SIMPLE STORAGE SERVICE BASICS

SIMPLE STORAGE SERVICE (S3)

Simple Storage Service (S3)

What is S3?
A global object storage platform that can be used to store objects in the form of text files, photos, audio, movies, large binaries, or other object types.

Why S3?

* Fully managed object-based storage service
* Most used storage service
* Useful for a variety of use cases
* Can integrate with many AWS services
* Highly available and durable
* Very cost-effective
* Widely and easily accessible
* Unlimited storage capabilities
* Highly scalable (more than the on-premise solutions)
* The smallest file size supported - 0 bytes
* The largest file size supported - 5 TB
* Objects are created in specific regions that can be chosen at the time of the creation
* S3 replicates the data in various AZs present in the region of choice
* Offers storage for any type of data including, but not limited to video, audio, and text

Availability V/S Durability

The ability of a system to operate without failing for as long as the application is deployed and allows it to continue functioning, even when some of its components fail.	The probability of maintaining the customer data without it being lost by corruption, data degradation, etc.

* Objects stores in S3 have durability of 99.999999999%
* Objects stores in S3 have availability of 99.99%

By: Dasika Madhu Nimeshika

S3 BUCKET BASICS

Simple Storage Service (S3) - Buckets

* A container for storing objects
* Bucket names must be globally unique and DNS-compliant - the namespace is shared by all AWS accounts
* Upload data into buckets or folders nested within the bucket
* The default bucket limit- 100 per account - increase the number of buckets to 1000 by raising a support ticket
* For lower latency and cost optimization, create buckets in Regions geographically close to your location
* Region cannot be changed after bucket creation
* Bucket ownership is non-transferable
* Folders can be nested within buckets but buckets within buckets cannot be created
* Buckets containing more than 100,000 objects cannot be deleted
* Buckets with versioning enabled cannot be deleted via the CLI
* Buckets must be empty before they can be deleted

Bucket Configurations

Naming Rules

* Names must be between 3 and 63 characters long.
* Names can consist only of lowercase letters, numbers, dots, and hyphens.
* Names must begin and end with a letter or number.
* Names must not be formatted as an IP address; for example, 12.11.5.4.
* Names must be unique within a partition (grouping of Regions).
* Buckets used with Amazon S3 Transfer Acceleration can't have dots in their names

Recommended

docexamplebucket1
log-delivery-march-2020
my-hosted-content

Website Hosting

docexamplewebsite.com
www.docexamplewebsite.com
my.example.s3.bucket

Invalid Names

doc_example_bucket
DocExampleBucket
doc-example-bucket-

By Dasika Madhu Nimeshika

Use Cases

* Data storage
* Data archiving
* Application hosting for deployment, installation and management of web apps
* Data backup
* Disaster recovery (DR)
* To run big data analytics tools on stored data
* Data lakes
* Mobile applications
* Internet of things (IoT) devices
* Media hosting for images, videos and music files
* Website hosting

Bucket Configurations — Bucket Sub-Resources & Properties

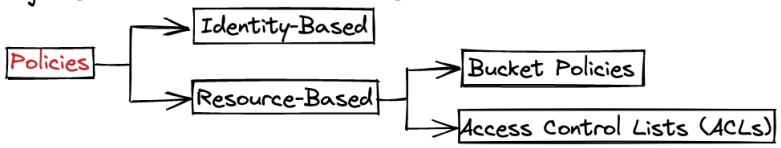
Sub-Resource	Description
CORS (cross-origin resource sharing)	Allows web applications loaded in one domain to interact with resources in a different domain
event notification	The Amazon S3 Event Notifications send notifications when certain events happen in the bucket
lifecycle	Create lifecycle rules to delete objects in the bucket after a specific amount of time; archive objects one year after the creation, or delete an object 10 years after the creation.
location	Specify the AWS Region to create the bucket and AWS provides an API for you to retrieve this information
Server Access Logging	Disabled by default. Configure a bucket for static website hosting by enabling it.
object locking	Disabled by default. Lock objects in that bucket using retention periods, legal holds, or both
policy and ACL (access control list)	Private by default. Used to grant and manage bucket-level permissions.
replication	Automatic, asynchronous copying of objects across buckets in different or the same AWSRegions
requestPayment	After enabling Requester Pays, the bucket owner pays for downloads from the bucket.
tagging	Used to store and manage tags on a bucket. Can generate a cost allocation report with usage and costs aggregated by the tags.
transfer acceleration	Uses AWS CloudFront to enable fast, easy, and secure transfers of files over long distances between clients and an S3 bucket.
versioning	Stores multiple versions of the same object and helps in recovering accidental overwrites and deletes.
website	Disabled by default. Configure a bucket for static website hosting by enabling it.
encryption	Disabled by default. Automatically encrypt new objects stored in this bucket.
AWS CloudTrail data events	Configure CloudTrail data events to log Amazon S3 object-level API operations in the CloudTrail console.

