

SVKM'S NMIM'S Nilkamal School of Mathematics, Applied Statistics & Analytics Master of Science (Data Science)

Practical-4 To configure Amazon Simple Storage Service (Amazon S3)

Writeup:

STORAGE AS A SERVICE S3:

Storage as a Service (STaaS) is a cloud computing model that enables businesses and individuals to store, manage, and access data remotely over the internet. Instead of maintaining physical storage infrastructure on-premises, users can leverage third-party cloud service providers to store their data securely in off-site data centers.

Key characteristics of Storage as a Service include:

Scalability: STaaS offers scalable storage solutions, allowing users to increase or decrease storage capacity based on their needs without significant upfront investments in hardware or infrastructure.

Cost-Effectiveness: Users typically pay for STaaS on a subscription or pay-as-you-go basis, which can be more cost-effective than purchasing and maintaining on-premises storage infrastructure. This model eliminates the need for upfront capital expenditure and reduces operational costs associated with hardware maintenance and upgrades.

Accessibility: Data stored in the cloud can be accessed from anywhere with an internet connection, providing flexibility and enabling remote collaboration among users in different locations.

Reliability and Redundancy: Reputable STaaS providers often offer robust data redundancy and backup mechanisms to ensure data durability and availability. Data replication across multiple geographically dispersed data centers helps mitigate the risk of data loss due to hardware failures, disasters, or other unforeseen events.

Security: STaaS providers employ various security measures, including encryption, access controls, and compliance certifications, to protect sensitive data from unauthorized access, data breaches, and cyber threats.

Management and Maintenance: STaaS providers handle the management, maintenance, and upgrades of the underlying storage infrastructure, freeing users from the administrative burden of managing storage hardware and software.

Integration: Many STaaS solutions offer seamless integration with other cloud services, applications, and workflows, enabling users to leverage their existing tools and environments.

Overall, Storage as a Service offers a flexible, cost-effective, and reliable solution for storing and managing data in the cloud, making it an attractive option for businesses of all sizes looking to streamline their storage infrastructure and optimize resource utilization.

USECASES:

Static Content Hosting: Amazon S3 can host static website content, such as HTML, CSS, JavaScript files, images, and videos. Users can upload their website assets directly to an S3 bucket, making them accessible via unique URLs.

Cost-Effective: Hosting a static website on Amazon S3 is cost-effective, as users only pay for the storage used and data transfer out of the S3 bucket. There are no charges for requests made to the website, making it suitable for low-traffic websites or applications.

Scalability: S3 automatically scales to accommodate increasing traffic and storage requirements. Users do not need to worry about provisioning or managing servers; Amazon S3 handles the scalability aspects transparently.

High Availability and Durability: Amazon S3 provides high availability and durability for hosted content. S3 stores data across multiple data centers within a region, ensuring redundancy and fault tolerance. This means that static websites hosted on S3 benefit from reliable and consistent performance.

Content Delivery: Users can configure Amazon CloudFront, AWS's content delivery network (CDN), to distribute website content globally with low latency and high transfer speeds. By integrating S3 with CloudFront, users can deliver website assets efficiently to users worldwide.

Security: Amazon S3 offers various security features, including encryption at rest and in transit, access control policies, and integration with AWS Identity and Access Management (IAM). Users can secure their static websites and restrict access to specific users or groups.

Versioning and Lifecycle Policies: S3 supports versioning, allowing users to retain multiple versions of objects and recover them if needed. Additionally, users can define lifecycle policies to automatically transition or expire objects based on predefined rules, helping manage storage costs effectively.

STEPS FOR S3:

To Configure S3 with following task:

