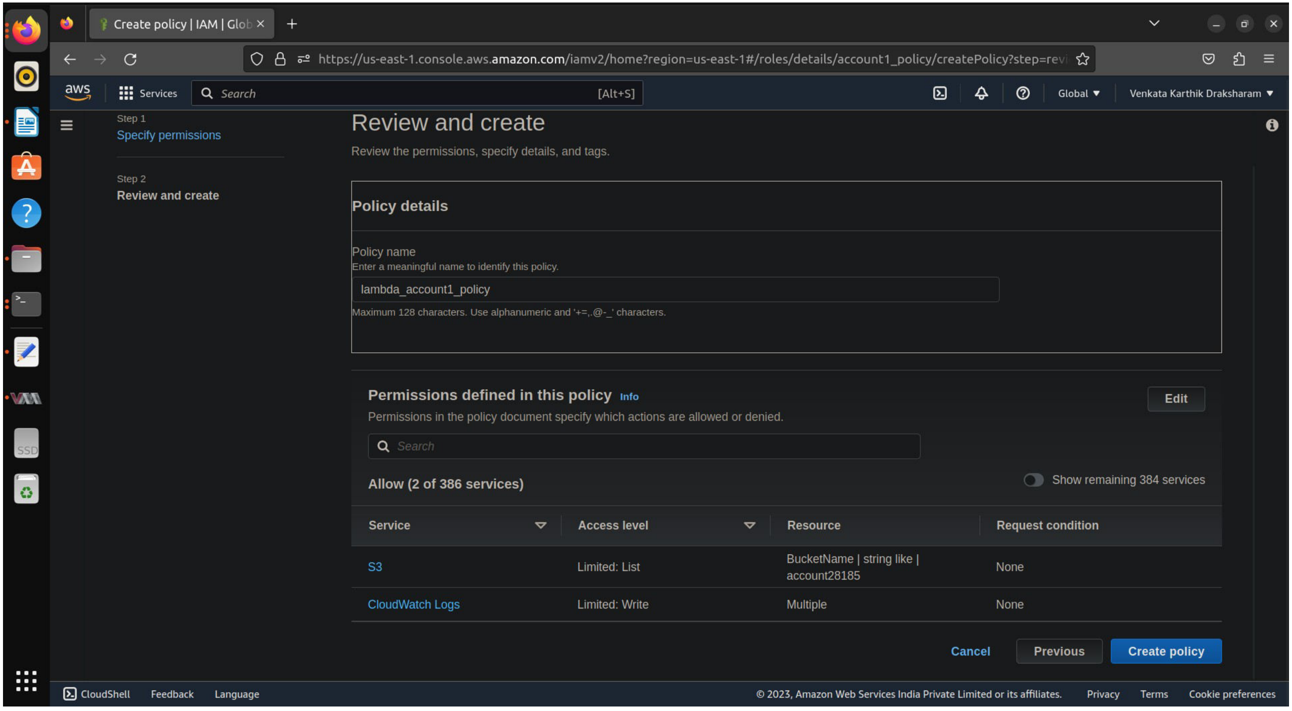
Name	Folder	Type	Size	Status	Error
account1_bucketcreated.png	-	image/png	187.3 KB	Succeeded	-

Step 3: Use created S3 bucket from Account-2 and identify bucket name, account-2 ID.

Step 4: Create a IAM role in Account 1, with an inline policy and AWSLambdaBasicExecutionRole.