By Dasika Madhu Nimeshika

Bucket Configurations

Permissions & Access

* Appropriate permissions are required to be able to create, update, delete and interact with buckets
* Permissions added to the bucket apply to the bucket as well as the objects in it
* Access to buckets is given through identity-based or resource-based policies
* Block public access settings feature provides settings for access points, buckets, and accounts to allow or deny public access to bucket resources
* Add 'Object Ownership' for objects uploaded to the buckets from other AWS accounts - either the object write will be the owner or the bucket owner is the owner (if the object is uploaded using the `bucket-owner-full-control` canned ACL)



* If a request isn't explicitly allowed, it's implicitly (default) is denied
* If a request is explicitly denied, it overrides everything else
* If a request is explicitly allowed, it's allowed unless denied by an explicit deny
* Only attached policies have any impact
* When evaluating policies, all applicable policies are merged: All identity (user, group, role) and any resource policies

explicit DENY > explicit ALLOW > default DENY

Principal of Least-Privilege

Policies include:
* Resources – buckets and objects
* Actions – set of operations
* Effect – can be either allow or deny -> need to explicitly grant allow to a resource
* Principal – the account, service or user who is allowed access to the actions and resources in the statement

By Dasika Madhu Nimeshika

Bucket Configurations — Resource-Based Policies

* Resource-based policies are associated with the resource than the identity
* Can either be Bucket Policies or Access Control Lists (ACLs)

Bucket Configurations — Bucket Policies

* Provides centralized access control to buckets and objects based on a variety of conditions, including S3 operations, requesters, resources, and aspects of the request
* Written in JSON
* Limited to 20 KB in size
* Directly attached to a bucket to grant and restrict access control
* By default, when a bucket is created, no bucket policy exists
* Impose and set access controls within a specific bucket
* Has no effect when the policy is not attached to any bucket
* Provides added granularity to the buckets the policy is attached to (specific permissions for users in an AWS account or for a specific time range or only when the request is coming from a specific time range)

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid":"GrantAnonymousReadPermissions",
            "Effect":"Allow",
            "Principal": "*",
            "Action":["s3:GetObject"],
            "Resource":["arn:aws:s3:::awsexamplebucket1/*"]
        }
    ]
}
```

Bucket Configurations — Access Control Lists (ACLs)

* A list of grants identifying grantees and permissions granted
* ACLs use an S3-specific XML schema, not JSON format
* ACLs are far less granular than bucket policies and IAM/ policies and they can be applied at the bucket level or the object level
* It's not possible to implicitly deny access using ACLs
* ACL Permissions are based on the Grantee: The Bucket Owner, Everyone (public access), Authenticated Users, and S3 Log delivery group
* ACL permissions for buckets include List, Write, Bucket ACL Read, bucket ACL Write
* There is no WRITE permission when working with object ACLs
* ACL permissions for objects include: Read Object, Read object permissions, Write object permissions

