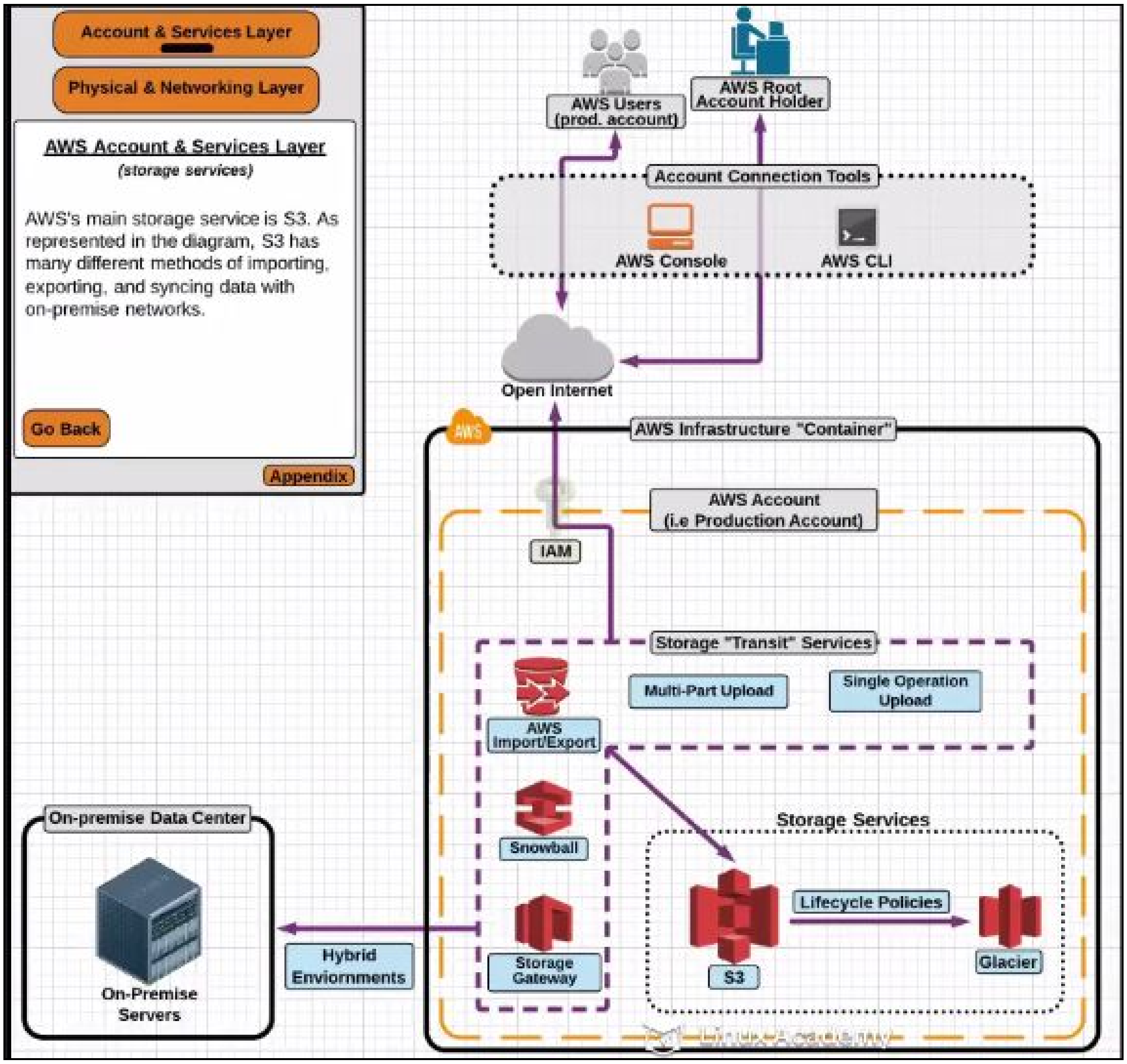
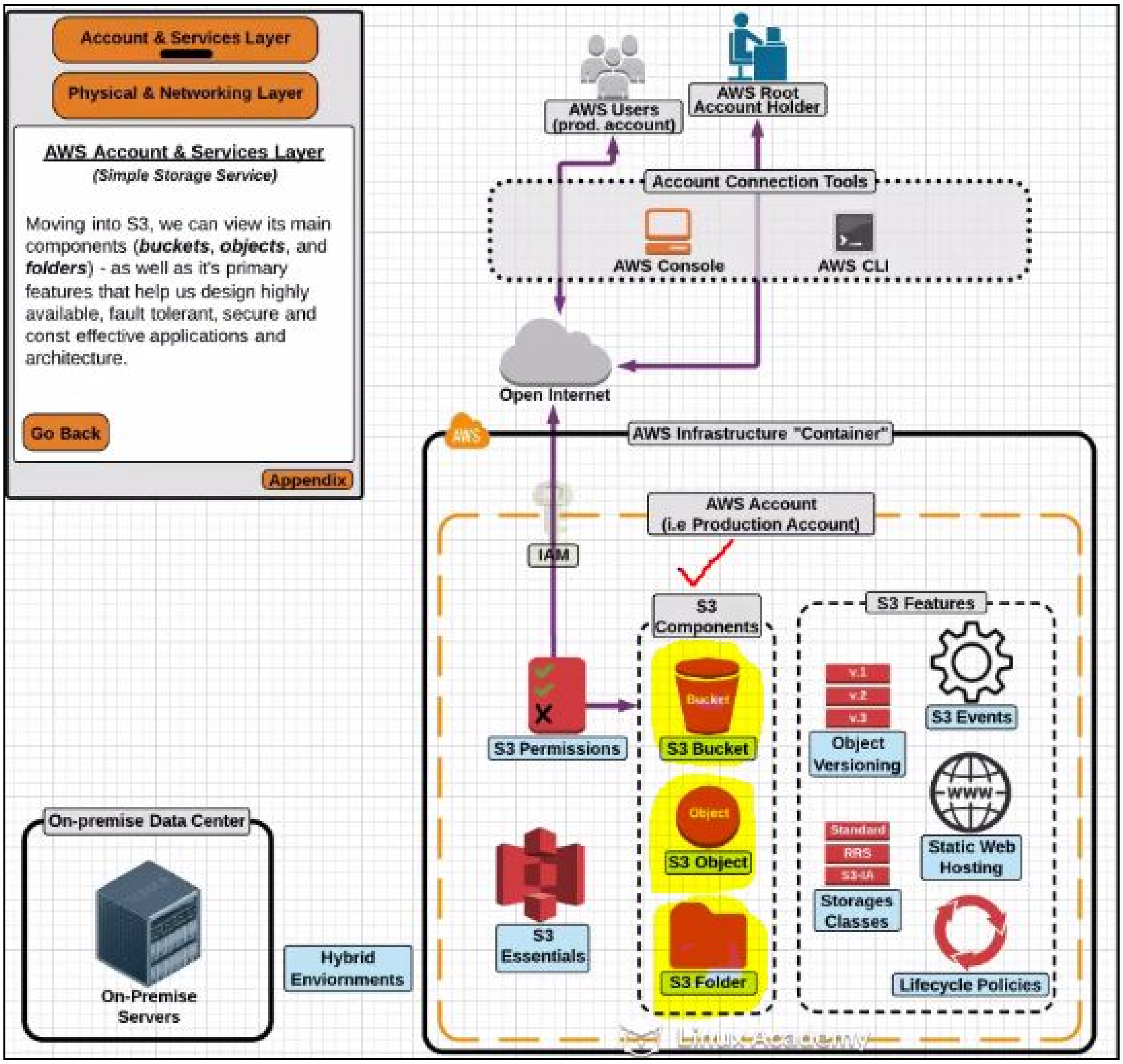
S3 (Simple Storage Service)



AWS Account & Services Layer

(storage services)

AWS's main storage service is S3. As represented in the diagram, S3 has many different methods of importing, exporting, and syncing data with on-premise networks.



Account & Services Layer

Physical & Networking Layer

AWS Account & Services Layer

(Simple Storage Service)

Moving into S3, we can view its main components (***buckets***, ***objects***, and ***folders***) - as well as it's primary features that help us design highly available, fault tolerant, secure and const effective applications and architecture.

Go Back

Appendix

S3 Essentials

S3 basics (Simple Storage Service)

- Amazon S3 is storage for the internet.
- Amazon S3 is a object based storage not a block-based storage. It means you cannot install apps in this storage, only you can keep files.
- Amazon S3 provides a simple web service interface that you can use to store and retrieve any amount of data, at any time, from anywhere on the internet.

Eg: Using this web service, developers can easily build applications that make use of internet storage. Since S3 is highly scalable and you can pay for what you use, developers can start with small and grow their application as they wish, with no compromise on performance or reliability

- AWS S3 is the main storage service which serves many purposes when designing ***highly available, fault tolerant, highly scalable, reliable, low-latency*** data storage infrastructure at ***very low-cost***.
- S3 is also known as key-value data store (key,value, versionID, MetaData , ACL and sub-resources)

Note: Example of S3 URL : https://s3-ap-south1.amazonaws.com/mybucket1
If upload is successful, then status of returned code is HTTP200

Use Cases:

- Bulk (basically unlimited) static object storage
- Hosting static files & websites
- Object versioning
- Origin for CloudFront CDN
- And many more

Important S3 Facts

- Objects stay within an AWS region and are synced across all AZ's for extremely High Availability and Durability
- You should always create an S3 bucket in a region that makes sense to its purpose:
    - service content to customer
    - sharing data with EC2

Benefits of S3

- 99.99% availability
- 99.999999999% durability (called as eleven 9's durability)

S3 Read Consistency Rule

- ALL regions now support read-after-write consistency for PUTS of new objects into S3
  i.e it means objects are immediately available after putting it in S3 bucket
- All regions use eventual consistency for PUTS overwriting existing objects and DELETES of object
  i.e If you delete or overwrite then there may be some delay

S3 Bucket

S3 Buckets

- Buckets are the main storage container of S3, and contain a grouping of information and have sub namespaces that are similar to folders (but yet are called folders)
- Tags can be used to organize buckets (i.e tags based on application the buckets belongs to)
- **Each bucket must have a unique name across ALL of AWS. Also it should be in small case letters.**

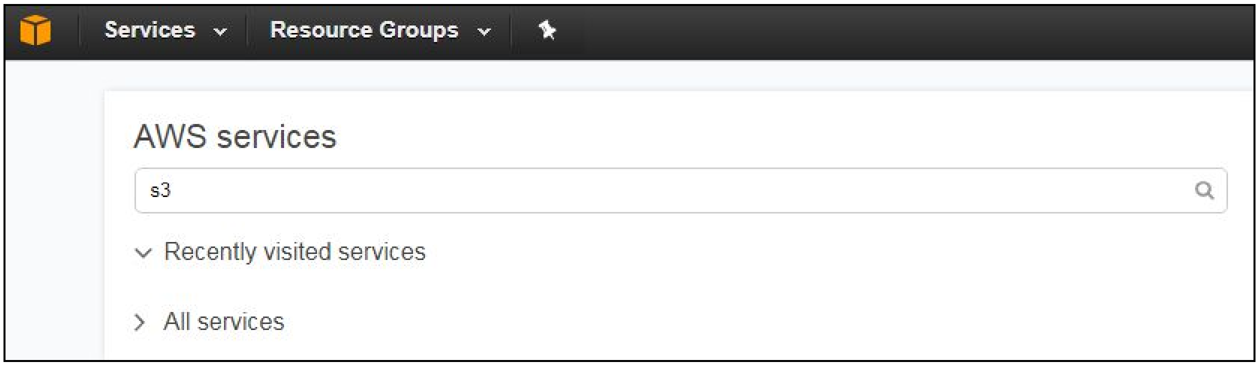
S3 bucket limitations

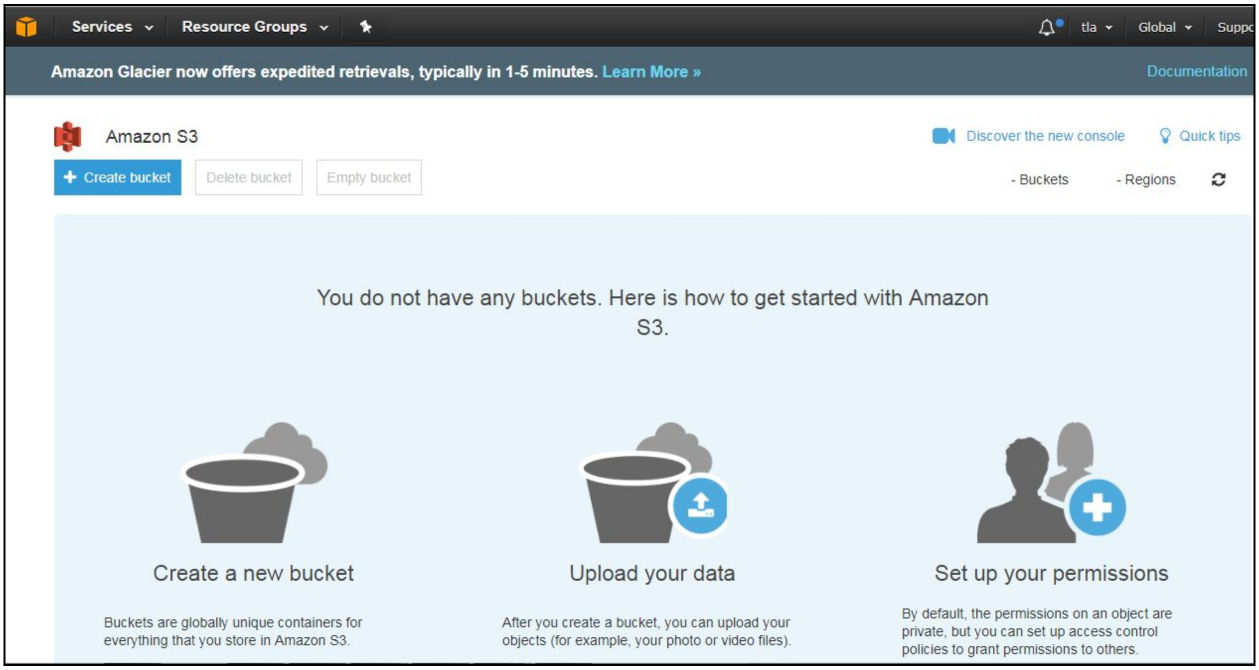
- Only 100 buckets can be created
- Bucket ownership cannot be changed later also

NOTE: S3 as a service is actually global, you can view all the buckets that are in different regions at the same place (global)

LAB 1:Creating a Bucket

Login to management console and search for S3 service





Amazon Glacier now offers expedited retrievals, typically in 1-5 minutes. Learn More »

You do not have any buckets. Here is how to get started with Amazon S3.

Create a new bucket

Buckets are globally unique containers for everything that you store in Amazon S3.

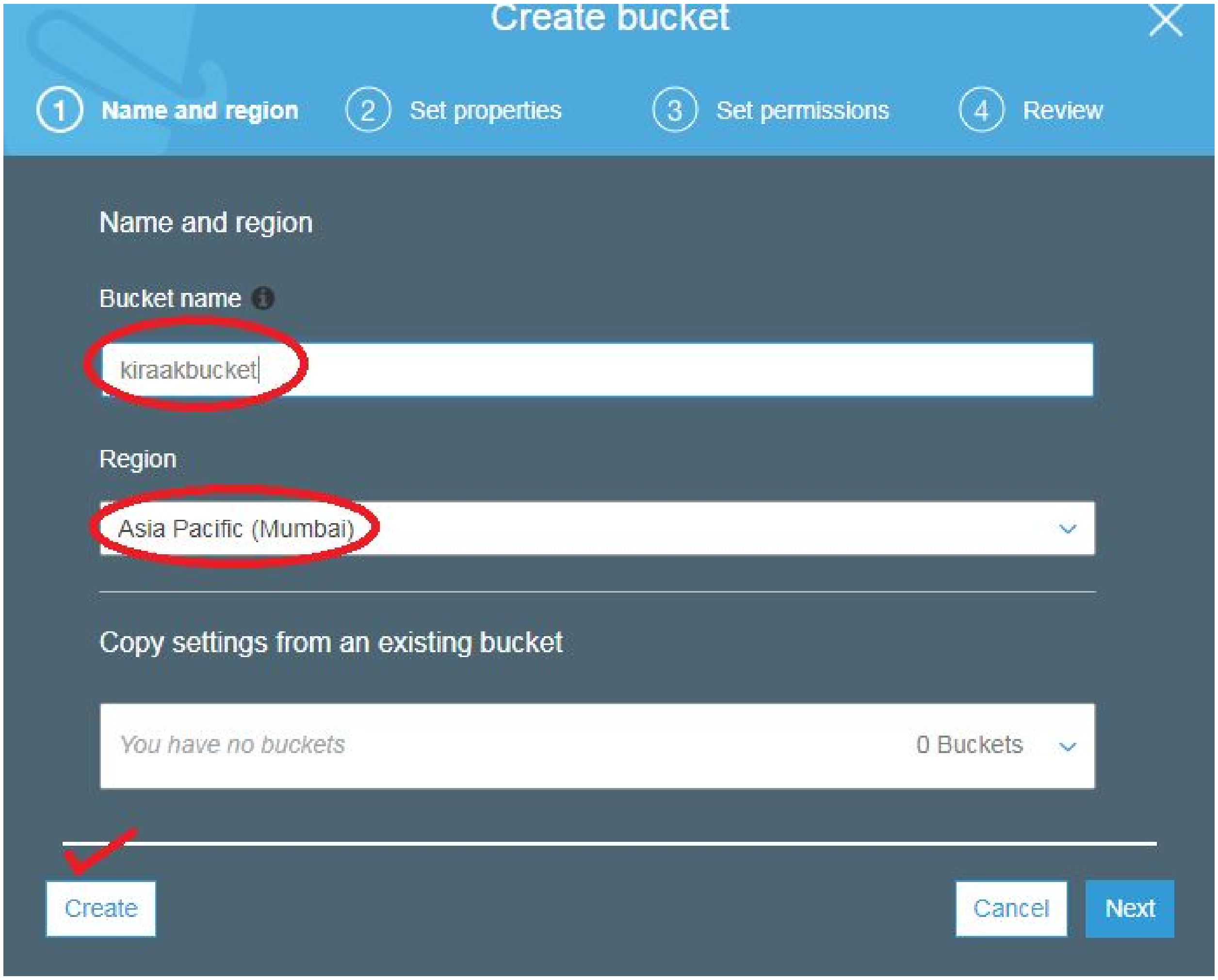
Upload your data

After you create a bucket, you can upload your objects (for example, your photo or video files).

Set up your permissions

By default, the permissions on an object are private, but you can set up access control policies to grant permissions to others.