Bucket name: account128185
Account 1 ID: XXXXXXXXXXXX

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::accoun18185/*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:PutObjectAcl"
            ],
            "Resource": "arn:aws:s3:::account28185/*"
        }
    ]
}
```


Create policy | IAM | Glob
https://us-east-1.console.aws.amazon.com/iamv2/home?region=us-east-1#/roles/details/account1_policy/createPolicy?step=revi
aws
Services
Search
[Alt+S]
Global
Venkata Karthik Draksharam
Step 1
Specify permissions
Step 2
Review and create
Info
Edit
Search
Cancel
Previous
Create policy
CloudShell
Feedback
Language
© 2023, Amazon Web Services India Private Limited or its affiliates.
Privacy
Terms
Cookie preferences

Review and create

Review the permissions, specify details, and tags.

Policy details

Policy name
Enter a meaningful name to identify this policy.
lambda_account1_policy
Maximum 128 characters. Use alphanumeric and '+=,.@-_' characters.

Permissions defined in this policy

Permissions in the policy document specify which actions are allowed or denied.

Allow (2 of 386 services)

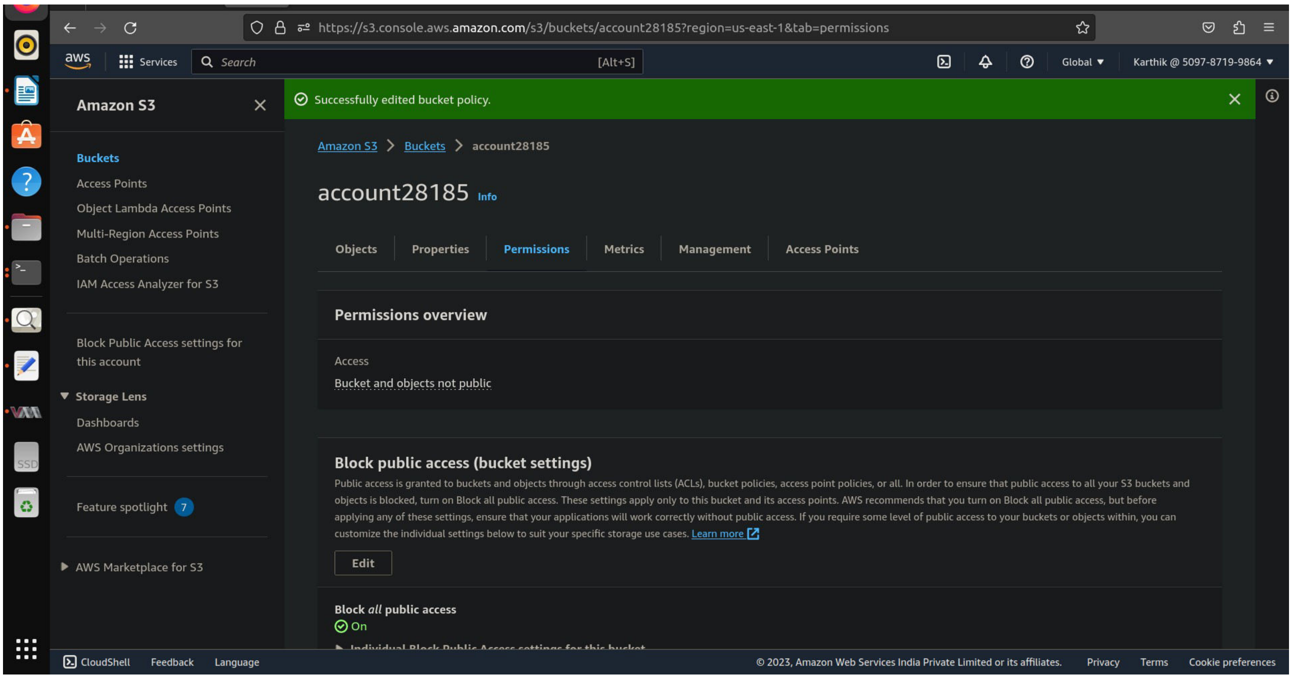
Show remaining 384 services

Service	Access level	Resource	Request condition
S3	Limited: List	BucketName | string like | account28185	None
CloudWatch Logs	Limited: Write	Multiple	None

Step 5: Create a Bucket Policy for S3 Bucket in Account 2.

Bucket name: account18185
Account 1 ID: XXXXXXXXXXXX

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::XXXXXXXXXXXX:role/role_a"
            },
            "Action": [
                "s3:PutObject",
                "s3:PutObjectAcl"
            ],
            "Resource": "arn:aws:s3:::account28185/*"
        }
    ]
```



https://s3.console.aws.amazon.com/s3/buckets/account28185?region=us-east-1&tab=permissions

Successfully edited bucket policy.