Sign Up for Amazon S3

Create a Bucket

Add an Object to a Bucket

Add an folder to Bucket

View an Object

Move an Object

Delete an Object and Bucket

To empty a bucket

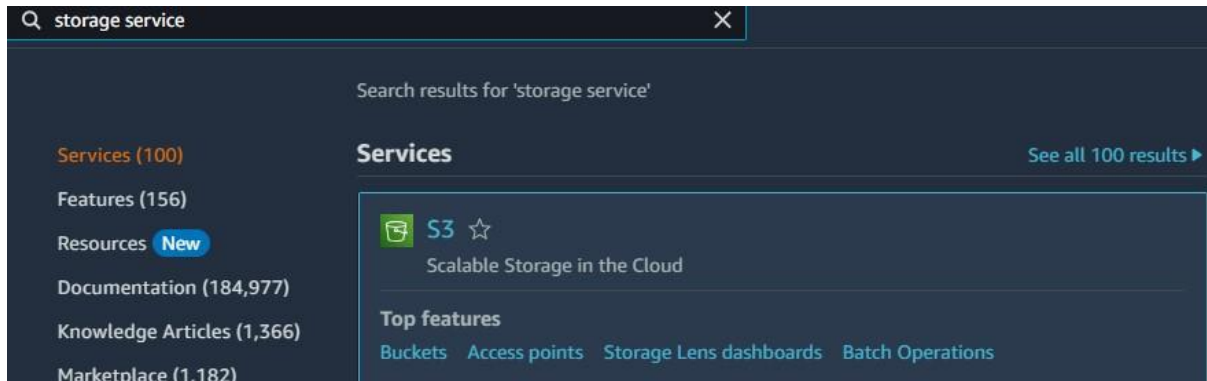
To delete a bucket

Hosting a Static Website on Amazon S3

AWS user to control S3

Step 1 : To create S3 bucket for storing objects that is files and folders.

Select Storage service and click on S3



Do General configuration :

A screenshot of the 'Create bucket' configuration page in the AWS Management Console. The page title is 'Create bucket' with an 'Info' link. Below the title, it says 'Buckets are containers for data stored in S3.' The 'General configuration' section is expanded. It contains a dropdown for 'AWS Region' set to 'Asia Pacific (Mumbai) ap-south-1'. Below that is a text input for 'Bucket name' with the value 'upasanabucket' and an 'Info' link. A note states: 'Bucket name must be unique within the global namespace and follow the bucket naming rules. See rules for bucket naming'. There is a section for 'Copy settings from existing bucket - optional' with a note that 'Only the bucket settings in the following configuration are copied.' and a 'Choose bucket' button. At the bottom, the format 'Format: s3://bucket/prefix' is shown.

Rest keep same as follows:

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.

- Object Ownership
Bucket owner enforced

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 this bucket and its 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 this bucket or objects within, you can customize the individual settings below to suit your specific storage use cases. [Learn more](#)

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

- ### Create bucket

Buckets are containers for data stored in S3.

 Find buckets by name

	Name	AWS Region	Access	Creation date
<input type="radio"/>	elasticbeanstalk-ap-south-1-143861992431	Asia Pacific (Mumbai) ap-south-1	Objects can be public	February 26, 2024, 21:53:56 (UTC+05:30)
<input type="radio"/>	upasanabucket1	Asia Pacific (Mumbai) ap-south-1	Bucket and objects not public	February 29, 2024, 08:00:51 (UTC+05:30)

Select the upasanabucket1

Objects	Properties	Permissions	Metrics	Management	Access Points
---------	------------	-------------	---------	------------	---------------

Objects (0) [Info](#)

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](#)

Find objects by prefix

Name	Type	Last modified	Size	Storage class
------	------	---------------	------	---------------

No objects
You don't have any objects in this bucket.

 Upload

[Amazon S3](#) > [Buckets](#) > [upasanabucket1](#) > Upload

Upload

[Info](#)

Add the files and folders you want to upload to S3. To upload a file larger than 160GB, use the AWS CLI, AWS SDK or Amazon S3 REST API. [Learn more](#)

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

Files and folders (0)

All files and folders in this table will be uploaded.

< 1 >

<input type="checkbox"/>	Name	Folder
No files or folders		
You have not chosen any files or folders to upload.		

Destination

[Info](#)

Destination

s3://upasanabucket1

► 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.

UploadInfo

Add the files and folders you want to upload to S3. To upload a file larger than 160GB, use the AWS CLI, AWS SDK or Amazon S3 REST API. [Learn more](#)

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

Files and folders (1 Total, 9.4 KB)

All files and folders in this table will be uploaded.

Remove

Add files

Add folder

Find by name

< 1 >

<input checked="" type="checkbox"/>	Name	Folder
<input checked="" type="checkbox"/>	Hibiscus.jpg	-

DestinationInfo

Destination

s3://upasanabucket1

► 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.

Cancel

Upload

Upload succeeded

View details below.

Amazon S3 > Buckets > upasanabucket1 > Hibiscus.jpg

Hibiscus.jpg info

Copy S3 URI

Download

Open

Object actions

PropertiesPermissionsVersions

Object overview

Owner

ace1b026c0e865430af265fc9a28725dfb35ca40fdc6640fd87e8a63e0885ba

AWS Region

Asia Pacific (Mumbai) ap-south-1

Last modified

February 29, 2024, 08:07:59 (UTC+05:30)

Size

9.4 KB

Type

jpg

Key

Hibiscus.jpg

S3 URI

s3://upasanabucket1/Hibiscus.jpg

Amazon Resource Name (ARN)

arn:aws:s3:::upasanabucket1/Hibiscus.jpg

Entity tag (Etag)

386cef272ee0814a88300dbd6bfe9d64

Object URL

https://upasanabucket1.s3.ap-south-1.amazonaws.com/Hibiscus.jpg

Click on Open button to see

To see the url

PropertiesPermissionsVersions

Bucket properties

Bucket Versioning

When enabled, multiple variants of an object can be stored in the bucket to easily recover from unintended user actions and application failures.