Click on **Create bucket**



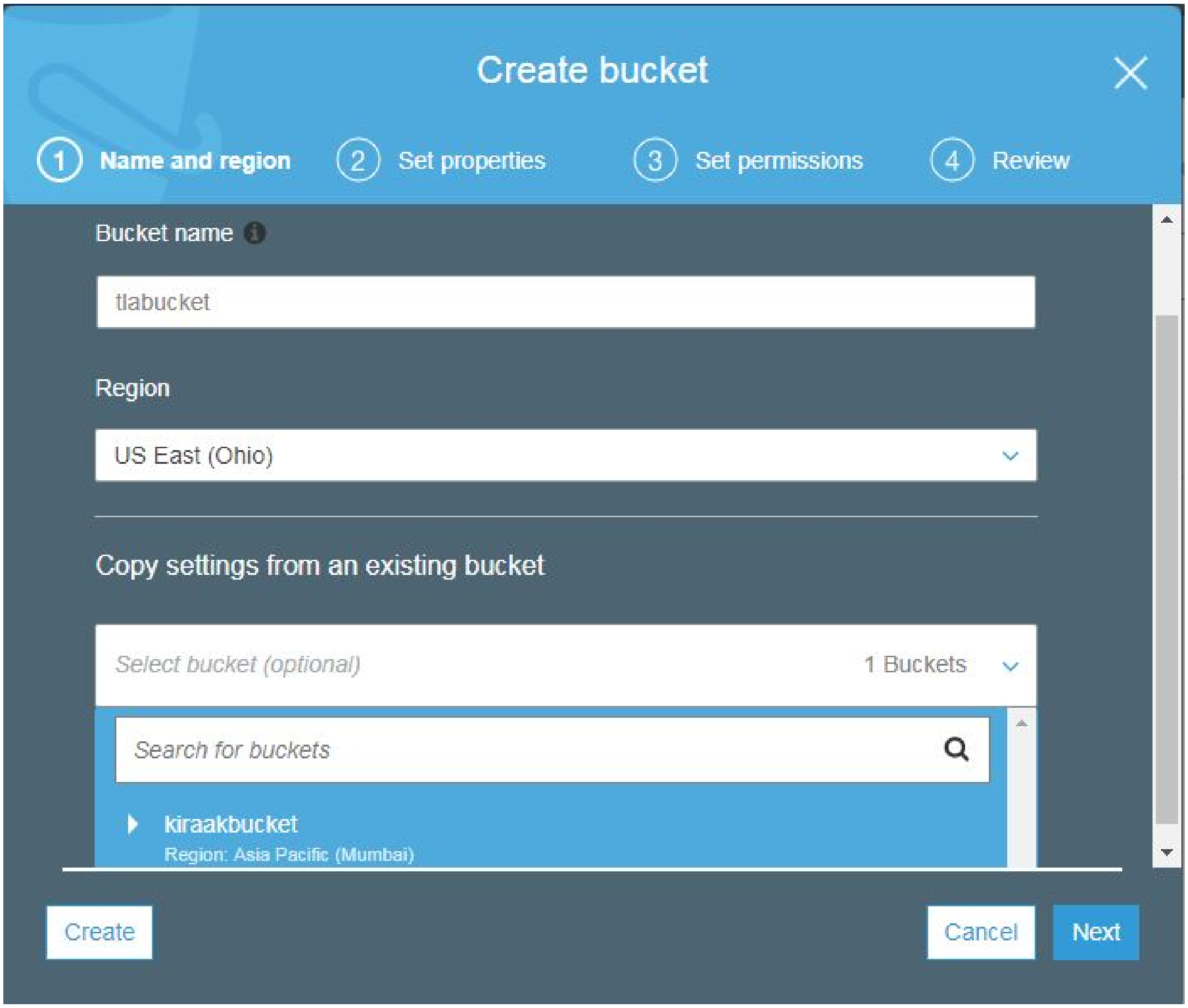
Bucket name: kiraakbucket

Region: Asia Pacific (Mumbai)

Copy settings from an existing bucket: You have no buckets 0 Buckets

Fill out the below fields
Bucket name:
Region:
Then Click on **Create**

Or you can do the below all 4 tabs



Create bucket

1 Name and region
2 Set properties
3 Set permissions
4 Review

Bucket name: tlabucket

Region: US East (Ohio)

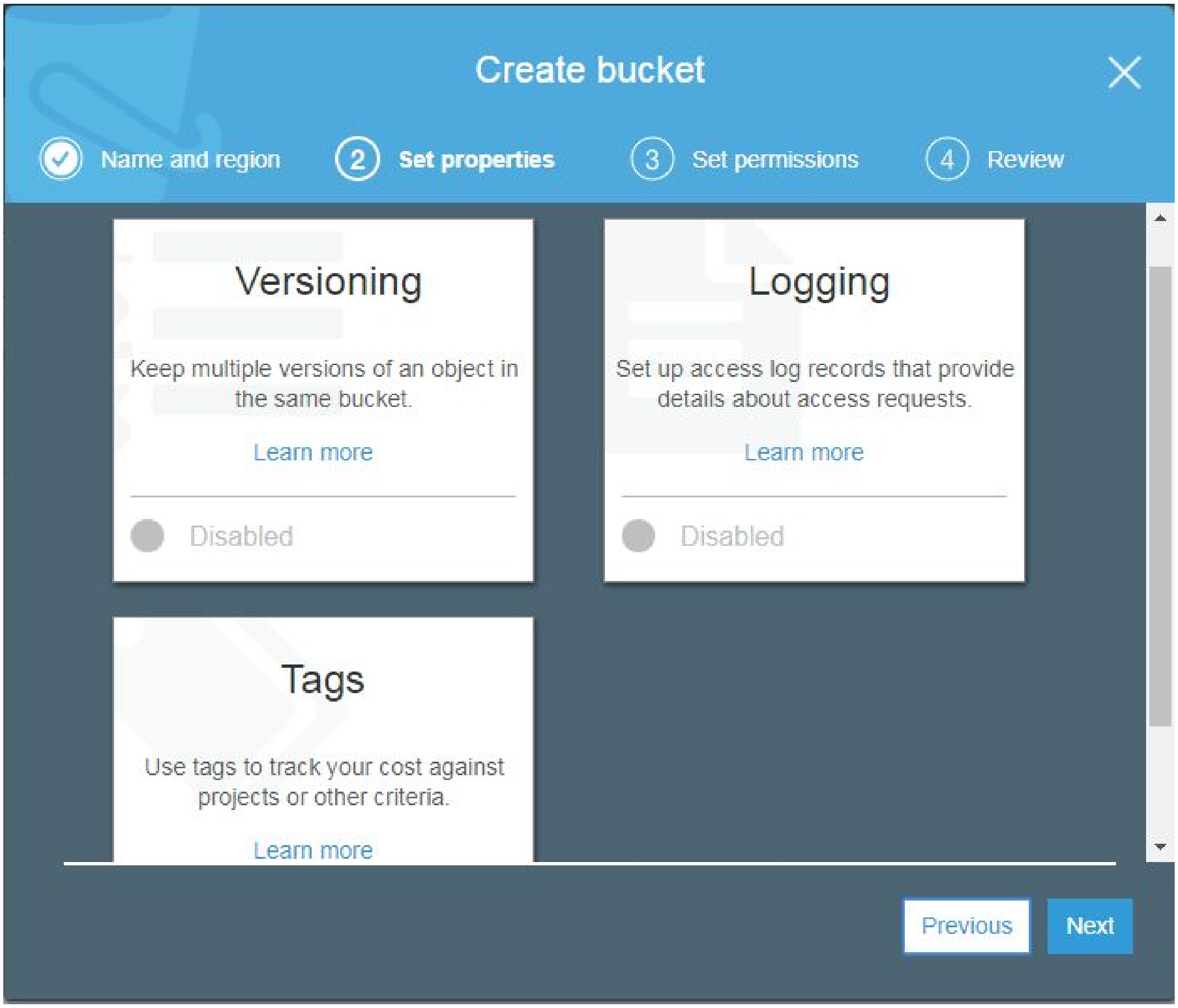
Copy settings from an existing bucket

Select bucket (optional) — 1 Buckets

Search for buckets

kiraakbucket
Region: Asia Pacific (Mumbai)

Click on Next , then you will see **properties**



Versioning

Keep multiple versions of an object in the same bucket.

Learn more

Disabled

Logging

Set up access log records that provide details about access requests.

Learn more

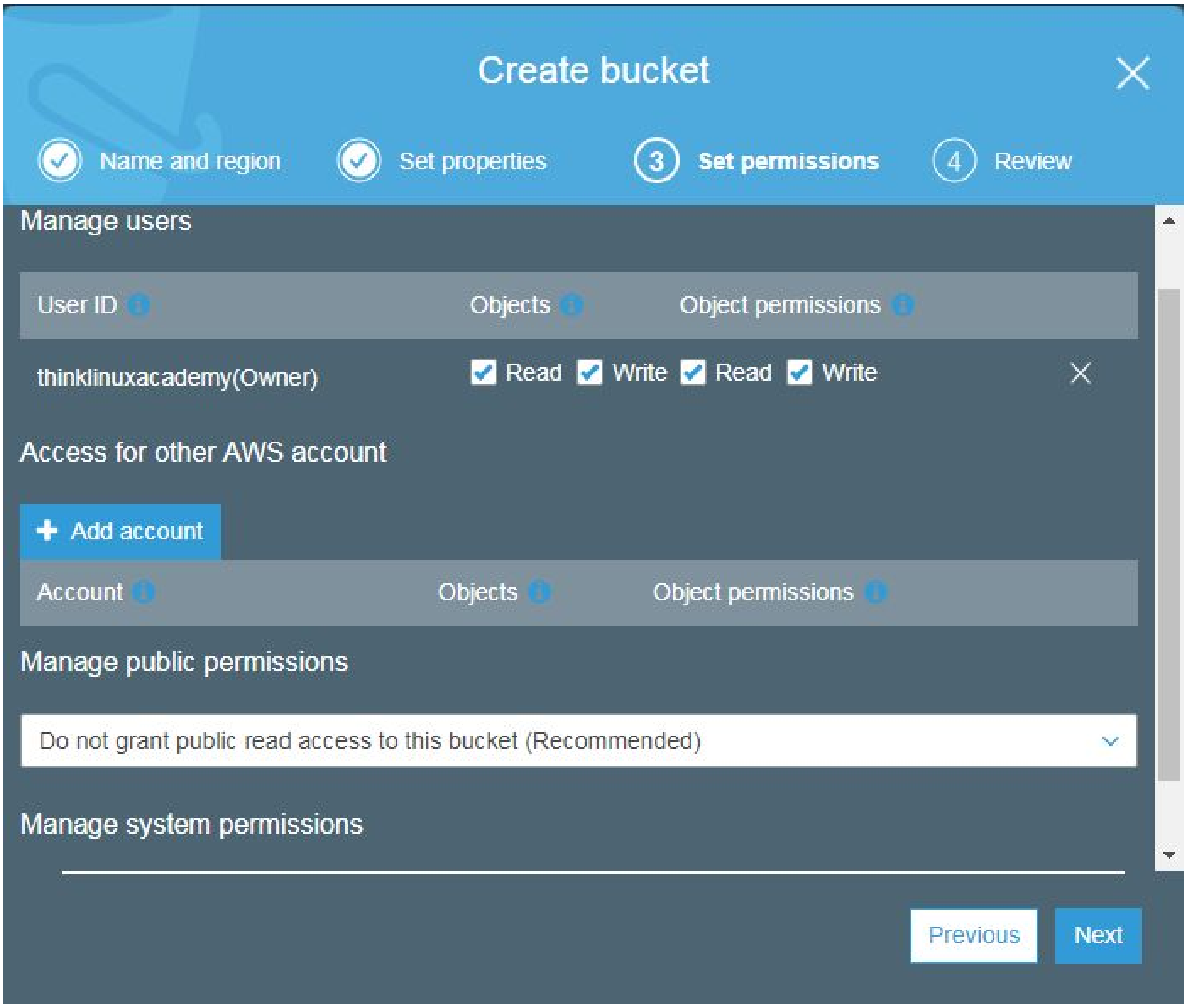
Disabled

Tags

Use tags to track your cost against projects or other criteria.

Learn more

Click on Next, then you will see **Permissions**



Create bucket

Manage users

User ID	Objects	Object permissions
thinklinuxacademy(Owner)	☑ Read ☑ Write	☑ Read ☑ Write

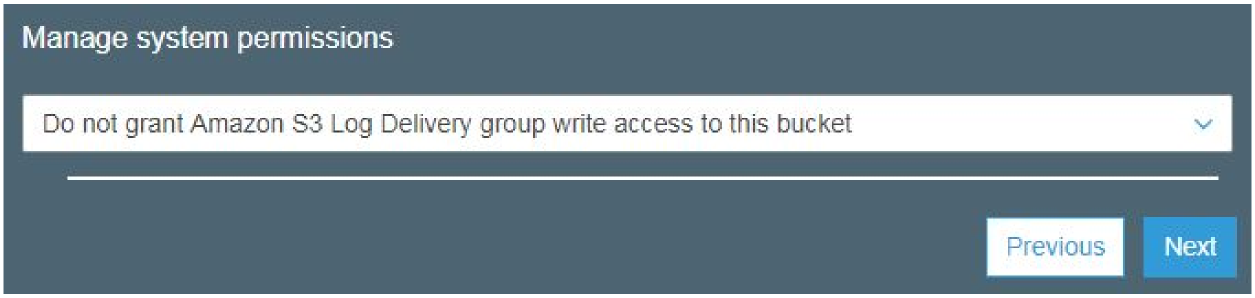
Access for other AWS account

Account	Objects	Object permissions

Manage public permissions

Do not grant public read access to this bucket (Recommended)

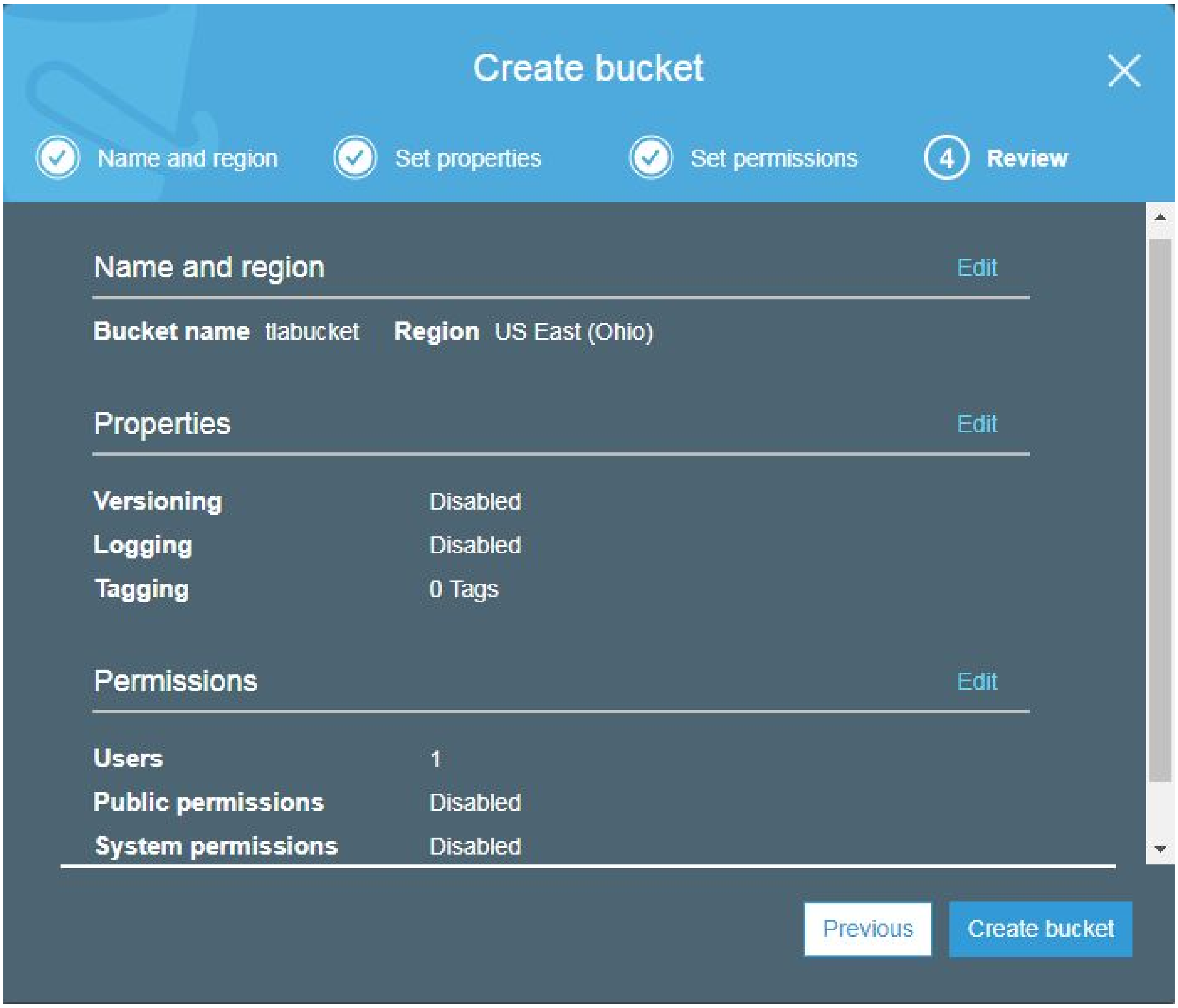
Manage system permissions



Manage system permissions

Do not grant Amazon S3 Log Delivery group write access to this bucket

Click on Next, then you will see **Review screen**



Name and region

Bucket name tlabucket **Region** US East (Ohio)

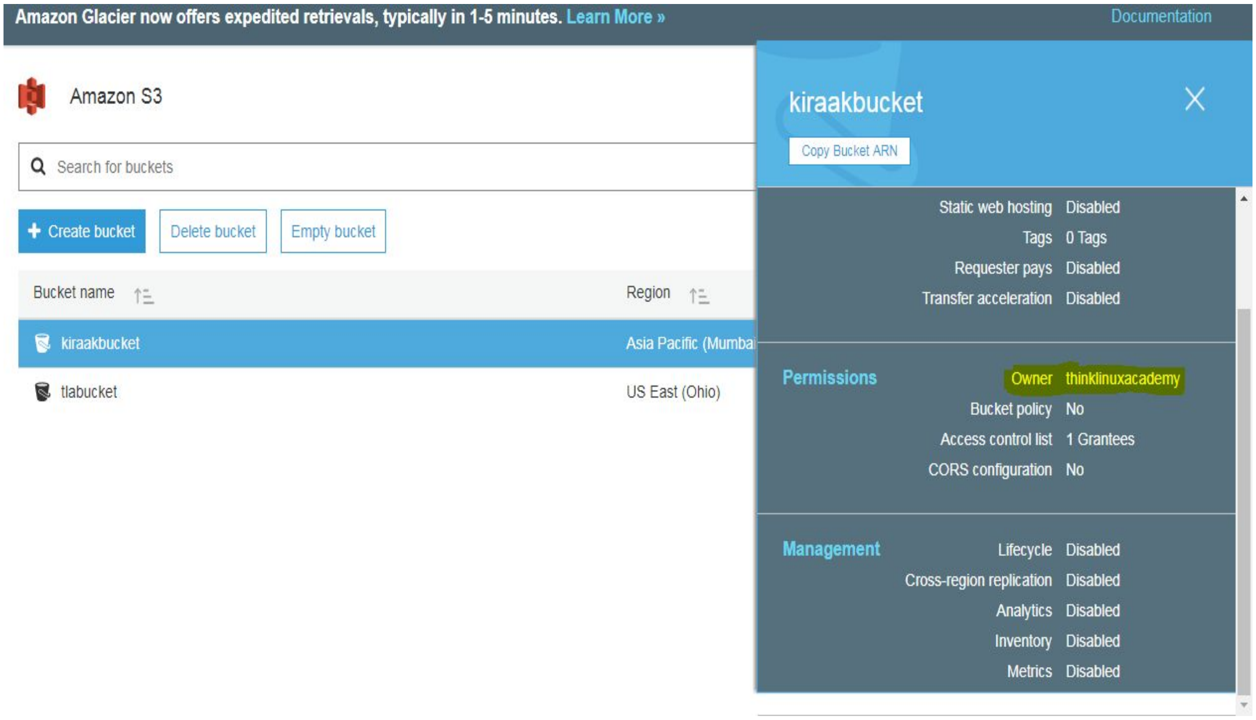
Properties

Versioning	Disabled
Logging	Disabled
Tagging	0 Tags

Permissions

Users	1
Public permissions	Disabled
System permissions	Disabled

Then click on Create bucket, THE BUCKET IS CREATED



Bucket name	Region
kiraakbucket	Asia Pacific (Mumbai
tlabucket	US East (Ohio)

kiraakbucket

Static web hosting Disabled
Tags 0 Tags
Requester pays Disabled
Transfer acceleration Disabled

Permissions
Owner thinklinuxacademy
Bucket policy No
Access control list 1 Grantees
CORS configuration No

Management
Lifecycle Disabled
Cross-region replication Disabled
Analytics Disabled
Inventory Disabled
Metrics Disabled

You can see the ownership of the bucket

S3 Objects

S3 Objects

- Objects are static files that contain metadata information :
- Set of name-key pairs
- Contain information specified by the user, and AWS information such as storage type
- Each object must be assigned a **storage type,** which determines the object's **availability, durability and cost**
- By default, all objects are private (means not public)