Amazon S3 > Buckets > account28185

account28185 Info

Permissions overview

Access
Bucket and objects not public

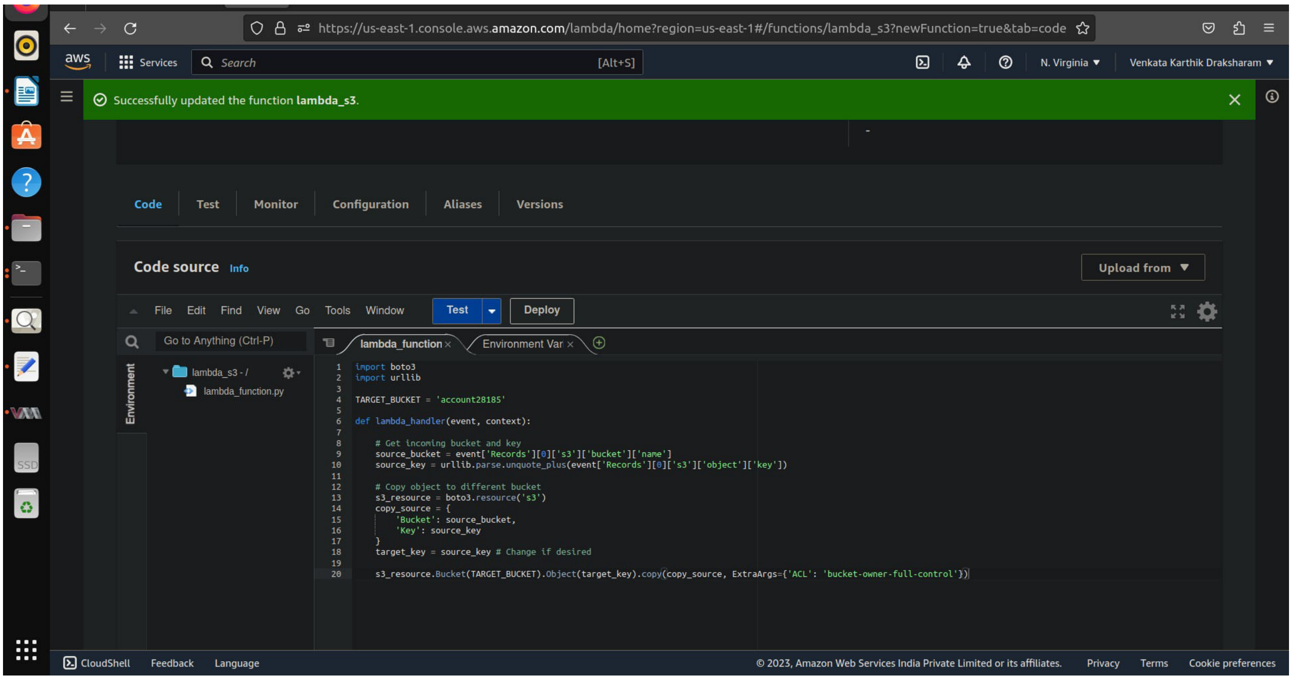
Block public access (bucket settings)

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
On

Step 6: Create a Lambda Function which will trigger every time we upload a object in accoun18185.

```
import boto3
import urllib

TARGET_BUCKET = 'accoun28185'

def lambda_handler(event, context):

    # Get incoming bucket and key
    source_bucket = event['Records'][0]['s3']['bucket']['name']
    source_key = urllib.parse.unquote_plus(event['Records'][0]['s3']['object']['key'])

    # Copy object to different bucket
    s3_resource = boto3.resource('s3')
    copy_source = {
        'Bucket': source_bucket,
        'Key': source_key
    }
    target_key = source_key

    s3_resource.Bucket(TARGET_BUCKET).Object(target_key).copy(copy_source, ExtraArgs={'ACL': 'bucket-owner-full-control'})
```