```
<?xml version="1.0" encoding="UTF-8"?>
<AccessControlPolicy xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Owner>
    <ID>*** Owner-Canonical-User-ID ***</ID>
    <DisplayName>owner-display-name</DisplayName>
  </Owner>
  <AccessControlList>
    <Grant>
      <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
               xsi:type="Canonical User">
        <ID>*** Owner-Canonical-User-ID ***</ID>
        <DisplayName>display-name</DisplayName>
      </Grantee>
      <Permission>FULL_CONTROL</Permission>
    </Grant>
  </AccessControlList>
</AccessControlPolicy>
```

* Different permissions can be applied depending on where ACL is applied (bucket level or object level)
* It is not possible to implicitly deny access using ACLs or to implement conditional elements, like with identity-based access
* Levels of permissions that can be applied to the Bucket
*. Permissions are given to the grantee groups for the bucket, which are either List or Write
* Permissions are given to enable the grantees access to either Read or Write against the Bucket ACL
* ACLs include:

1. Bucket Owner: This will be your AWS account and will have full control over all objects and the bucket itself
2. Everyone (public access): Anyone can access using the permissions applied, providing the the object had been made public. These requests can be signed (authenticated) or unsigned (unauthenticated).
3. Authenticated Users: This option will allow IAM/ users from any AWS account to access the object, via signed requests (authenticated).
4. S3 Log delivery group: This is a group used to deliver log files when the server access logs have been configured and the bucket is used to store and WRITE log files to

Bucket Configurations — Identity-Based Policies

* Identity-based policies are attached to the IAM identity requiring access
* They can either be in-line, customer-managed or AWS-managed policies
* In-Line Policies - A policy that's embedded in an IAM identity
* AWS-Managed Policies - A standalone policy that is created and administered by AWS
* Customer-Managed Policies - A standalone policy to administer in a personal AWS account
* Associated to users, groups or roles
* Defines the resources in the policy (bucket name/object name)
* Specify conditions to limit permissions additionally
* Centrally manage access control methods all in one service
* Use a lesser number of policies across multiple buckets from IAM, rather than creating one bucket policy per bucket
* Can control access for more than one service at a time
* Can be a maximum of 2 kb in size for users, 5 kb for groups and 10 kb for rules

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AssignUserActions",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:GetObject",
                "s3:ListBucket",
                "s3:DeleteObject",
                "s3:GetBucketLocation"
            ],
            "Resource": [
                "arn:aws:s3 ::: awsexamplebucket1/*",
                "arn:aws:s3 ::: awsexamplebucket1"
            ]
        },
        {
            "Sid": "ExampleStatement2",
            "Effect": "Allow",
            "Action": "s3:ListAllMyBuckets",
            "Resource": "*"
        }
    ]
}
```

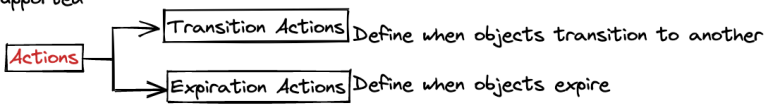
Bucket Configurations — Server-Access Logging

* When enabled, it captures details of the requests made to the bucket and objects in it
* Important for security, root-cause analysis and can be specified for important governance and compliance certifications and regulations
* It is not guaranteed and is conducted on a best-effort basis
* There is no hard and fast rule that every request will be captured or that a specific the request will be logged within a set timeframe
* By default, it is disabled
* While enabling, enter a target bucket where the log files will be stored with a target prefix for management/organization (optional)
* The source and target buckets must reside in the same region
* S3 requires specific permissions to write access logs to a target bucket
* These permissions require to write access to a 'Log Delivery Group' - a predefined S3 used to deliver log files to the target buckets
* If the configuration of the access logging is configured using the Management Console, the Log Delivery Group is automatically added to the Access Control List (ACL) of the target bucket allowing relevant access
* If the configuration of the access logging is configured using the S3 API or AWS SDK, these permissions need to be granted and managed manually
* The name of the log files stored in the bucket follow a standard naming pattern - `(prefix)YYYY-MM-DD-HH-MM-SS-UniqueString`
* It is possible to grant permission to deliver access logs through bucket Access Control Lists (ACLs), but not through a bucket policy
* Adding deny conditions to a bucket policy might prevent Amazon S3 from delivering access logs
* Can use default bucket encryption on the target bucket- only AES256 (SSE-S3); SSE-KMS encryption is not supported
* Cannot enable S3 Object Lock on the target bucket

By Dasika Madhu Nimeshika

Bucket Configurations — Lifecycle Rules

* A set of rules that define actions that Amazon S3 applies to a group of objects
* An automatic method of managing the lifecycle of data stored in a bucket
* With S3 Lifecycle configuration rules, Amazon S3 transitions objects to less expensive storage classes, or archive or delete them
* Configure specific criteria to automatically move data between different storage classes, Glacier or delete it altogether for cost optimization
* The time frame in the rule is configurable based on the requirements - archive data older than 30 days or delete data older than 90 days
* An S3 Lifecycle configuration is an XML file that consists of a set of rules with predefined actions that you want Amazon S3to perform on objects during their lifetime
* Define separate Lifecycle rules for current and noncurrent object versions
* Lifecycle configuration on multi-factor authentication (MFA)-enabled buckets is not supported