Object facts

- Be as small as 0 bytes and as large as 5TB
- Can have multiple versions (if versioning is enabled)
- Be made publicly available via a URL
- Automatically switch to a different storage class or deleted (via lifecycle policies)
- Encrypted
- Organized into "sub-name" spaces called folders

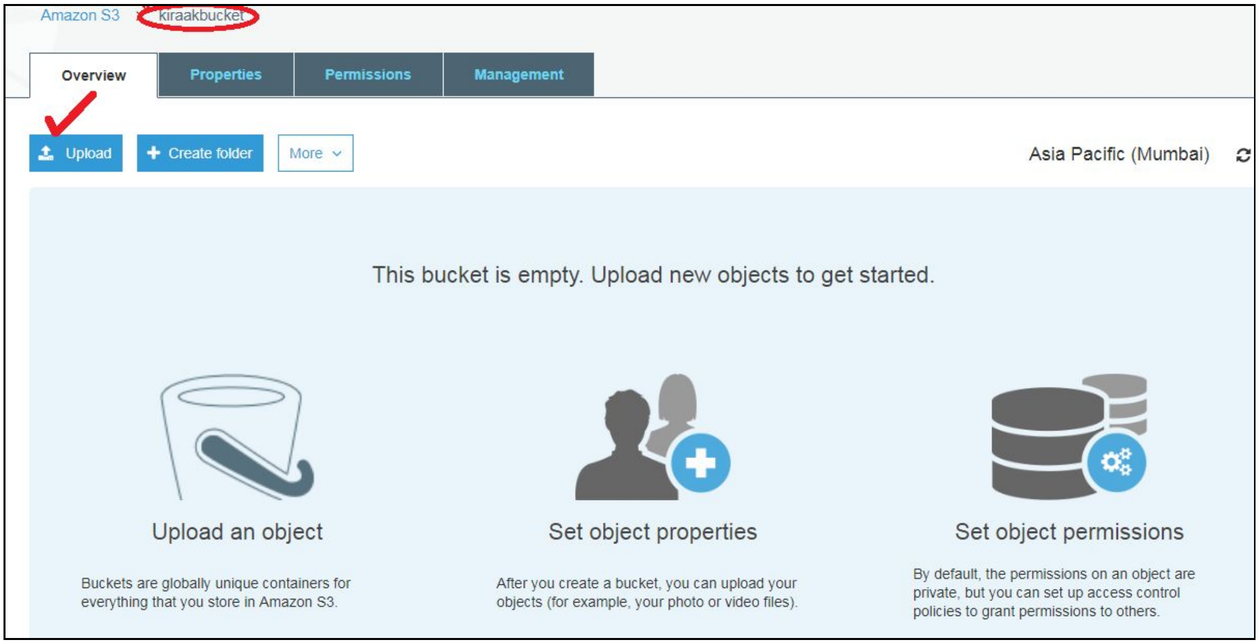
Object Encryption

- SSE (Server Side Encryption)
- S3 can encrypt the object before saving it on the partitions in the data centers and decrypt it when it is downloaded
- AES-256
- SSE-KMS
- Also you can use your own encryption keys
- Considered client side encryption where you encrypt the data before upload

- SSL terminated endpoints for the API

LAB 2 : Create a object in the Bucket

First go into the bucket → then upload object (suppose one photo)



This bucket is empty. Upload new objects to get started.

Upload an object

Buckets are globally unique containers for everything that you store in Amazon S3.

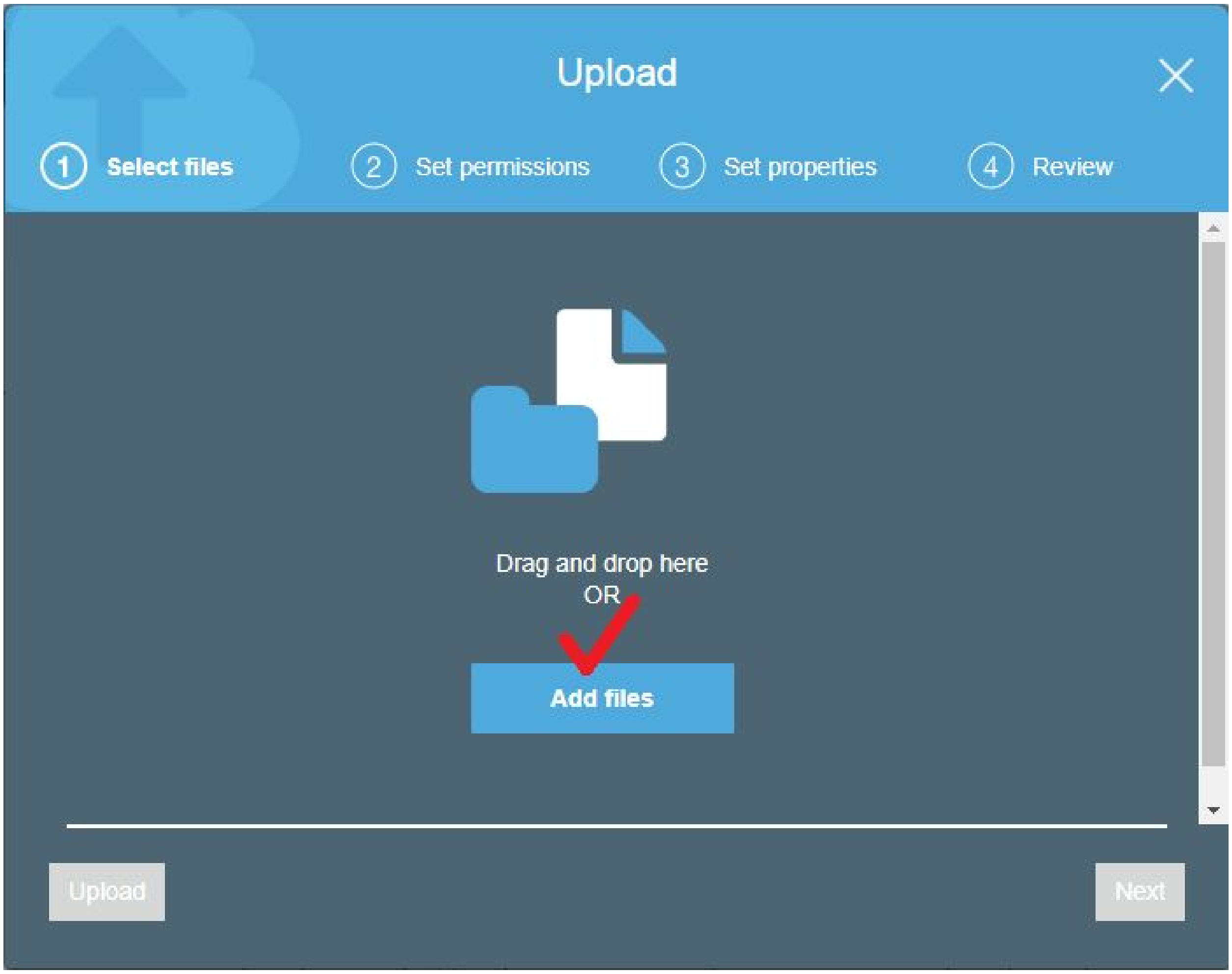
Set object properties

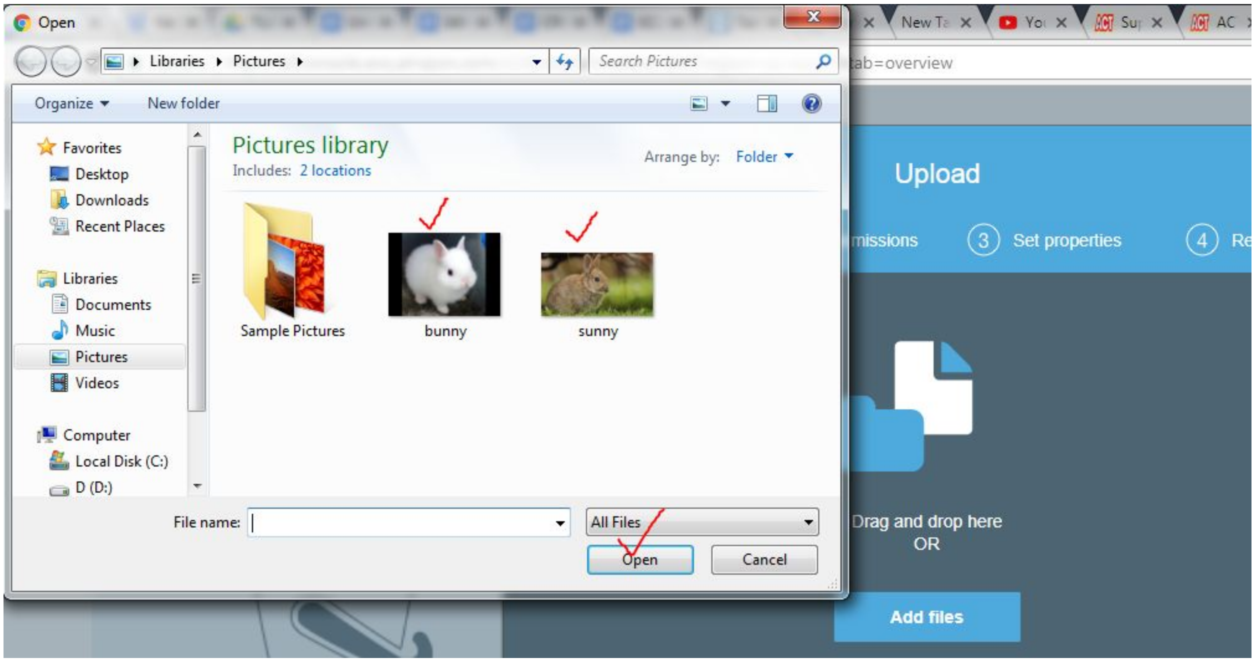
After you create a bucket, you can upload your objects (for example, your photo or video files).

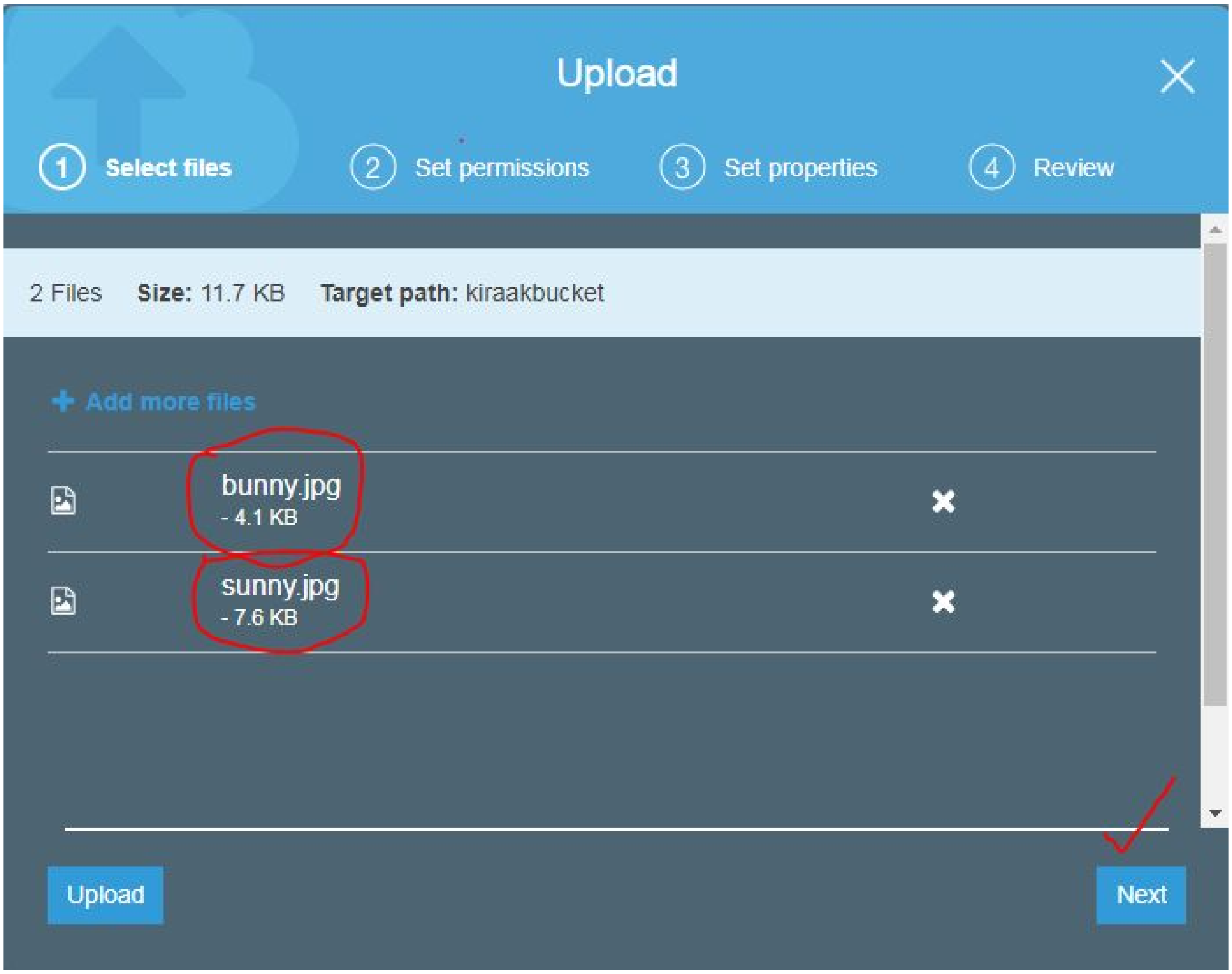
Set object permissions

By default, the permissions on an object are private, but you can set up access control policies to grant permissions to others.

Click on **Upload,** then browse and select the photo


Upload
1 Select files
2 Set permissions
3 Set properties
4 Review
Drag and drop here
OR
Add files
Upload
Next

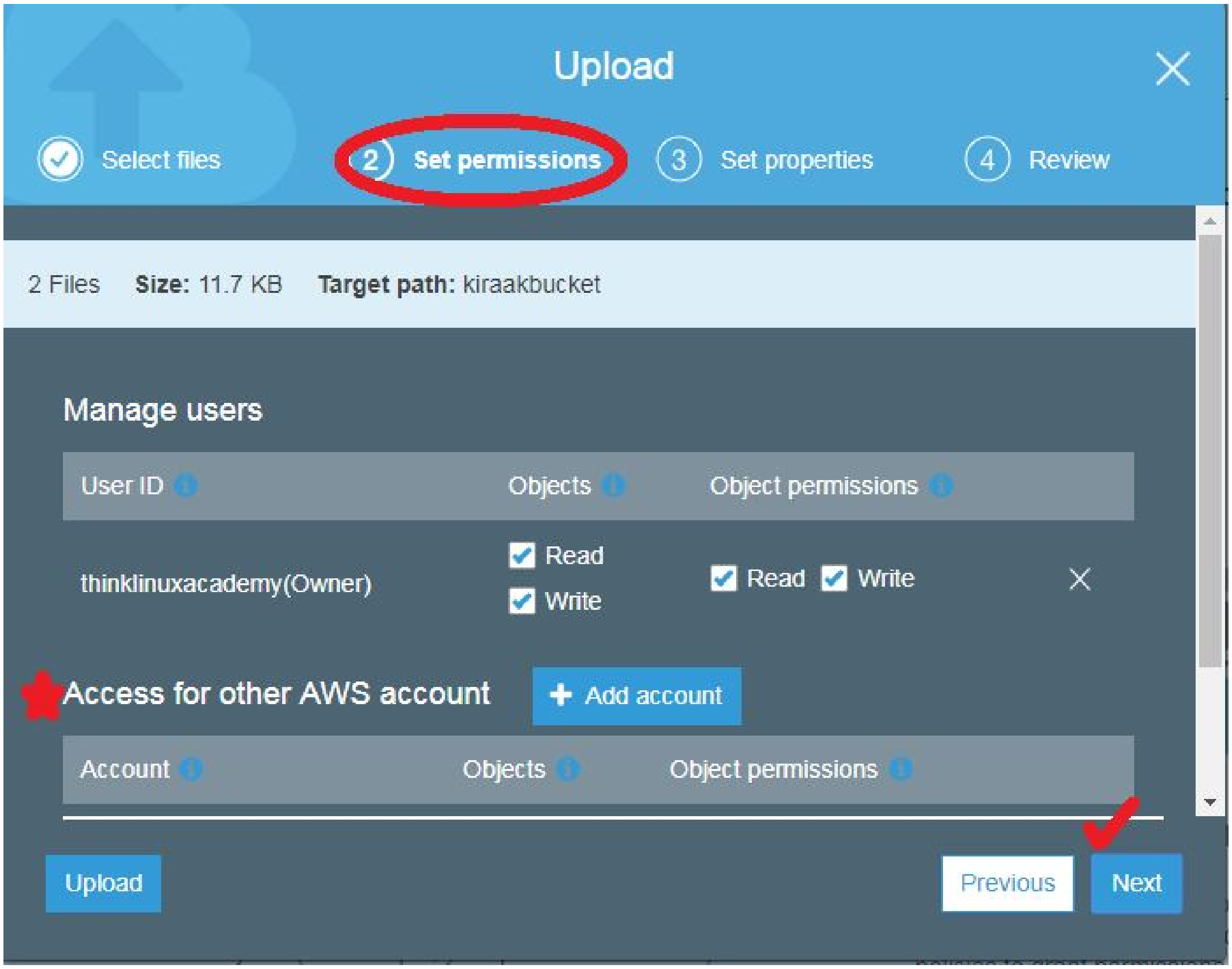

Open
New Ta
Yo
Su
AC
Libraries ▸ Pictures ▸
Search Pictures
ab=overview
Organize
New folder
Favorites
Desktop
Downloads
Recent Places
Libraries
Documents
Music
Pictures
Videos
Computer
Local Disk (C:)
D (D:)
Pictures library
Includes: 2 locations
Arrange by: Folder
Sample Pictures
bunny
sunny
File name:
All Files
Open
Cancel
Upload
missions
3 Set properties
4 Re
Drag and drop here
OR
Add files



2 Files **Size:** 11.7 KB **Target path:** kiraakbucket

bunny.jpg - 4.1 KB

sunny.jpg - 7.6 KB



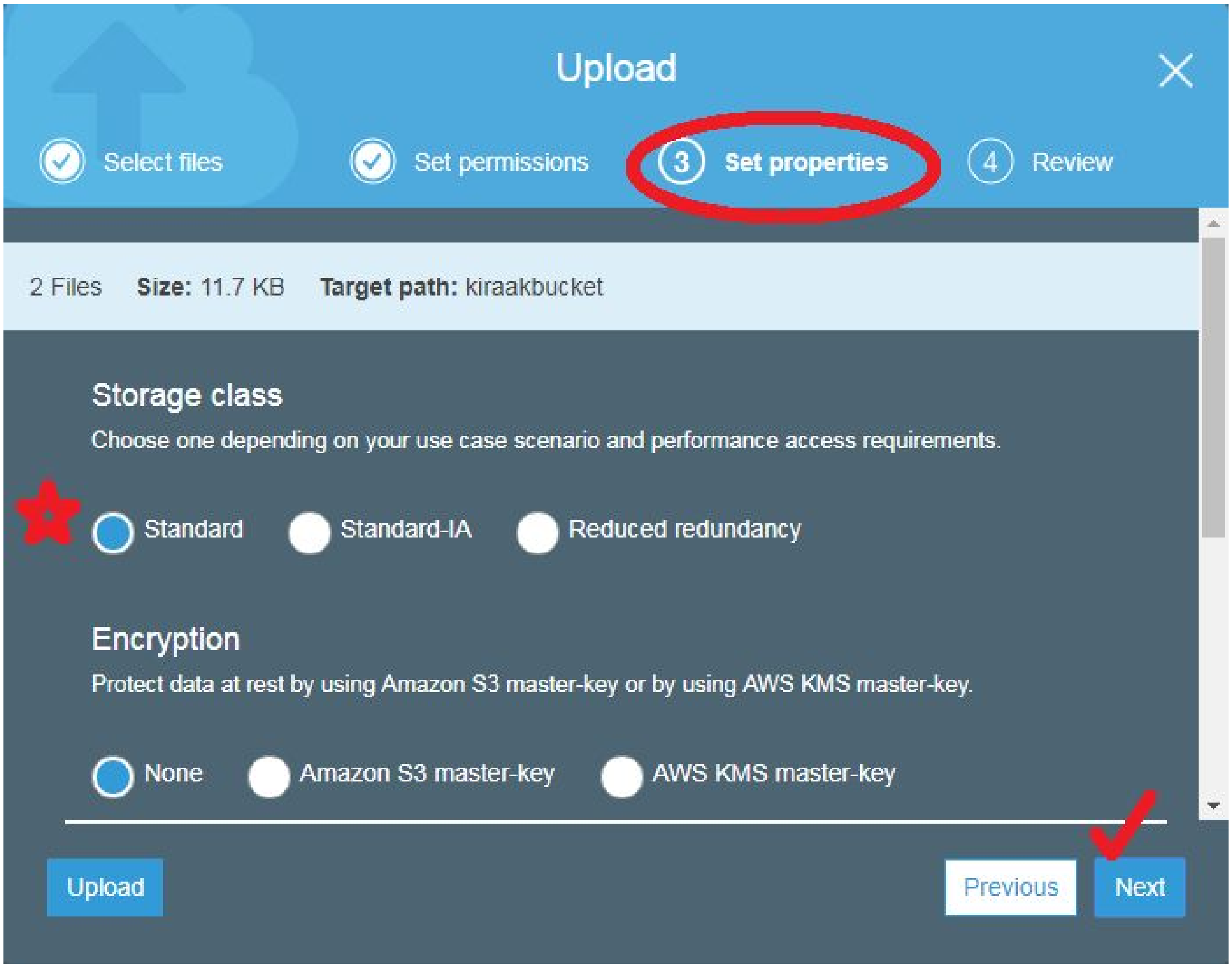
2 Files **Size:** 11.7 KB **Target path:** kiraakbucket