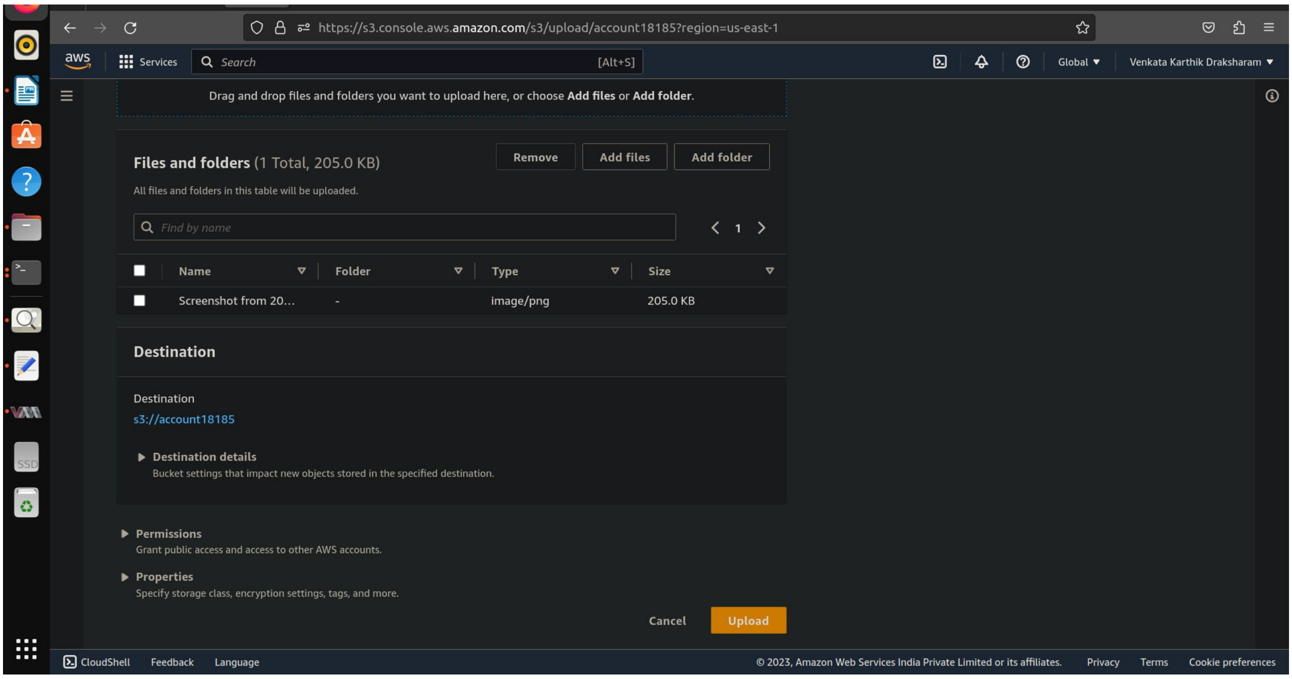


Successfully updated the function **lambda_s3**.

```
import boto3
import urllib

TARGET_BUCKET = 'account28185'

def lambda_handler(event, context):

    # Get incoming bucket and key
    source_bucket = event['Records'][0]['s3']['bucket']['name']
    source_key = urllib.parse.unquote_plus(event['Records'][0]['s3']['object']['key'])

    # Copy object to different bucket
    s3_resource = boto3.resource('s3')
    copy_source = {
        'Bucket': source_bucket,
        'Key': source_key
    }
    target_key = source_key # Change if desired

    s3_resource.Bucket(TARGET_BUCKET).Object(target_key).copy(copy_source, ExtraArgs={'ACL': 'bucket-owner-full-control'})
```

With lambda deployed, now we will be able to fetch data to S3 bucket of account 2.

Testing: I am uploading a different image to see if this works.



https://s3.console.aws.amazon.com/s3/upload/account18185?region=us-east-1

Drag and drop files and folders you want to upload here, or choose **Add files** or **Add folder**.

Files and folders (1 Total, 205.0 KB)

All files and folders in this table will be uploaded.