By Dasika Madhu Nimeshika

```
<LifecycleConfiguration>
   <Rule>
      <Filter>
         <Prefix>projectA/</Prefix>
      </Filter>
      transition/expiration actions.
      ...
   </Rule>

   <Rule>
      <Tag>
         <Key>key1</Key>
         <Value>value1</Value>
      </Tag>
      <Tag>
         <Key>key2</Key>
         <Value>value2</Value>
      </Tag>
      ...
   </Rule>

   <Rule>
      <Filter>
         <And>
            <Prefix>key-prefix</Prefix>
            <Tag>
               <Key>key1</Key>
               <Value>value1</Value>
            </Tag>
            <Tag>
               <Key>key2</Key>
               <Value>value2</Value>
            </Tag>
            ...
         </And>
      </Filter>
      <Status>Enabled</Status>
      transition/expiration actions.
   </Rule>
</LifecycleConfiguration>
```

Bucket Configurations — Object-Lock

* Object locking is an advanced property in an S3 bucket
* Enables WORM - Write Once Read Many
* Enables the object from being deleted at any point of time or enables the object to be deleted at a set point of time
* Ability to add retention lock on any object helps in the governance of any data that must be deleted after a specific amount of time
* Enabling the object log can only be done when the bucket is being created
* Error message is displayed when an object lock is set after bucket creation is complete
* Without enabling versioning, object lock cannot be enabled
* After creating a bucket with object lock enabled, it will be permanently enabled until the bucket is deleted and cannot be disabled otherwise
* Two retention modes are displayed when creating a bucket with object lock enabled
    1. Governance mode - Requires the retention period to be specified in days. When the retention the period is over, the object can be deleted.
    2. Compliance mode - Any user including the root user does not have permission to delete the object before the retention period is completed. Requires the retention period to be specified in days. Apart from a default retention period, a per object retention period can be set - can be done in the object properties.
* Specify the retention period and the date until which the object lock will be effective
* When this date is passed, the object log is no longer valid and the object can be deleted
* The 'legal hold' appears only at the object level and not the bucket level and do not have an expiration date
* The legal hold period can be terminated by a user who has the permission `s3:PutObjectLegalHold` `s3:PutObjectLegalHold` and `s3:GetBucketObjectLockConfiguration` in an IAM role allowing S3 Batch Operations to call S3 Object Lock on the users behalf
* Even if there is a retention period on an object, a legal hold can also be added to it
* In this case, if the retention period expires, the object will still be protected by the legal hold and cannot be deleted

By Dasika Madhu Nimeshika

Bucket Configurations — Versioning

By Dasika Madhu Nimeshika

* Allows multiple versions of the same object to exist
* Allows retrieving of previous object versions, recovery of an object's previous version on accidental deletion or intended malicious deletion
* Managed is automatically managed by AWS in a bucket that has versioning enabled
* Enables easy management- only the latest object version is displayed
* Not enabled by default
* After enabling, versioning cannot be disabled, only suspended
* Added cost for storing multiple versions of the same object
* All versions of the object can be retrieved when required
* S3 does not store all objects up to the point of suspension
* A bucket state can be any of these at one point in time
    1. Unversioned - Default state
    2. Versioning-Enabled - Extra cost is incurred due to storing multiple versions of the same object
    3. Versioning-Suspended - Once enabled, versioning cannot be completely disabled, it can only be suspended
* Enable Versioning via the Console on a new bucket or an existing bucket
* After enabling versioning on an existing bucket, existing objects are not associated with a version ID until they are modified or deleted and re-uploaded
* Every new version of an object receives a new version ID
* On deletion, a delete marker is added to the object
    1. A delete marker in Amazon S3 is a placeholder/marker for a versioned object that was named in a simple DELETE request
    2. Because the object was in a versioning-enabled bucket, the object was not deleted
    3. The delete marker makes Amazon S3 behave as if the object had been deleted as a request for viewing an object with the delete marker will return a 404 error
* Deleting an object which has versioning enabled has to be done via an AWS SDK in which the exact version ID of the object version to be deleted must be specified