Manage users

User ID	Objects	Object permissions
thinklinuxacademy(Owner)	☑ Read ☑ Write	☑ Read ☑ Write

Access for other AWS account

Account	Objects	Object permissions



2 Files **Size:** 11.7 KB **Target path:** kiraakbucket

Storage class

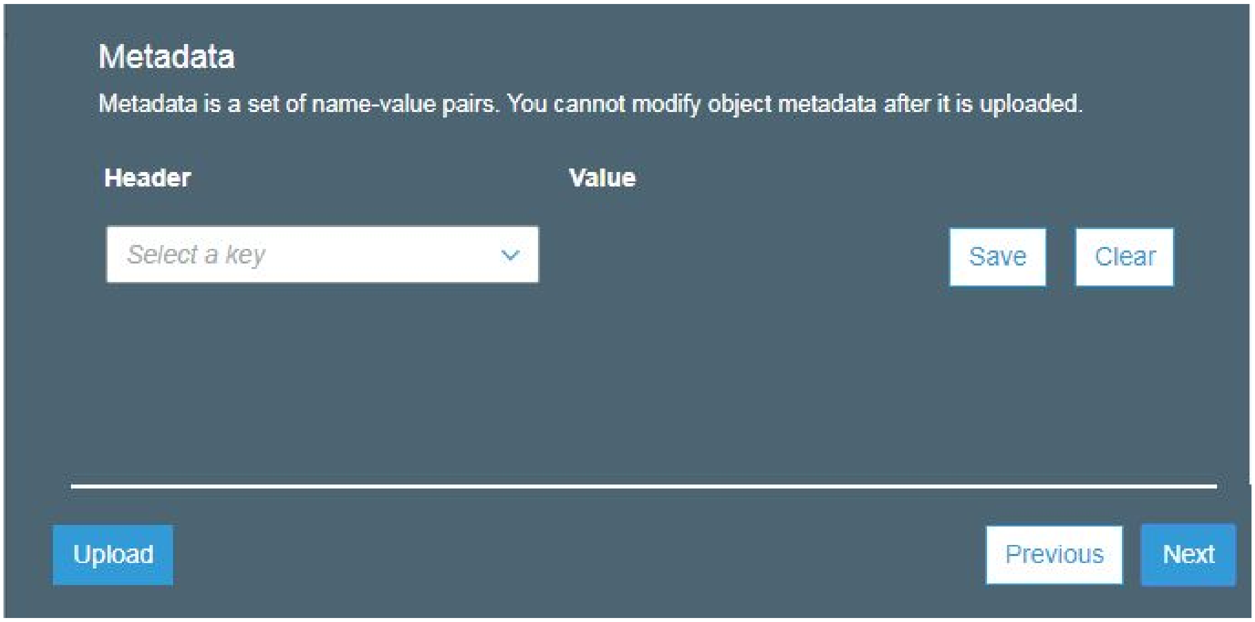
Choose one depending on your use case scenario and performance access requirements.

● Standard ○ Standard-IA ○ Reduced redundancy

Encryption

Protect data at rest by using Amazon S3 master-key or by using AWS KMS master-key.

● None ○ Amazon S3 master-key ○ AWS KMS master-key

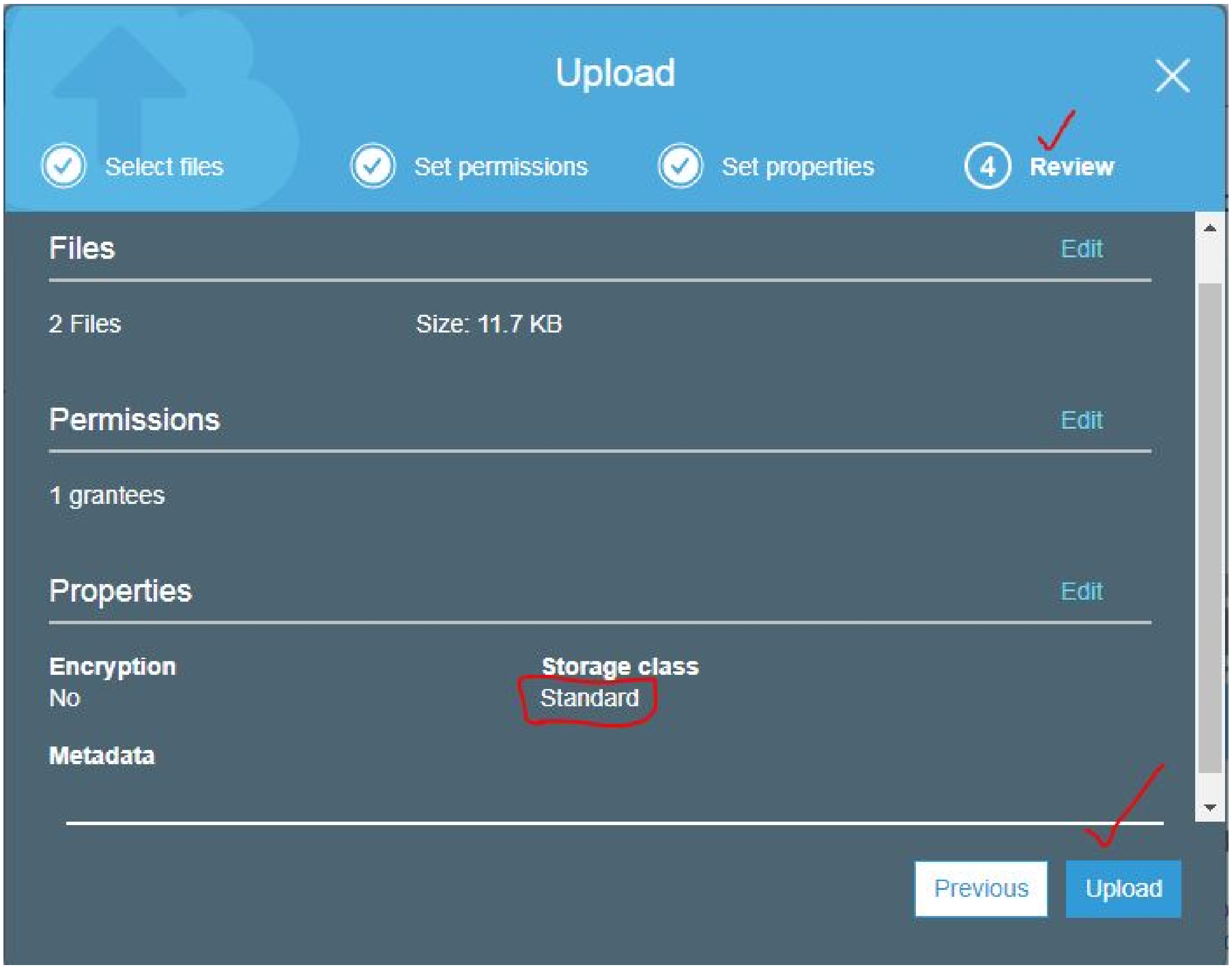


Metadata

Metadata is a set of name-value pairs. You cannot modify object metadata after it is uploaded.

Header	Value
Select a key	

Don’t select metadata, click on Next



Files

2 Files Size: 11.7 KB

Permissions

1 grantees

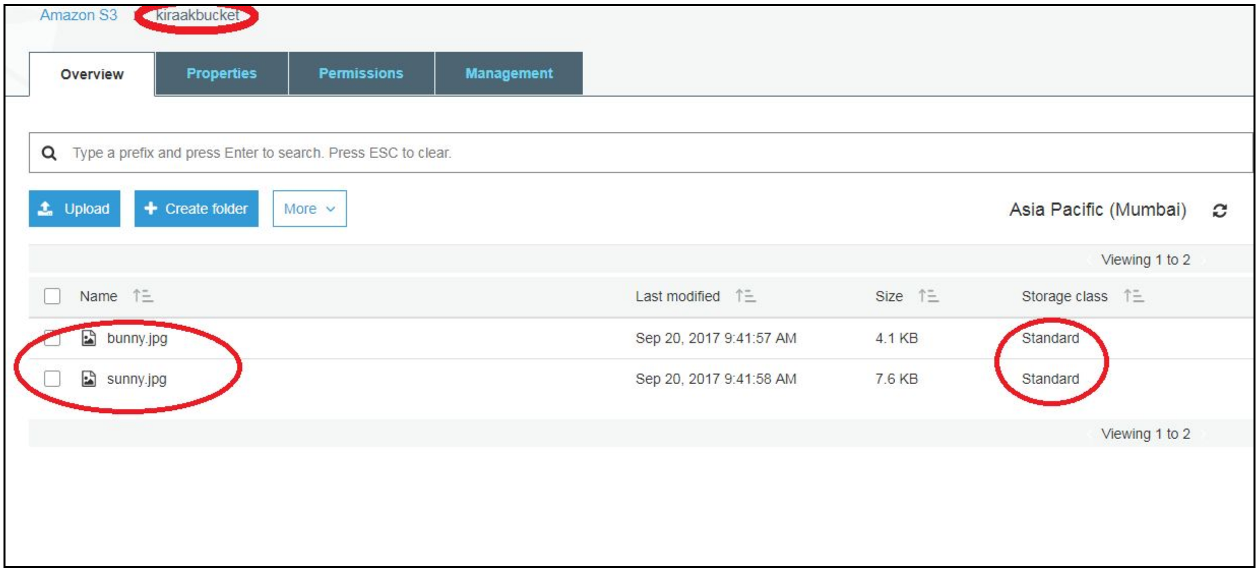
Properties

Encryption
No

Storage class
Standard

Metadata

So now you can see the bucket and its content



□	Name	Last modified	Size	Storage class
□	bunny.jpg	Sep 20, 2017 9:41:57 AM	4.1 KB	Standard
□	sunny.jpg	Sep 20, 2017 9:41:58 AM	7.6 KB	Standard

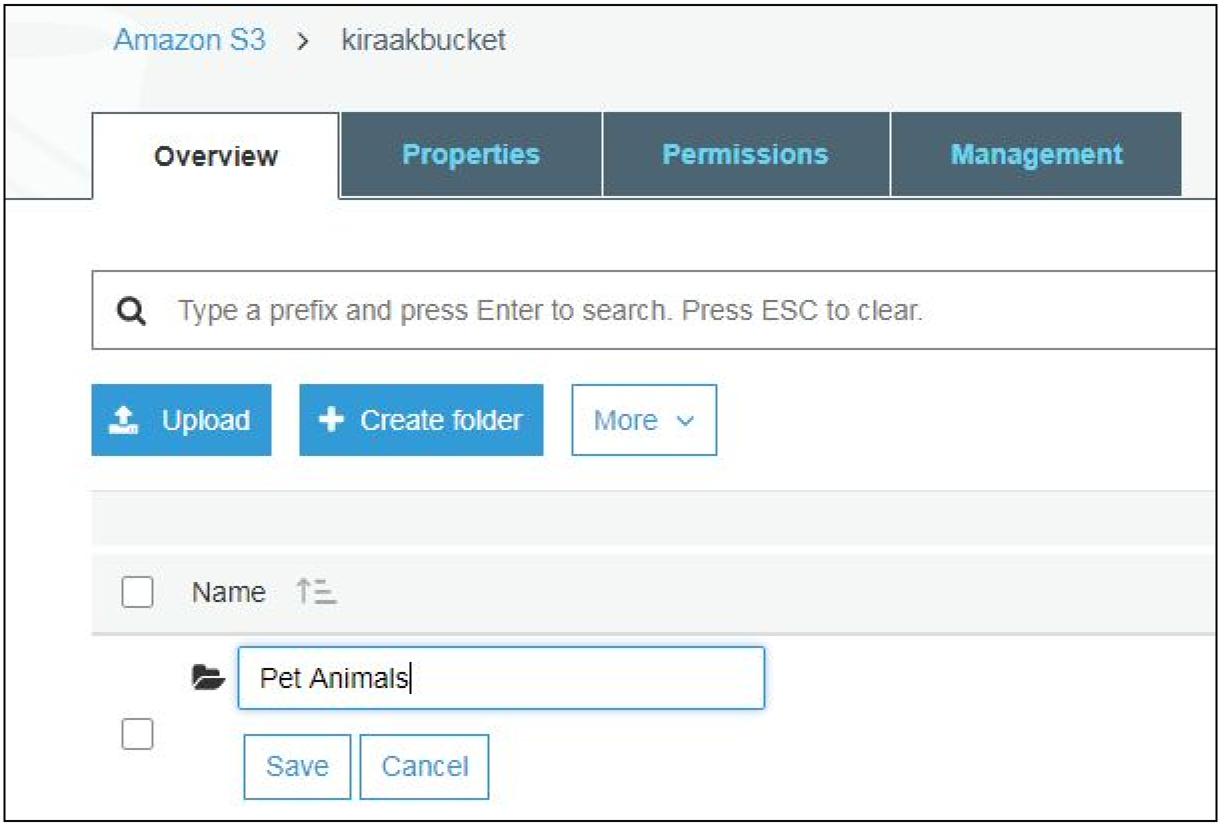
S3 Folders

S3 Folders

- For simplicity purpose, S3 supports the concept of “folders”
- This is done only as a means of grouping objects
- Amazon S3 does this by using key-name prefixes for objects

NOTE : Amazon S3 has a flat structure, there is no hierarchy like you would see in typical File System

Lab:



Pet Animals

But this is just done visually. At the backend for developers purpose, it is a flat file architecture. If i upload any file in this folder then it is technically it is exactly at the same level as the other files in bucket. It is basically attaching key-name prefixes

S3 Security

Till now we have seen Bucket is created, and upload an object into this bucket. Now time to see the permissions

S3 Permissions

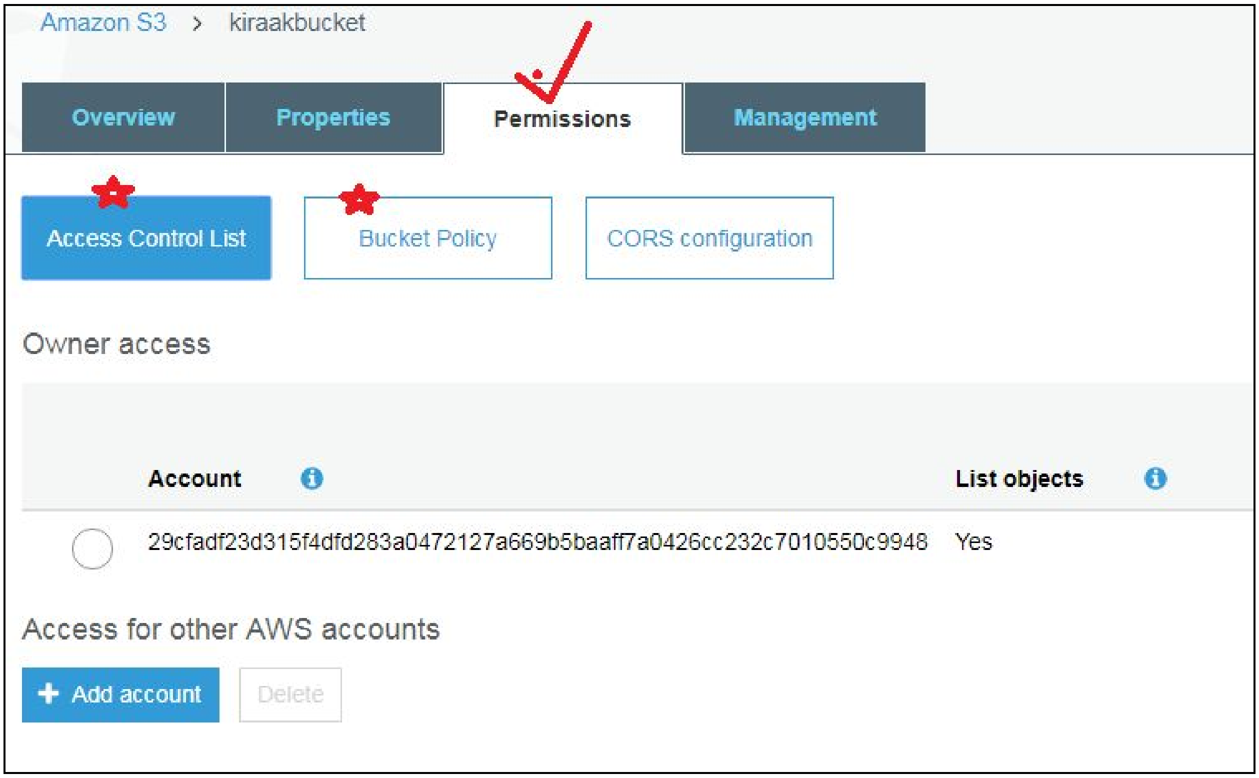
- All buckets and objects are private (not public) by default - only the resource owner has access.
- The resource owner can grant access to the resource (bucket/objects) through "S3 resource based policies" OR access can be granted via traditional IAM user policy

S3 Resource based policies are:

- **Bucket Policies:**
- It is the policies that are attached only to the S3 bucket (not an IAM user)
- The permissions are applied to all objects in the bucket
- The policy specifies what actions are allowed or denied for a particular user at that bucket

- **S3 ACL based policies**
- Grant access to the users in other AWS accounts or to the public
- Both buckets and objects has ACLs
- Object ACL allow us to share an S3 object with the public via a URL link

LAB 3: Try to Access that objects from browser(Permission)



Owner access

	Account	List objects
○	29cfadf23d315f4dfd283a0472127a669b5baaff7a0426cc232c7010550c9948	Yes

Access for other AWS accounts

Now go the objects and click on link
By default u can't open in public so give the permission as "make public" then u can access from anywhere.

Also take the access back.