Disabled

Bucket “upasanabucket1” doesn’t have Bucket Versioning enabled

We recommend that you enable Bucket Versioning to help protect against unintentionally overwriting or deleting objects. [Learn more](#)

Enable Bucket Versioning

Enable Bucket Versioning

Go to permissions

Access control list (ACL)

Grant basic read/write permissions to AWS accounts. [Learn more](#)

This bucket has the bucket owner enforced setting applied for Object Ownership

When bucket owner enforced is applied, use bucket policies to control access. [Learn more](#)

Edit Object Ownership [Info](#)

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. Access to this bucket and its objects is specified using only policies.

☒ **ACLs enabled**

Objects in this bucket can be owned by other AWS accounts. Access to this bucket and its objects can be specified using ACLs.

 We recommend disabling ACLs, unless you need to control access for each object individually or to have the object writer own the data they upload. Using a bucket policy instead of ACLs to share data with users outside of your account simplifies permissions management and auditing.



Enabling ACLs turns off the bucket owner enforced setting for Object Ownership

Once the bucket owner enforced setting is turned off, access control lists (ACLs) and their associated permissions are restored. Access to objects that you do not own will be based on ACLs and not the bucket policy.

☒ I acknowledge that ACLs will be restored.


Object Ownership

☒ **Bucket owner preferred**

If new objects written to this bucket specify the bucket-owner-full-control canned ACL, they are owned by the bucket owner. Otherwise, they are owned by the object writer.

☐ **Object writer**

The object writer remains the object owner.

 If you want to enforce object ownership for new objects only, your bucket policy must specify that the bucket-owner-full-control canned ACL is required for object uploads. [Learn more](#)

Cancel

Save changes

Hibiscus.jpg [Info](#)





[Copy S3 URI](#) [Download](#) [Open](#) [Object actions](#)

Properties **Permissions** Versions

Access control list (ACL)

Grant basic read/write permissions to AWS accounts. [Learn more](#)

Edit

Grantee	Object	Object ACL
Object owner (your AWS account) Canonical ID:  ace1b026c0e865430af2651cf9a28725dfb35ca40fd6640fd87e8ab5e0883ba	Read	Read, Write
Everyone (public access) Group:  http://acs.amazonaws.com/groups/global/AllUsers	 Read	 Read
Authenticated users group (anyone with an AWS account) Group:  http://acs.amazonaws.com/groups/global/AuthenticatedUsers	 Read	 Read



url : <https://upasanabucket1.s3.ap-south-1.amazonaws.com/Hibiscus.jpg>

Static website

Amazon S3 > Buckets > upasanabucket1 > web.html

web.html info

Copy S3 URI Download Open Object actions

Properties Permissions Versions

Object overview

Owner ace1b026c0e865430af265f9a28725dfb35ca40fdc6640fd87e8a63e0885ba	S3 URI s3://upasanabucket1/web.html
AWS Region Asia Pacific (Mumbai) ap-south-1	Amazon Resource Name (ARN) arn:aws:s3::upasanabucket1/web.html
Last modified February 29, 2024, 08:30:46 (UTC+05:30)	Entity tag (Etag) e003336f0a9229864e5f3d4e197293a2
Size 616.0 B	Object URL https://upasanabucket1.s3.ap-south-1.amazonaws.com/web.html
Type html	
Key web.html	



url : <https://upasanabucket1.s3.ap-south-1.amazonaws.com/web.html>

Go to google , bootstrap search buttons , and copy paste file in html

```
<html>
<div class="btn-group" role="group" aria-label="Basic radio toggle button group">
  <input type="radio" class="btn-check" name="btnradio" id="btnradio1" autocomplete="off"
checked>
  <label class="btn btn-outline-primary" for="btnradio1">Radio 1</label>

  <input type="radio" class="btn-check" name="btnradio" id="btnradio2" autocomplete="off">
  <label class="btn btn-outline-primary" for="btnradio2">Radio 2</label>

  <input type="radio" class="btn-check" name="btnradio" id="btnradio3" autocomplete="off">
  <label class="btn btn-outline-primary" for="btnradio3">Radio 3</label> </div>
</html>
```