Bucket Configurations — Requester Pays

* Usually, bucket owners pay for all Amazon S3 storage and data transfer costs that are associated with their buckets
* In a bucket configured with Requester Pays, the requester instead of the bucket owner pays the cost of the request and the data download from the bucket
* The bucket owner always pays the cost of storing data

> Configure buckets to be Requester Pays buckets when you want to share data but not incur charges associated with others accessing the data

* Authentication is a must for all requests involving Requester Pays buckets
* The request authentication enables Amazon S3 to identify and charge the requester for their use of the Requester Pays bucket
* When the requester assumes an AWS Identity and Access Management (IAM) role before making their request, the account to which the role belongs is charged for the request
* Requester Pays buckets do not support the following:
    1. Anonymous requests
    2. SOAP requests
    3. Using a Requester Pays bucket as the target bucket for end-user logging or vice versa
    4. Turn on end-user logging on a Requester Pays bucket where the target bucket is not a Requester Pays bucket
* How Requester Pays charges to work -
    1. The requester pays for the data transfer and the request
    2. The bucket owner pays for the data storage and other conditions -
        - The requester doesn't include the parameter `x-amz-request-payer` in the header (GET, HEAD, or POST) or as a parameter (REST) in the request (HTTP code 403)
        - Request authentication fails (HTTP code 403)
        - The request is anonymous (HTTP code 403)
        - The request is a SOAP request
* Configuring Requester Pays on a bucket
    1. Using the AWS S3 Console
    2. Using the REST API

Bucket Configurations

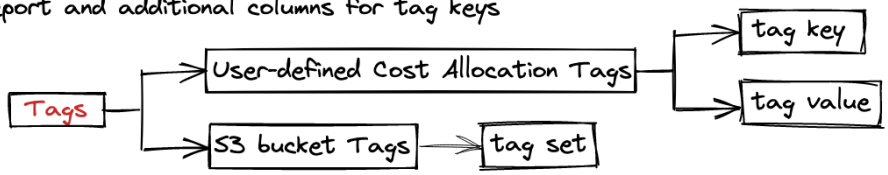
Transfer Acceleration

* Use an accelerated endpoint for faster data transfers
* USE S3TA to speed up content transfers to and from Amazon S3 by as much as 50-500% for long-distance transfer of larger objects
* Reduces the variability in Internet routing, congestion and speeds that can affect transfers, and logically shortens the distance to S3 for remote applications
* Improves transfer performance by routing traffic through Amazon CloudFront's globally distributed Edge Locations and over AWS backbone networks, and by using network protocol optimizations
* Why use Transfer Acceleration -
    1. Your customers upload to a centralized bucket from all over the world
    2. You transfer gigabytes to terabytes of data regularly across continents
    3. You cannot use all available bandwidth over the internet when uploading to S3
* Benefits of using S3TA -
    1. Move data faster over long distances
    2. Reduce network variability
    3. Shorten the distance to S3
    4. Maximize bandwidth utilization
* Enable Transfer Acceleration on a bucket -
    1. TheAmazon S3 console
    2. The REST API PUT Bucket accelerate the operation
    3. The AWS CLI and AWS SDKs
* Use the Amazon S3 Transfer Acceleration Speed Comparison tool to compare accelerated and non-accelerated upload speeds across Amazon S3 Regions
* The Speed Comparison tool uses multipart uploads to transfer a file from a browser to various Amazon S3 Regions with and without using Transfer Acceleration

Bucket Configurations

Tags

* Used to categorize storage and track storage cost or other criteria by tagging buckets
* Track the storage cost or other criteria for individual projects or groups of projects, label buckets using cost allocation tags - the report contains the same line items as the detailed billing report and additional columns for tag keys



* User-defined cost allocation tags -
  1. Tag Key
     - name of the tag
     - case-sensitive string
     - can contain 1 to 128 Unicode characters
  2. Tag Value
     - a required string
     - case-sensitive string
     - can contain from 0 to 256 Unicode characters
* S3 bucket tags
     - Has a tag set that contains all of the tags that are assigned to that bucket
     - a tag set can contain as many as 50 tags, or it can be empty
     - keys must be unique within a tag set, but values in a tag set don't have to be unique
     - adding a tag that has the same key as an existing tag, the new value overwritten over the old value
     - AWS doesn't apply any semantic meaning to tags, they are nterpreted strictly as character strings
     - Use the Amazon S3 console, the AWS Command Line Interface (AWS CLI), or the Amazon S3 API to add, list, edit, or delete tags