S3 Storage Class

S3 Storage Class:

- A storage class represents the “classification” assigned to each Object in S3. There are 4 types :

1. Standard
2. Reduced Redundancy Storage (RRS)
3. Infrequently Access (S3-IA)
4. Glacier

- Each storage class has varying attributes that dictate things like

- Cost
- Object Availability
- Object Durability
- Frequency of access (to the object)

STANDARD

→ Designed for general, all-purpose storage
→ It is the default storage option
→ 99.999999999% object durability(eleven 9’s)
→ 99.99% of object availability
→ It is the most expensive storage class

Reduced Redundancy (RRS)

→ Designed for non-critical, reproducible objects
→ 99.99% object durability
→ 99.99% object availability
→ it is less expensive than the standard

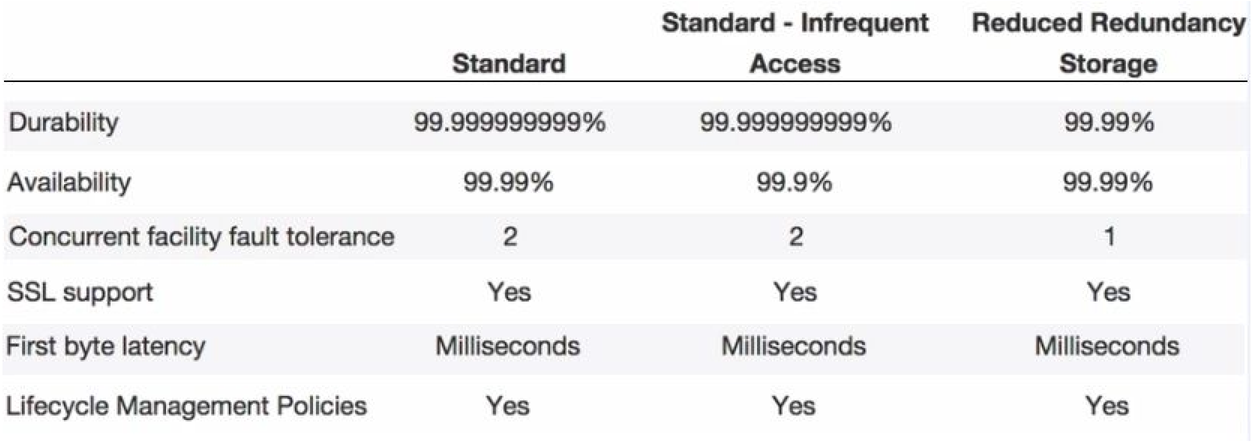
Infrequent Access (S3-IA)

→ Designed for objects that you do not frequently access, but must be immediately available when accessed
→ 99.999999999% object durability
→ 99.90% object availability
→ It is less expensive than the Standard/RRS

Glacier

- Designed for long term archival storage (not to be used for backup)
- May take several hours for objects stored in glacier to be retrieved
- 99.999999999% of object durability
- It is cheapest S3 storage class among all (very low cost)

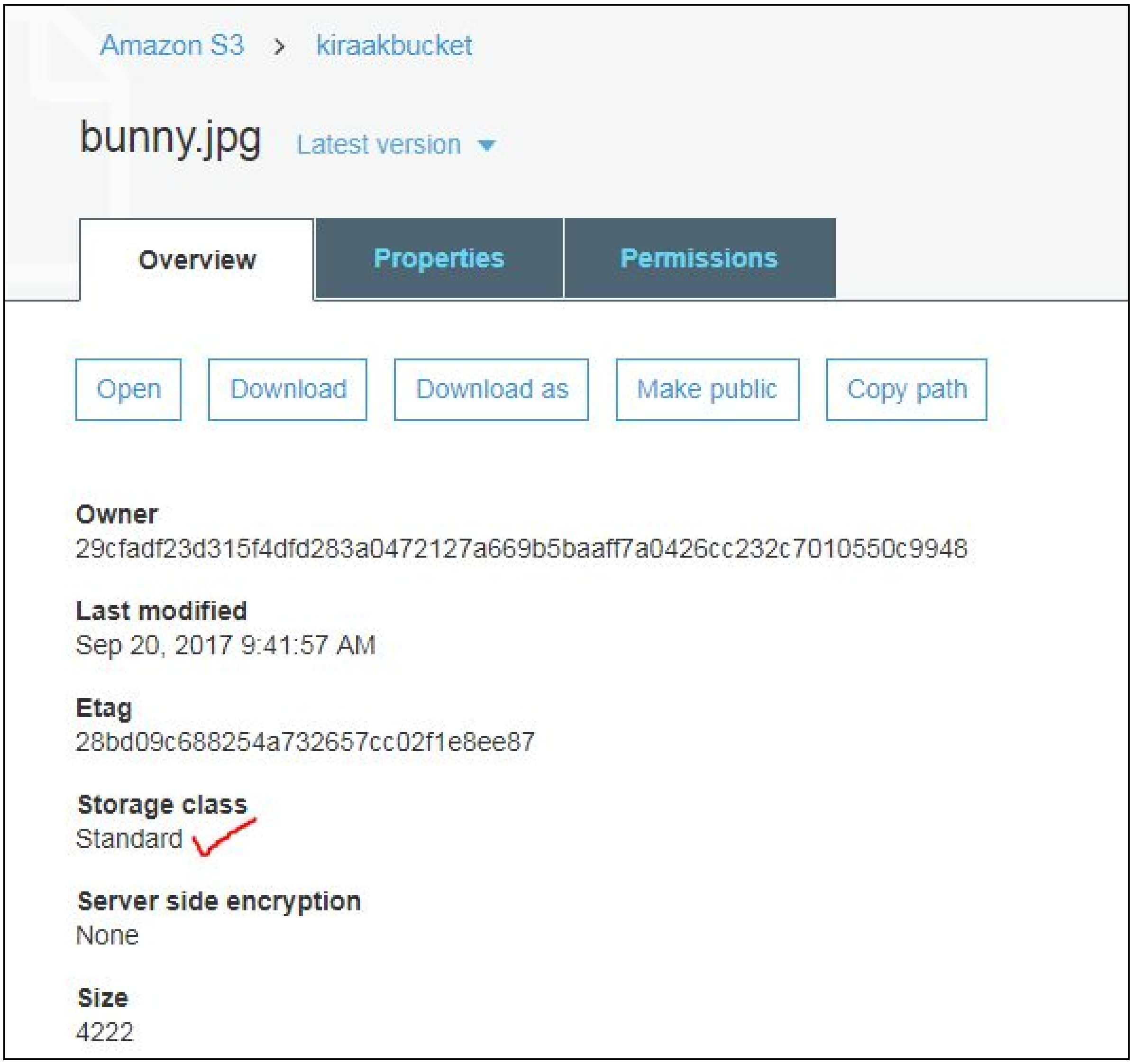
Storage classes features in table



	Standard	Standard - Infrequent Access	Reduced Redundancy Storage
Durability	99.999999999%	99.999999999%	99.99%
Availability	99.99%	99.9%	99.99%
Concurrent facility fault tolerance	2	2	1
SSL support	Yes	Yes	Yes
First byte latency	Milliseconds	Milliseconds	Milliseconds
Lifecycle Management Policies	Yes	Yes	Yes

LAB 4: Selecting Storage Type

Go to the object under bucket and then select properties tab
Select storage class



Owner
29cfadf23d315f4dfd283a0472127a669b5baaff7a0426cc232c7010550c9948

Last modified
Sep 20, 2017 9:41:57 AM

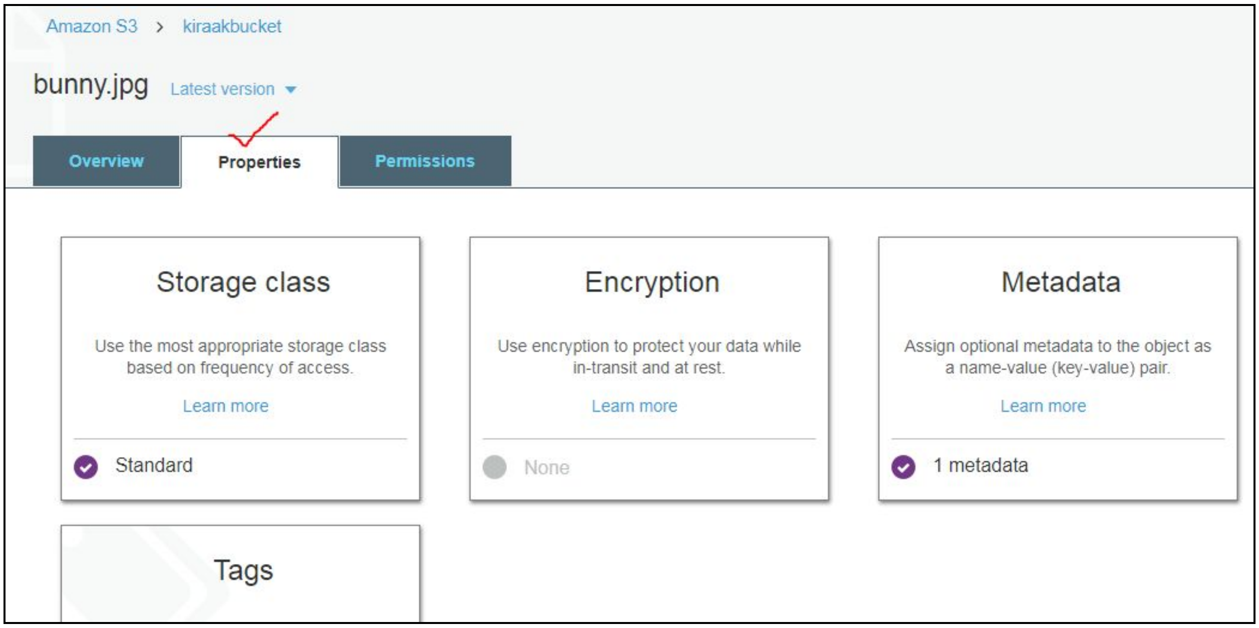
Etag
28bd09c688254a732657cc02f1e8ee87

Storage class
[✓ Standard]

Server side encryption
None

Size
4222

Click on Properties



Amazon S3 > kiraakbucket

bunny.jpg Latest version

Storage class

Use the most appropriate storage class based on frequency of access.

Learn more

✓ Standard

Encryption

Use encryption to protect your data while in-transit and at rest.

Learn more

● None

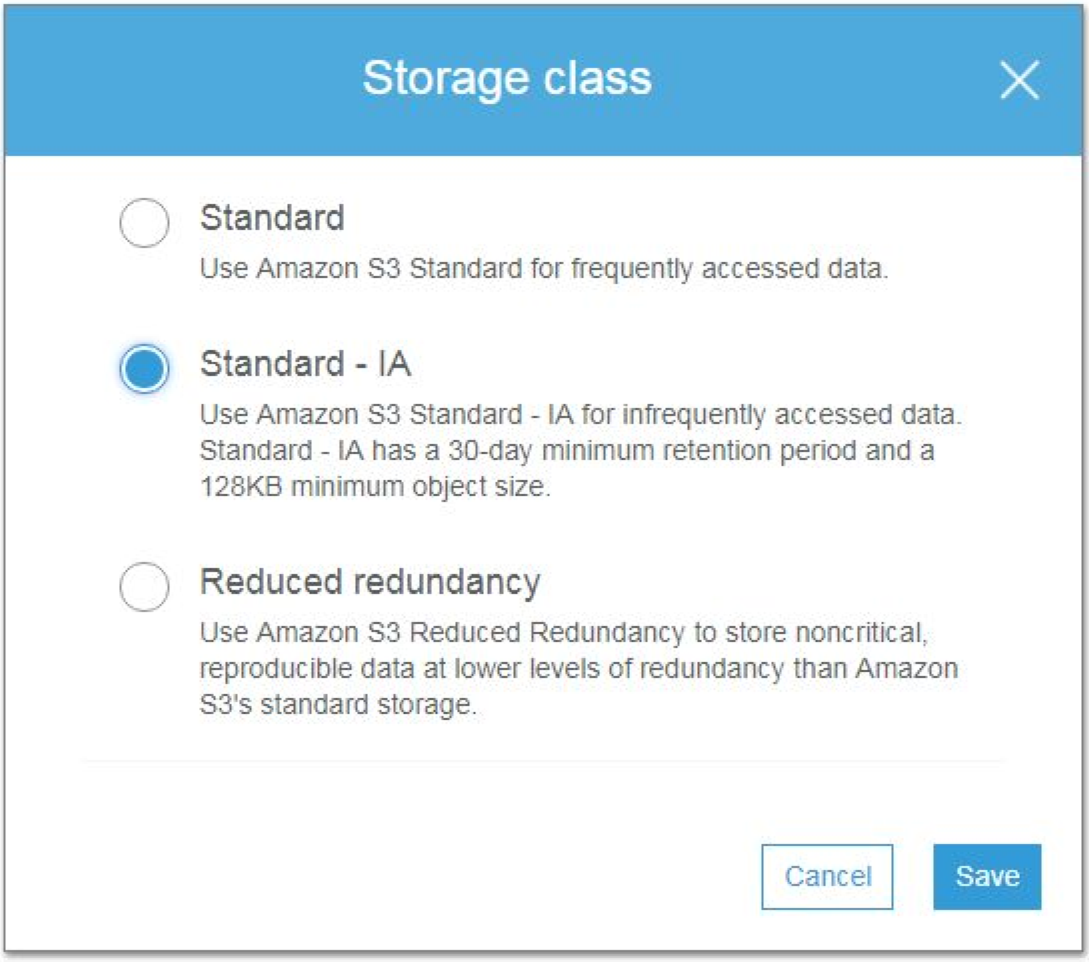
Metadata

Assign optional metadata to the object as a name-value (key-value) pair.

Learn more

✓ 1 metadata

Tags



Storage class

○ Standard

Use Amazon S3 Standard for frequently accessed data.

● Standard - IA

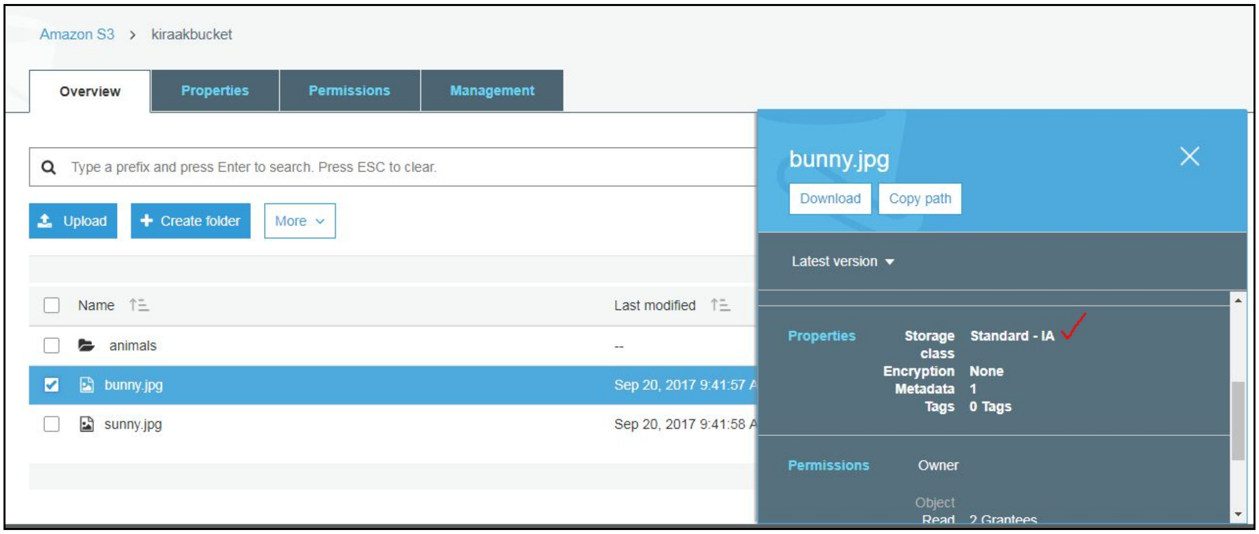
Use Amazon S3 Standard - IA for infrequently accessed data. Standard - IA has a 30-day minimum retention period and a 128KB minimum object size.

○ Reduced redundancy

Use Amazon S3 Reduced Redundancy to store noncritical, reproducible data at lower levels of redundancy than Amazon S3's standard storage.

Click on save

You can verify



	Name	Last modified
□	animals	--
☑	bunny.jpg	Sep 20, 2017 9:41:57 A
□	sunny.jpg	Sep 20, 2017 9:41:58 A

Properties

Storage class	Standard - IA ✓
Encryption	None
Metadata	1
Tags	0 Tags

Permissions

Owner

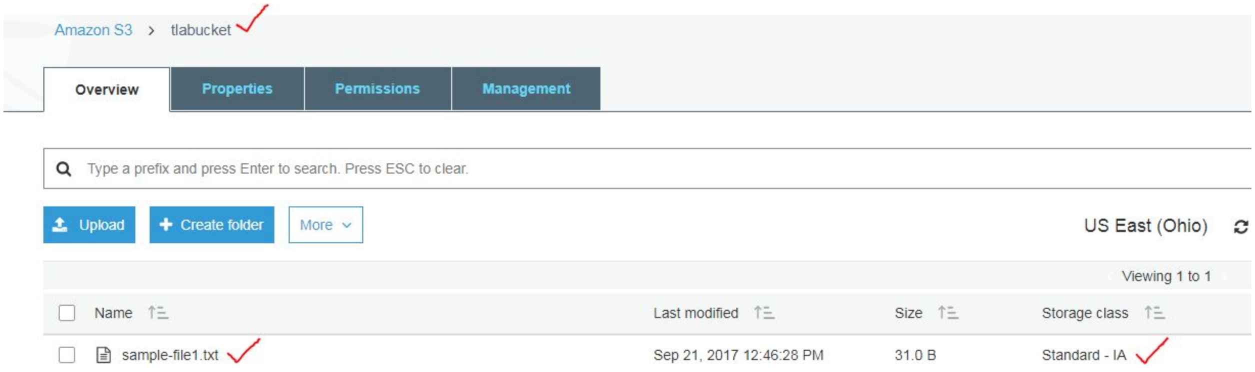
Object Read 2 Grantees

Object Versioning

- S3 versioning is a feature to manage and store all old/new/deleted versions of an object.
- By default, versioning is disabled on all buckets/objects
- Once versioning is enabled, you can only suspend versioning. It cannot be fully disabled
- Suspending versioning only prevents new versions from being created. All objects with existing versions will maintain their older versions
- Versioning can only be set on the bucket level and applies to ALL objects in the bucket
- Lifecycle policies can be applied to specific versions of an object
- Versioning and lifecycle policies can both be enabled on a bucket at the same time
- Versioning can be used with lifecycle policies to create a great archiving and backup solutions in s3

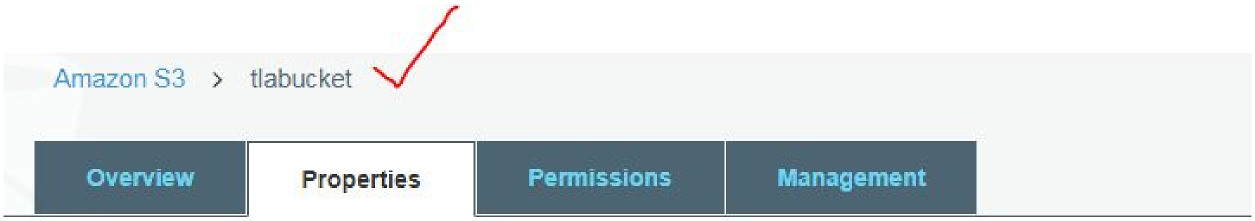
Lab 5: Versioning Exercise

Go to the bucket and upload one file

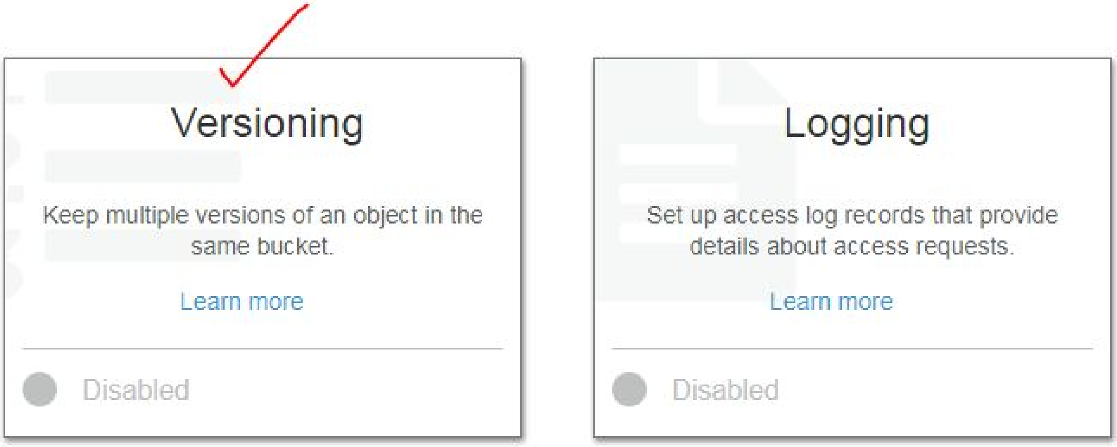


Name	Last modified	Size	Storage class
sample-file1.txt	Sep 21, 2017 12:46:28 PM	31.0 B	Standard - IA

Now enable the versioning in the bucket and then change the file



Amazon S3 > tlabucket ✓



✓

Versioning

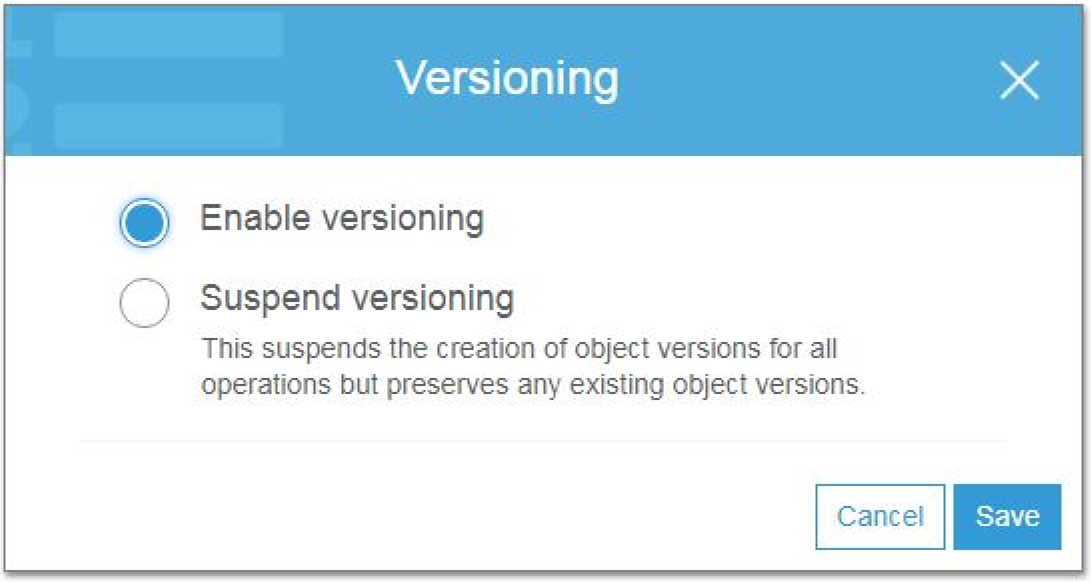
Keep multiple versions of an object in the same bucket.