☐	Name	Folder	Type	Size
☐	Screenshot from 20...	-	image/png	205.0 KB

Destination

Destination

s3://account18185

▶ **Destination details**

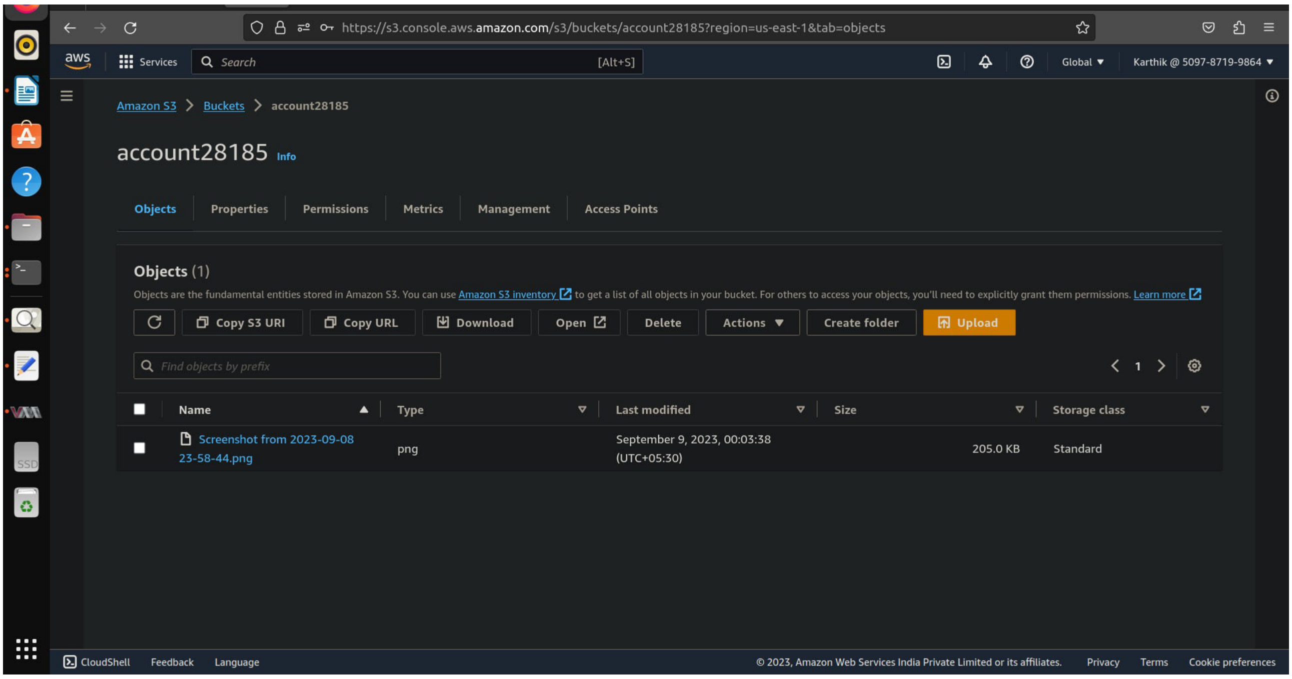
Bucket settings that impact new objects stored in the specified destination.

▶ **Permissions**

Grant public access and access to other AWS accounts.

▶ **Properties**

Specify storage class, encryption settings, tags, and more.



https://s3.console.aws.amazon.com/s3/buckets/account28185?region=us-east-1&tab=objects

Amazon S3 > Buckets > account28185

account28185

Objects | Properties | Permissions | Metrics | Management | Access Points

Objects (1)

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 2023-09-08 23-58-44.png	png	September 9, 2023, 00:03:38 (UTC+05:30)	205.0 KB	Standard

The above image successfully shows that the uploaded image has been successfully transferred an object from AWS lambda of different account.

Clean up:

1) Delete objects and bucket from account 1
2) Delete objects and bucket from account 2
3) Delete AWS lambda function which is created
4) If STS token is created, delete the STS token
5) Delete IAM role which is created on account 1.

Solutions and Add-on:

1) Using STS token, it creates a temporary access and a good security practice.
2) Replication of objects will be a better approach.

AWS reference:

https://docs.aws.amazon.com/prescriptive-guidance/latest/patterns/copy-data-from-an-s3-bucket-to-another-account-and-region-by-using-the-aws-cli.html