● Disabled

Logging

Set up access log records that provide details about access requests.

● Disabled



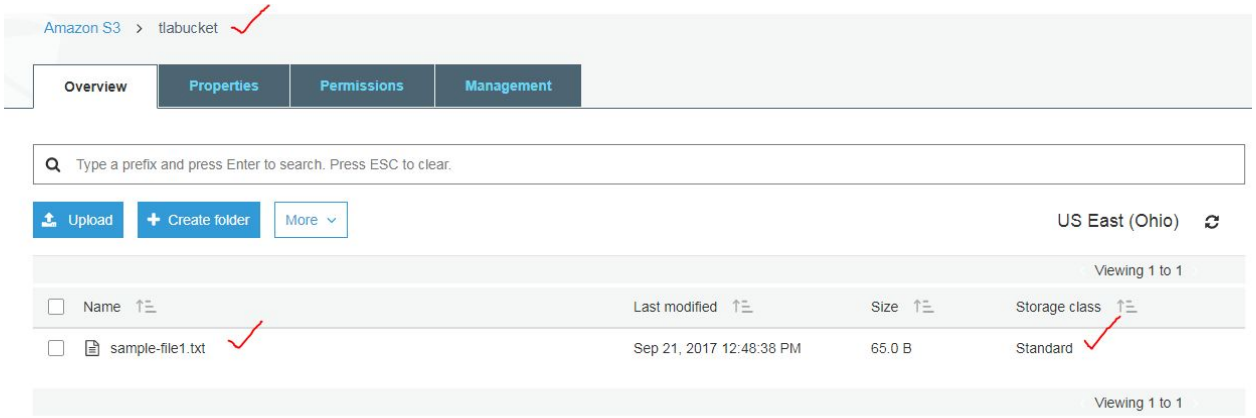
Versioning

● Enable versioning

○ Suspend versioning

This suspends the creation of object versions for all operations but preserves any existing object versions.

Now change the file content and again upload same file in bucket



Amazon S3 > tlabucket ✓

Overview | Properties | Permissions | Management

Type a prefix and press Enter to search. Press ESC to clear.

Upload | Create folder | More

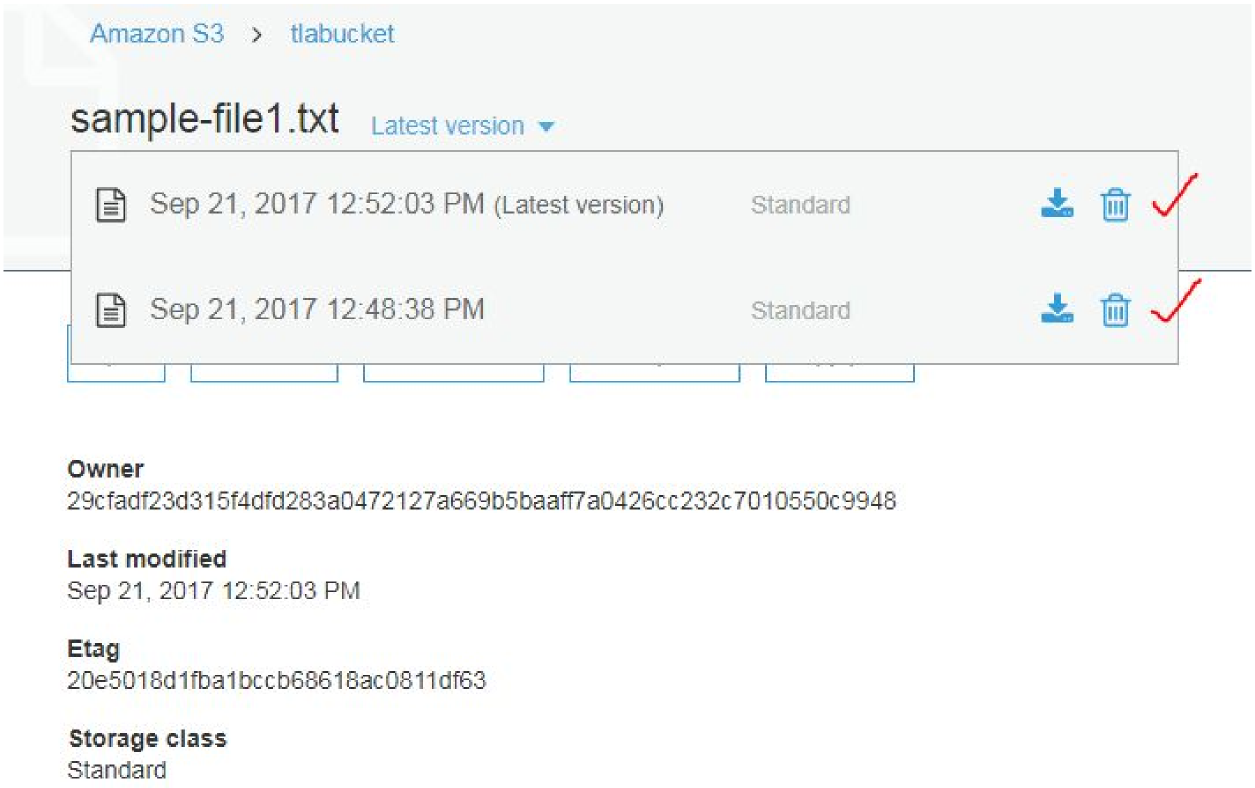
US East (Ohio)

Viewing 1 to 1

□ Name	Last modified	Size	Storage class
□ sample-file1.txt ✓	Sep 21, 2017 12:48:38 PM	65.0 B	Standard ✓

Viewing 1 to 1

This is the new file



Amazon S3 > tlabucket

sample-file1.txt Latest version

Sep 21, 2017 12:52:03 PM (Latest version)	Standard	✓
Sep 21, 2017 12:48:38 PM	Standard	✓

Owner
29cfadf23d315f4dfd283a0472127a669b5baaff7a0426cc232c7010550c9948

Last modified
Sep 21, 2017 12:52:03 PM

Etag
20e5018d1fba1bccb68618ac0811df63

Storage class
Standard

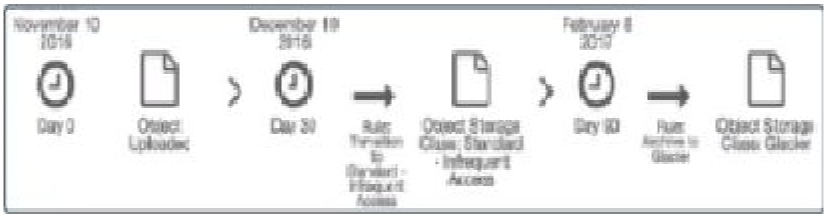
You can also see the various versions of the existing files.

Lifecycle Policies

- An object lifecycle policy is a **set of rules that automate** the migration of an object's storage class to a different storage class (or deletion), based on specified intervals
- By default, lifecycle policies are disabled on a bucket/objects
- These policies are customizable to meet your company's data retention policies
- Great for automating the management of object storage and to be more cost efficient
- Can be used with versioning to create a great archiving and backup solutions in s3

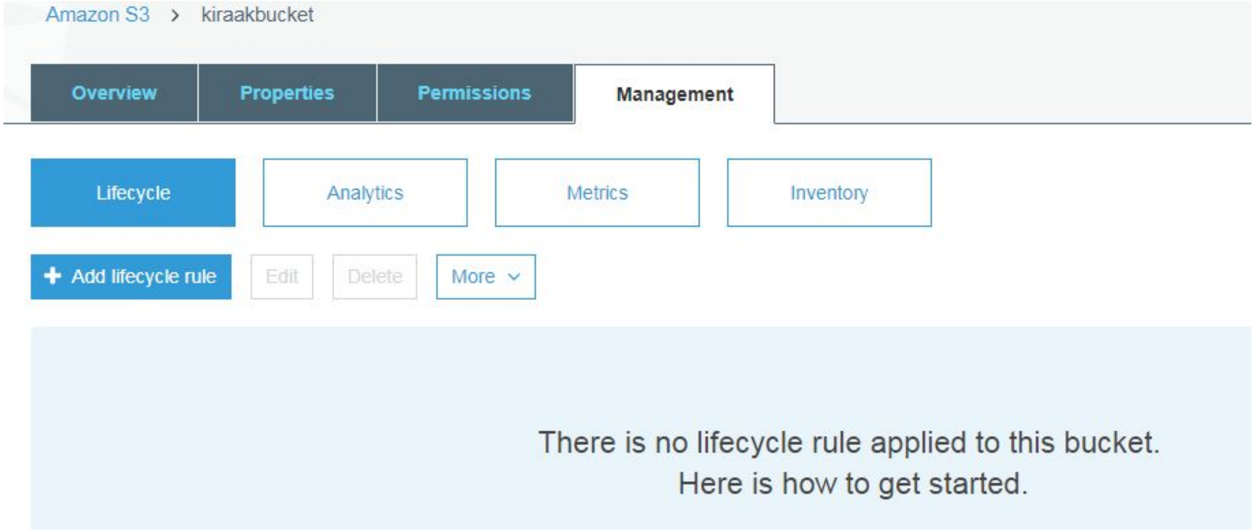
Scenario:

1) I have a work file that I am going to access every day for the next 30 days
2) After 30 days, I may only need to access that file once a week for the next 60 days
3) After which (90 days total) I will probably never access the file again but want to keep it just in case.

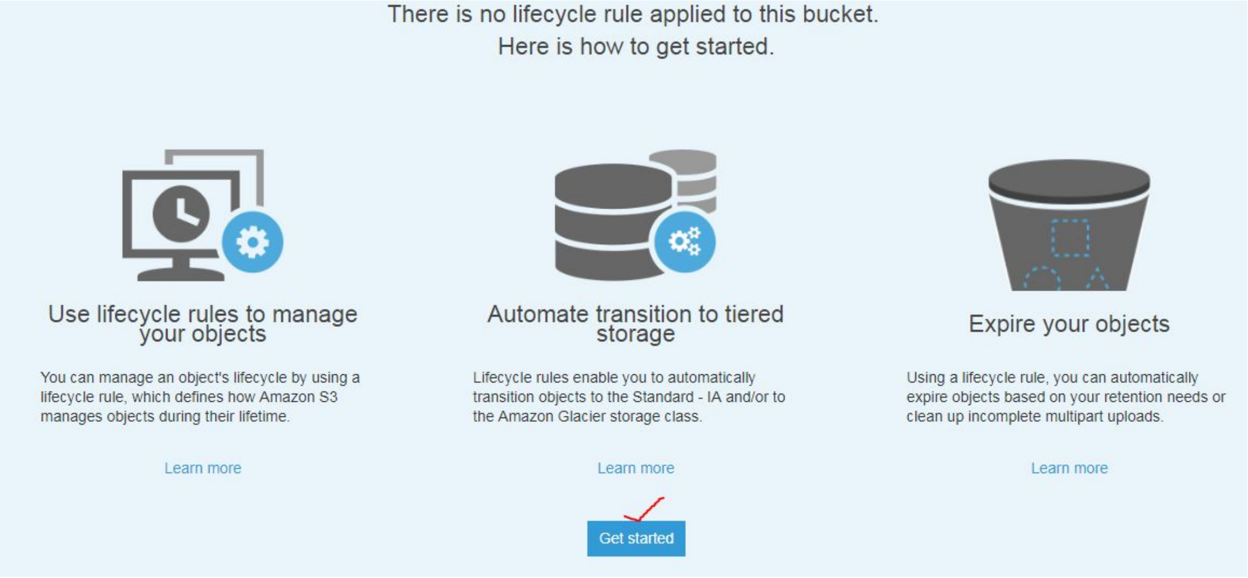

November 10 2016
Day 0
Object Uploaded
December 10 2016
Day 30
Rule: Transition to Standard - Infrequent Access
Object Storage Class: Standard - Infrequent Access
February 8 2017
Day 90
Rule: Archive to Glacier
Object Storage Class: Glacier

LAB 6: Setup the Lifecycle policy to the bucket as per req

Go to bucket and then click on management tab, then click on lifecycle



There is no lifecycle rule applied to this bucket.
Here is how to get started.



There is no lifecycle rule applied to this bucket.
Here is how to get started.

Use lifecycle rules to manage your objects

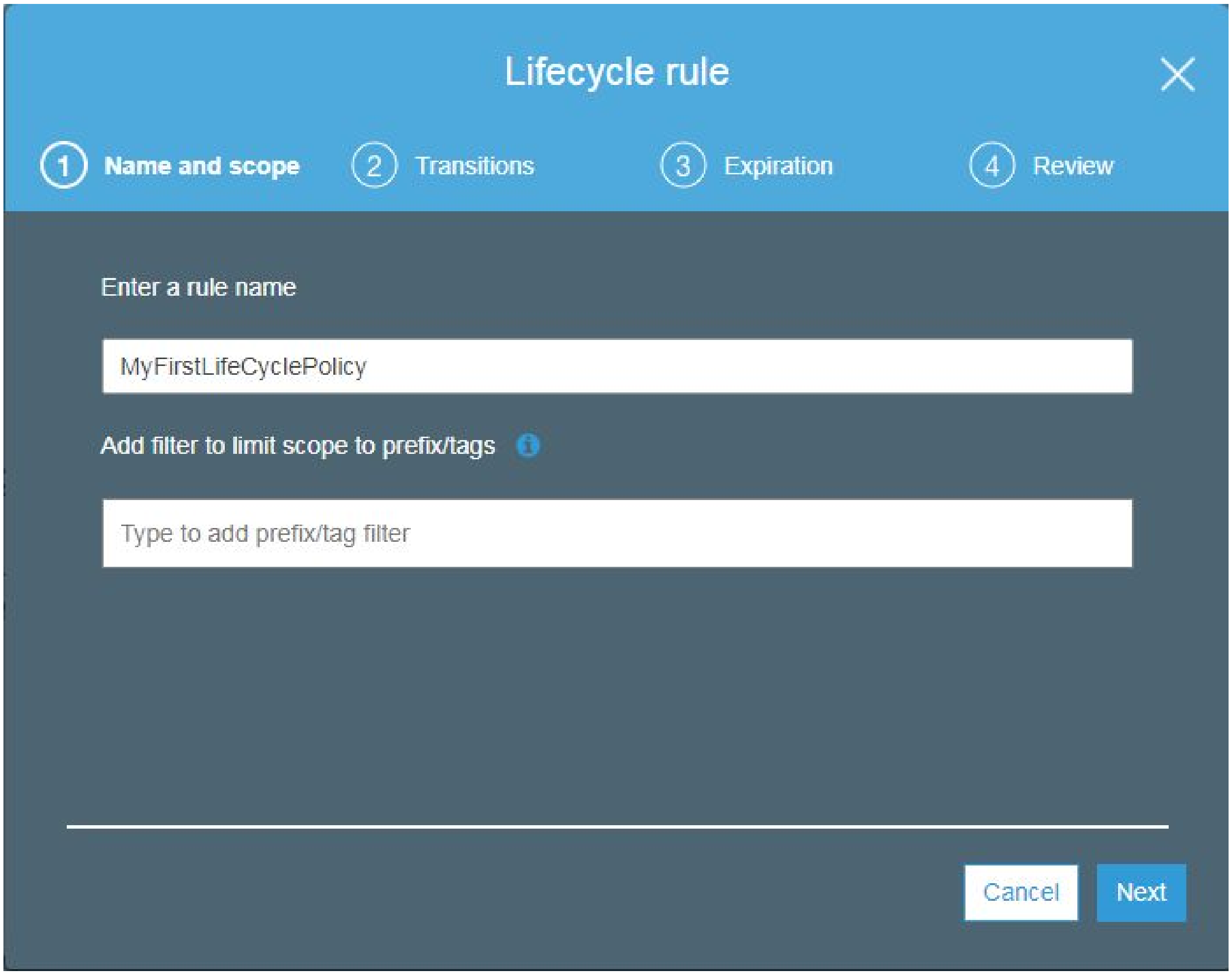
You can manage an object's lifecycle by using a lifecycle rule, which defines how Amazon S3 manages objects during their lifetime.

Automate transition to tiered storage

Lifecycle rules enable you to automatically transition objects to the Standard - IA and/or to the Amazon Glacier storage class.

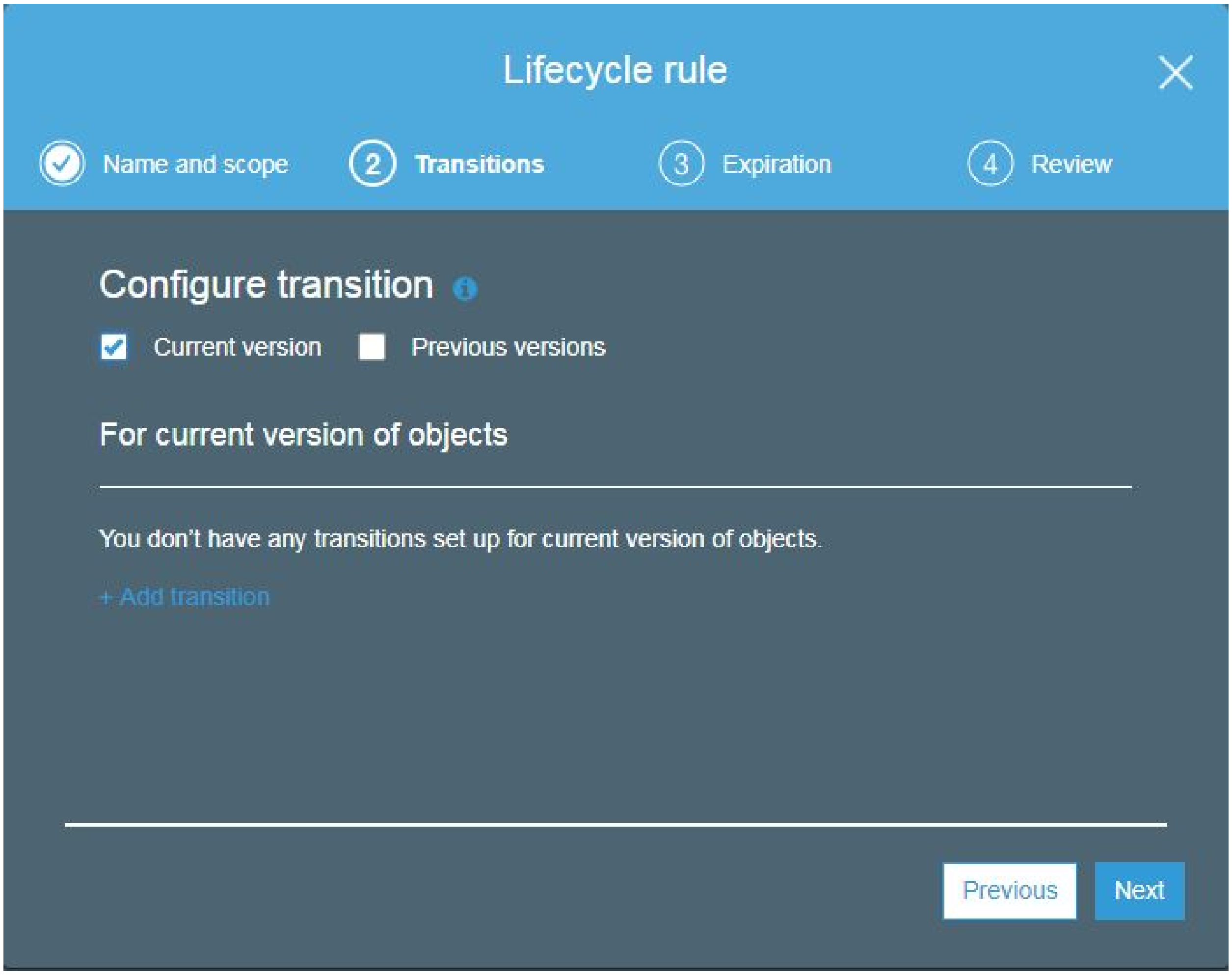
Expire your objects

Using a lifecycle rule, you can automatically expire objects based on your retention needs or clean up incomplete multipart uploads.



Enter a rule name

MyFirstLifeCyclePolicy

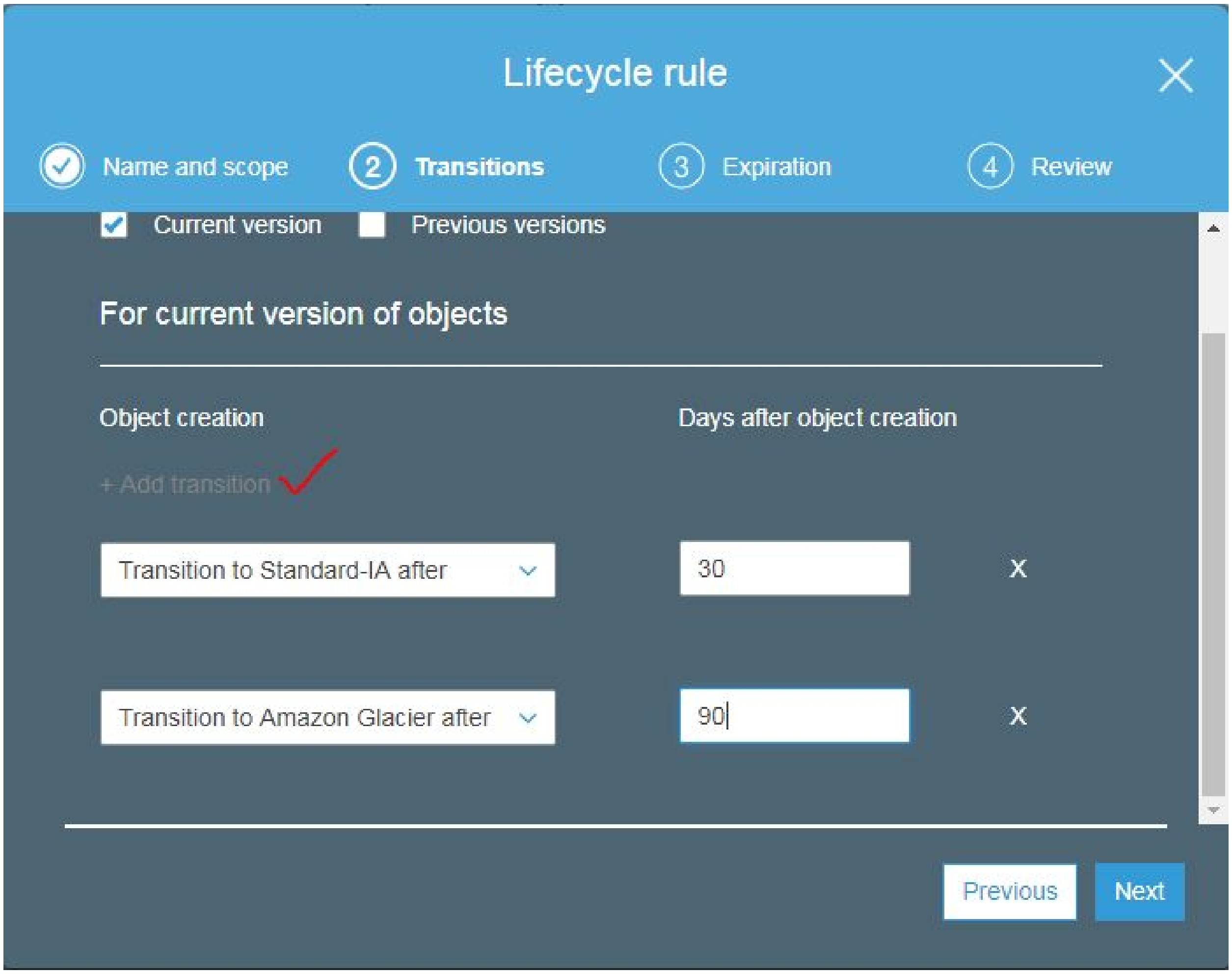


Configure transition

☑ Current version ☐ Previous versions

For current version of objects

You don’t have any transitions set up for current version of objects.

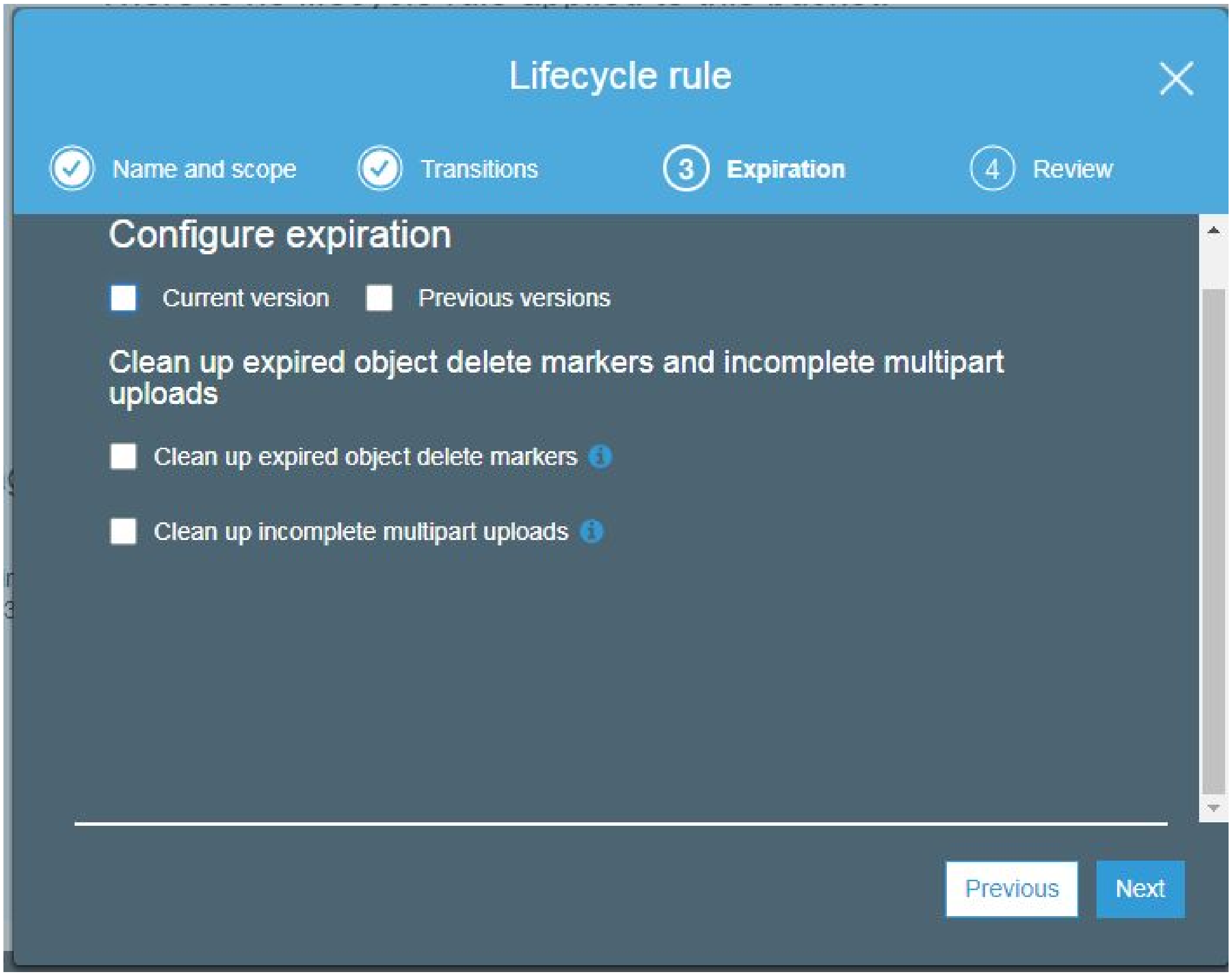


Lifecycle rule

☑ Current version □ Previous versions

For current version of objects

Object creation	Days after object creation
+ Add transition ✓	
Transition to Standard-IA after	30
Transition to Amazon Glacier after	90



Lifecycle rule

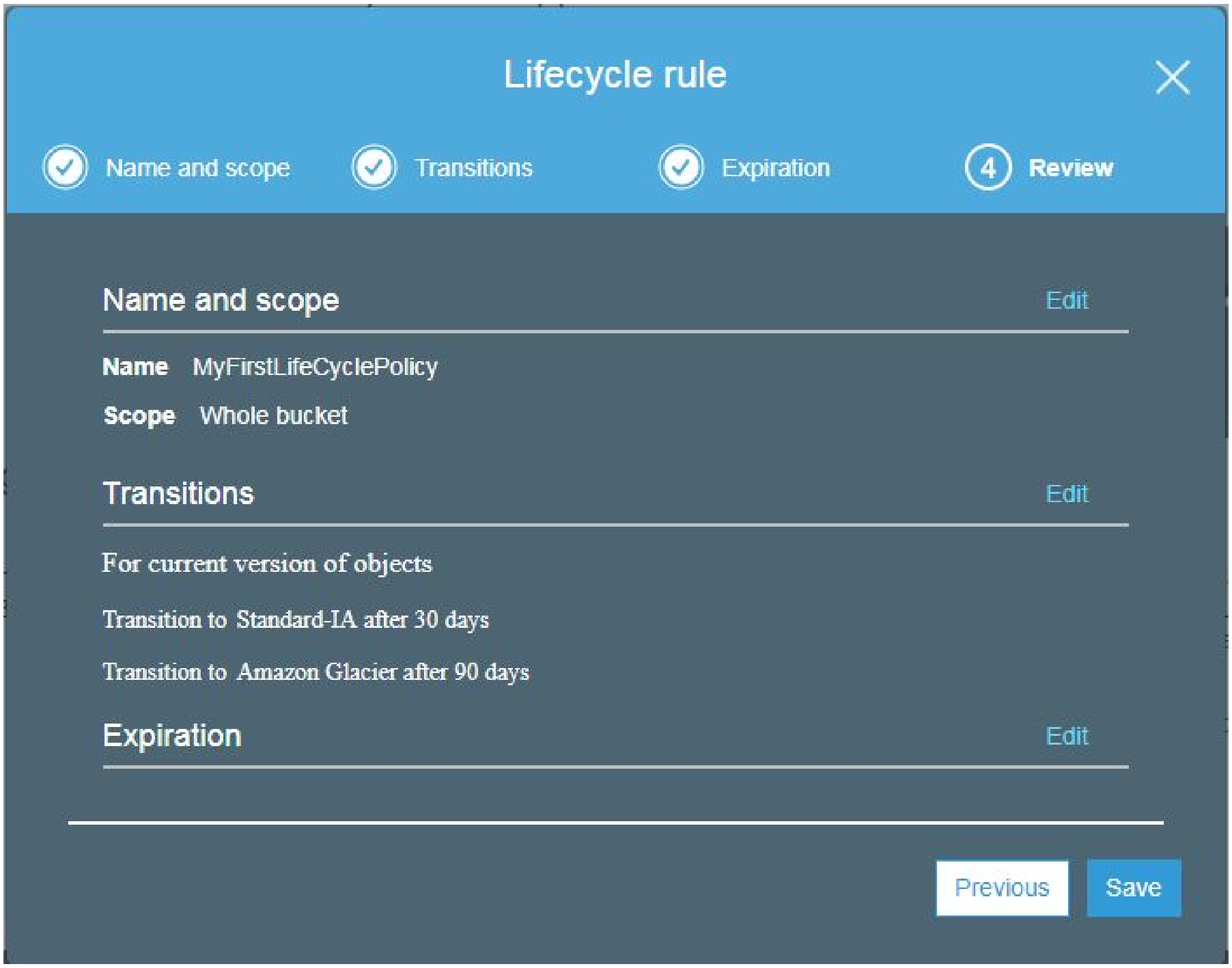
Configure expiration

☐ Current version ☐ Previous versions

Clean up expired object delete markers and incomplete multipart uploads

☐ Clean up expired object delete markers ℹ

☐ Clean up incomplete multipart uploads ℹ



Lifecycle rule

Name and scope

Name MyFirstLifeCyclePolicy

Scope Whole bucket

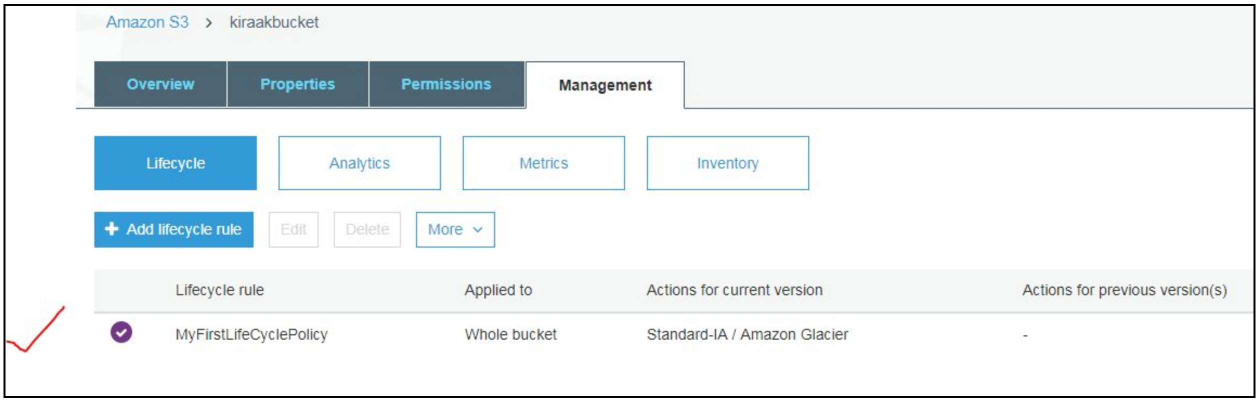
Transitions

For current version of objects

Transition to Standard-IA after 30 days

Transition to Amazon Glacier after 90 days

Expiration



	Lifecycle rule	Applied to	Actions for current version	Actions for previous version(s)
✓ ●	MyFirstLifeCyclePolicy	Whole bucket	Standard-IA / Amazon Glacier	-

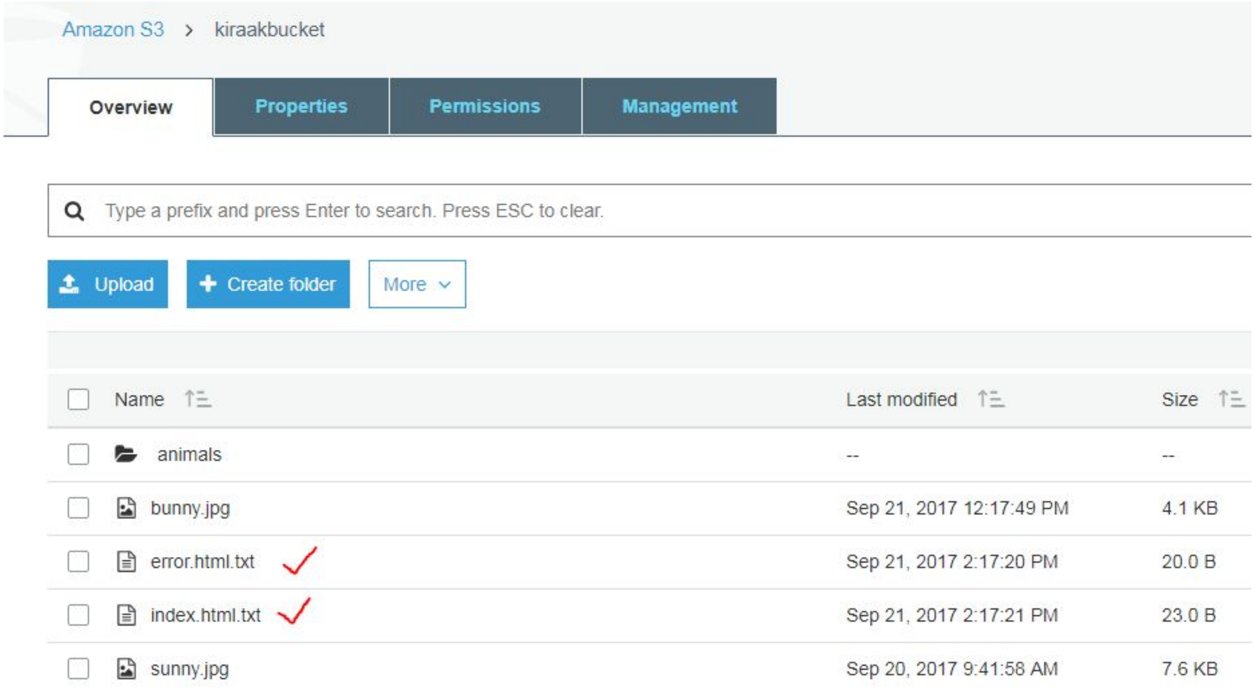
S3 Static Web Hosting

- AWS S3 provides an option for a low-cost, highly reliable web hosting service for a static websites (content that does not change frequently)
- When enabled, static web hosting will provide you with a unique endpoint (url) that you can point to any properly formatted file stored in an s3 bucket. Supported formats include:
- HTML
- CSS
- JavaScript

- AWS Route53 also helps to map human readable domain names to static web hosting buckets, which are ideal for DNS failover solutions

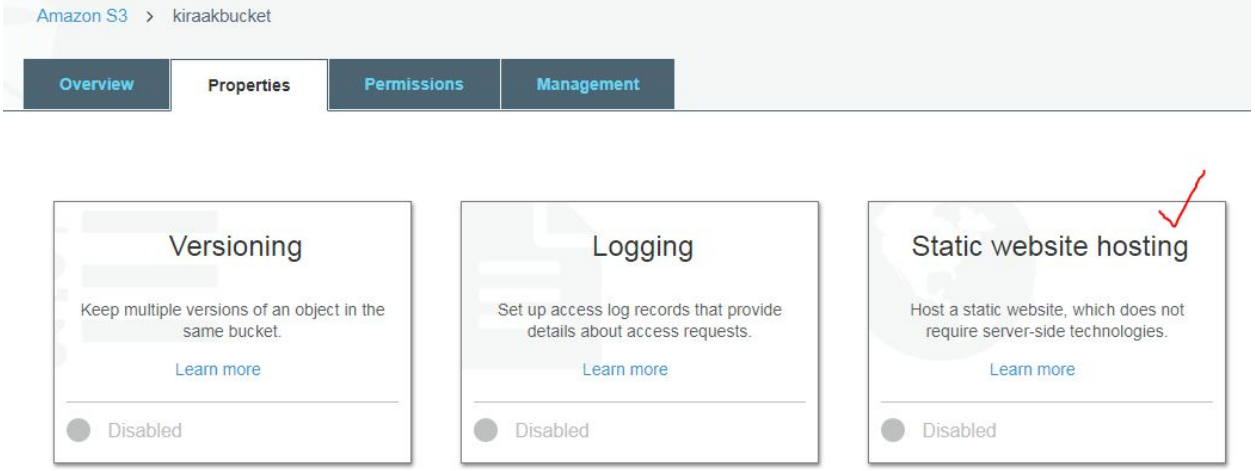
<u>LAB 7: Web Hosting the static site using S3</u>

Create 2 file by name index.html and error.html and upload into bucket



□	Name	Last modified	Size
□	animals	--	--
□	bunny.jpg	Sep 21, 2017 12:17:49 PM	4.1 KB
□	error.html.txt ✓	Sep 21, 2017 2:17:20 PM	20.0 B
□	index.html.txt ✓	Sep 21, 2017 2:17:21 PM	23.0 B
□	sunny.jpg	Sep 20, 2017 9:41:58 AM	7.6 KB

Now after that goto properties and click on static web hosting



Amazon S3 > kiraakbucket

Versioning

Keep multiple versions of an object in the same bucket.

Learn more

○ Disabled

Logging

Set up access log records that provide details about access requests.

Learn more

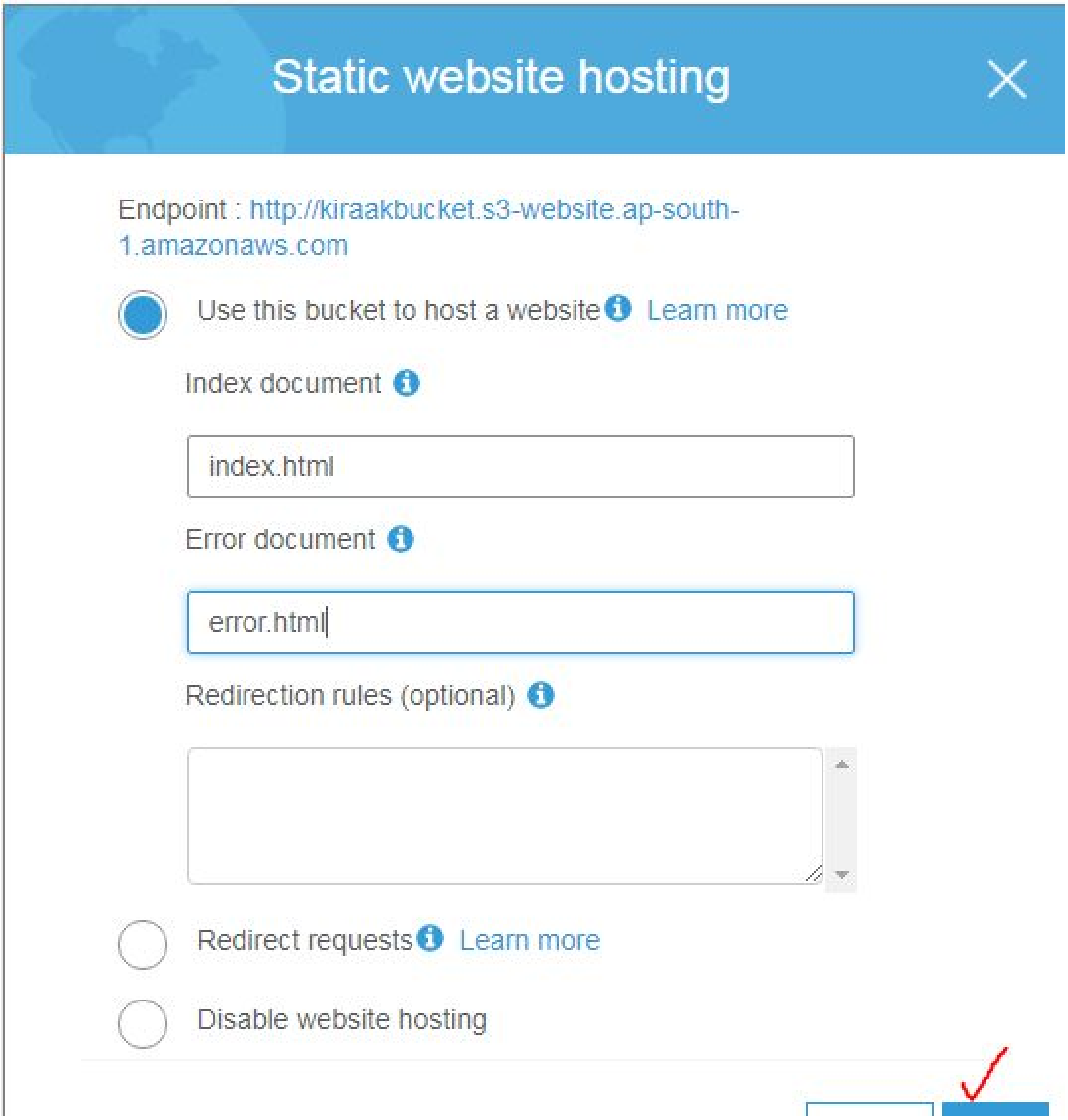
○ Disabled

Static website hosting

Host a static website, which does not require server-side technologies.

Learn more

○ Disabled



Static website hosting

Endpoint : http://kiraakbucket.s3-website.ap-south-1.amazonaws.com

● Use this bucket to host a website Learn more

Index document: index.html

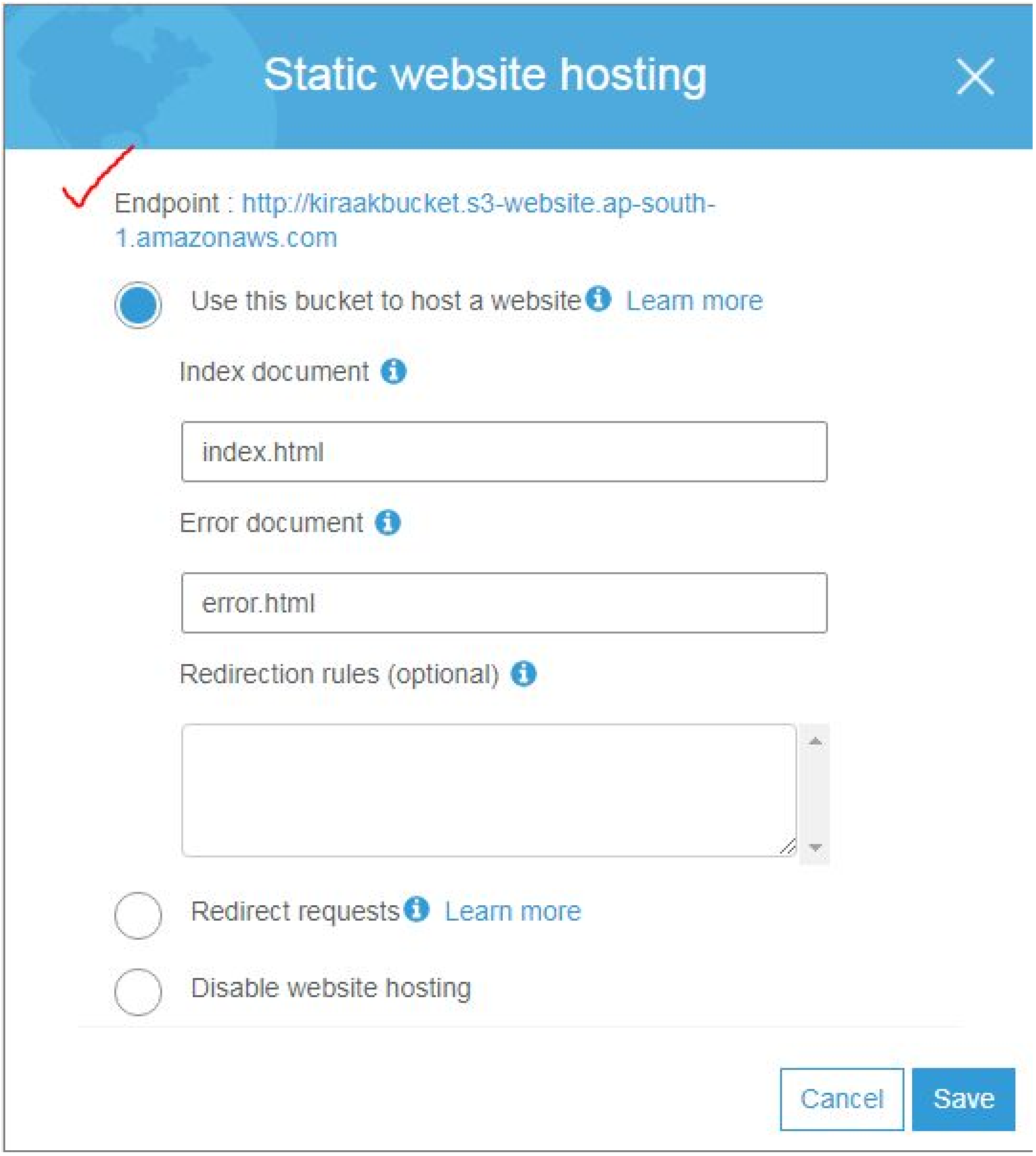
Error document: error.html

Redirection rules (optional):

○ Redirect requests Learn more

○ Disable website hosting

Now you can copy url by going inside static hosting tab



Static website hosting

Endpoint : http://kiraakbucket.s3-website.ap-south-1.amazonaws.com

● Use this bucket to host a website

Index document: index.html

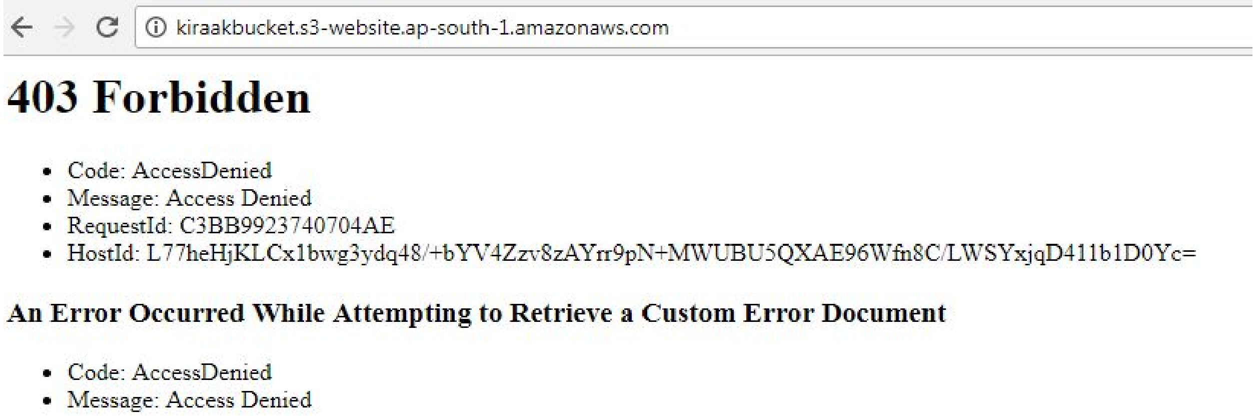
Error document: error.html

Redirection rules (optional)

○ Redirect requests

○ Disable website hosting

Paste that url in browser



403 Forbidden

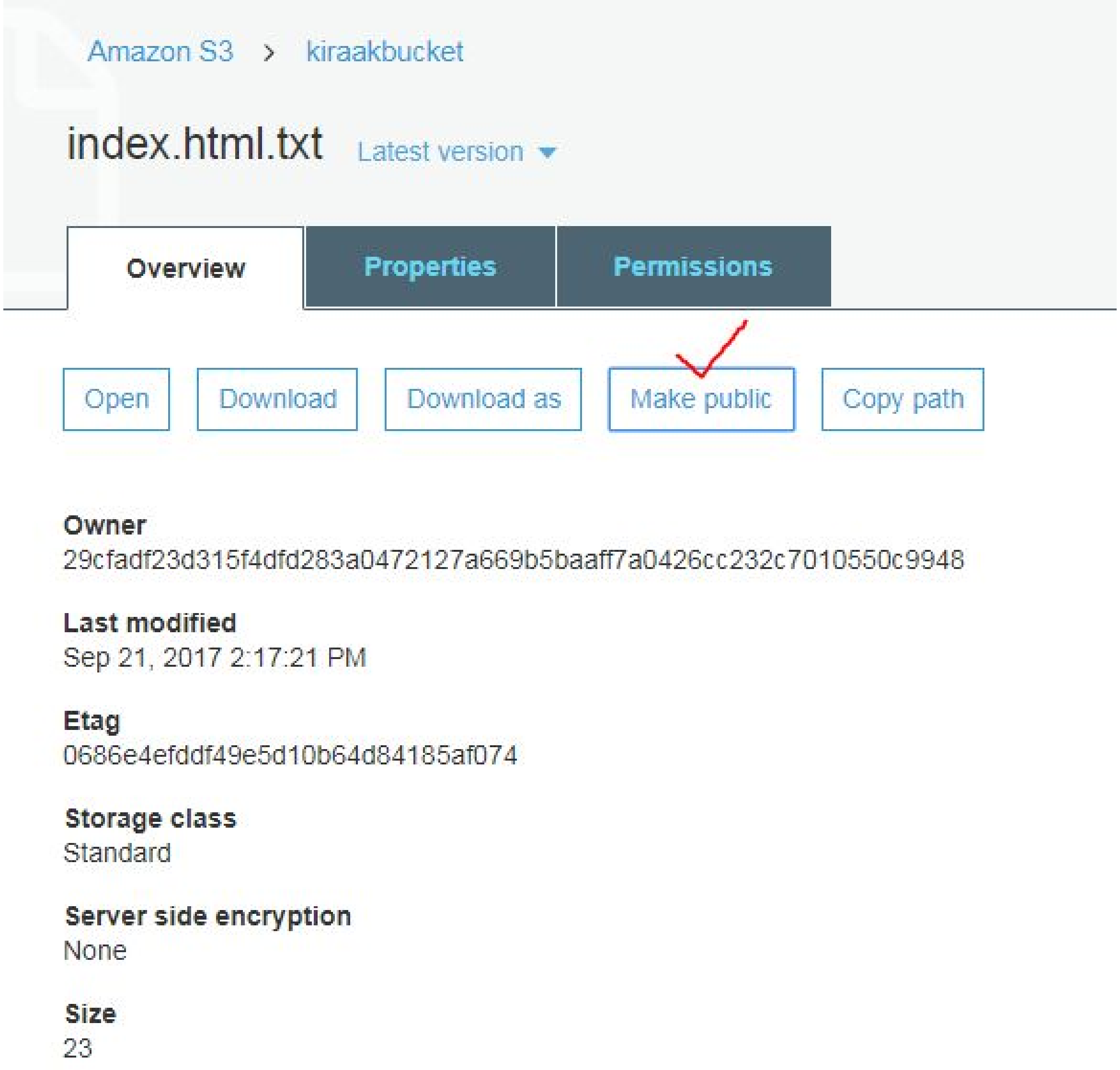
- Code: AccessDenied
- Message: Access Denied
- RequestId: C3BB9923740704AE
- HostId: L77heHjKLCx1bwg3ydq48/+bYV4Zzv8zAYrr9pN+MWUBU5QXAE96Wfn8C/LWSYxjqD411b1D0Yc=

An Error Occurred While Attempting to Retrieve a Custom Error Document

- Code: AccessDenied
- Message: Access Denied

You get the error as there is no proper permission to public

GO to the objects and give the public permission



Owner
29cfadf23d315f4dfd283a0472127a669b5baaff7a0426cc232c7010550c9948

Last modified
Sep 21, 2017 2:17:21 PM

Etag
0686e4efddf49e5d10b64d84185af074

Storage class
Standard

Server side encryption
None

Size
23

Same for error.html file too

CORS

- CORS is a method of allowing a web application located in one domain to access and use resources located in another domain
- This allows web application running Javascript or HTML5 to access resources in an S3 bucket without using a proxy server
- For AWS, this (commonly) means that a web applications hosted in one s3 bucket can access resources in another S3 bucket.

Storage Transit Services

There are 5 option available here

1) Single-operational Upload

- It is the traditional approach for upload where you upload the file in one part
- A single operation upload can upload a file upto 5GB in size, however any file over 100MB should use multipart upload

2) **Multipart Upload**

- It allows you to upload a single object as a set of parts
- Allows for uploading parts of a file concurrently
- Allows for stopping/resuming file uploads
- If transmission of any part fails, you can retransmit that part without affecting other parts.
- After all parts of your objects are uploaded, amazon S3 assembles these parts and creates an object
- Required for objects of 5GB and large, and **Highly suggested** for use when objects are 100MB and larger
- Can be used to upload a file up to 5TB in size

3) **AWS Import/Export**

- It gives the ability to take on-premise data and physically snail mail it to AWS (using a device that you own)
- AWS will import the data either to S3, EBS, or Glacier within one business day of the physical device arriving at AWS

Benefits:

➢ Off-site backup policy
➢ Quickly migrate LARGE amounts of data to the cloud (upto 16TB per job)
➢ Disaster Recovery (AWS will even take s3 data and ship it back to you)

4) **Snowball**

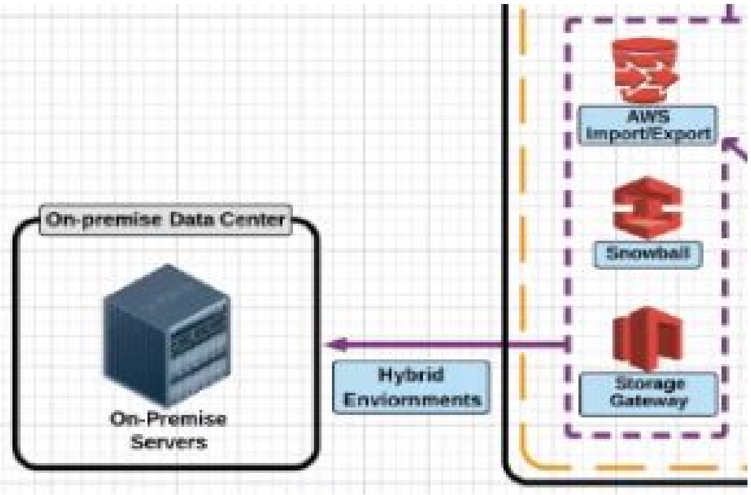
- Snowball is a petabyte-scale data transport solution
- It uses an AWS provided secure transfer appliance
- Quickly move large amounts of data into and out of the AWS cloud

5) **Storage Gateway**

- Connects local data center software appliances to cloud based storage such as Amazon S3.

- Gateway-cached volumes
  → create storage volumes and mount them as iscsi devices on the on-premise servers
  → the gateway will store the data written to this volume in amazon s3 and will cache frequently access data on-premise in the storage device

- Gateway-stored volumes
  → Stores all the data locally (on-premise) in storage volumes
  → Gateway will periodically take snapshots of the data as incremental backups and stores them on S3


AWS Import/Export
Snowball
On-premise Data Center
Hybrid Enviornments
Storage Gateway
On-Premise Servers