By Dasika Madhu Nimeshika

Bucket Configurations — Encryption

* AWS S3 provides default encryption for buckets
* On enabling default encryption, all the newly added objects will be encrypted
* However, the previously existing objects prior to enabling encryption will not be encrypted
* Default encryption can be enabled from the AWS management console in the Properties tile in the S3 bucket under the Default Encryption
* Requires minimal configuration and is managed by AWS
* Choose one of the two existing encryption options including AES-256 and AWS-KMS
    1. AES-256 (SSE-S3)
    2. Stands for server-side encryption on S3
    3. It is an S3 managed Key
* AWS-KMS (SSE-KMS)
    1. Stands for server-side encryption managed by the key management service
    2. Greater flexibility in managing the keys
* Using server-side encryption, Amazon S3 encrypts an object before saving it to disk and decrypts the objects when they are downloaded
* Enable default encryption on a bucket -
    1. The Amazon S3 console
    2. The AWS CLI
    3. The AWS SDKs
    4. The REST API
* S3 offers various server-side (SSE) and client-side (CSE) encryption mechanisms
    - SSE-S3 (S3 managed keys)
    - SSE-KMS (Key Management Service managed keys)
    - SSE-C (Customer managed keys)
    - CSE-KMS (Key Management Service managed keys)
    - CSE-C (Customer managed keys)
    - Client-side - Client part uploading objects
    - Server-side - AWS S3
    - S3 fully supports encryption in transit (via SSL/TLS - Secure Sockets Layer/Transport Layer Security)

Server-Side Encryption withS3 managed keys (SSE-S3)

* Requires minimal configuration
* Management of encryption keys managed by AWS
* Upload your data and S3 will handle all other aspects

Server-Side Encryption with KMS managed keys (SSE-KMS)

* Allows S3 to use the Key Management Service to generate data encryption keys
* Gives greater flexibility of key management: disable, rotate and apply access controls to the CMK

Server-Side Encryption with Customer provided keys (SSE-C)

* Gives the opportunity to provide personal Master keys
* The customer provided key would be sent with the data to S3, where S3 would then perform the encryption

Client-Side Encryption with KMS managed keys (CSE-KMS)

* Uses the Key Management Service to generate data encryption keys
* KMS is called upon via the client, not S3
* Encryption takes place client-side and the encrypted data is then sent to S3

Client-Side Encryption with Customer provided keys (CSE-C)

* Utilize your own provided keys
* Use an AWS SDK Client to encrypt data before sending it to S3 for storage

By Dasika Madhu Nimeshika

Bucket Configurations — Event Notifications

* Receive notifications when certain events happen in the bucket
* Add a notification configuration that identifies the events required to publish and the destinations to send the notifications
* Amazon S3 provides an API for you to manage this subresource
* Events supported -
    1. New object created events
    2. Object removal events
    3. Restore object events
    4. Reduced Redundancy Storage (RRS) object lost events
    5. Replication events
* Publish event notifications to services -
    1. Amazon Simple Notification Service (SNS) topic
    2. Amazon Simple Queue Service (SQS) queue
    3. AWS Lambda
* Enable events via -
    1. Amazon S3 console
    2. Programmatically using the AWS SDKs
* Before Amazon S3 can publish event notification messages to a destination, the S3 principal must have the necessary permissions to call the relevant API to publish messages to an SNS topic, an SQS queue, or a Lambda function

By Dasika Madhu Nimeshika

Bucket Configurations — Location

* A location is an endpoint for the Amazon S3 bucket that DataSync uses as a source or destination
* Has to be configured in the DataSync console
* To use as a source or a destination is an Amazon S3 bucket, configure -
    1. S3 bucket
    2. Storage class
    3. Folder
    4. IAM role
    5. Tag
    6. Access point
    7. Agents

By Dasika Madhu Nimeshika

Replication

* Enables automatic, asynchronous copying of objects across Amazon S3 buckets
* Buckets that are configured for object replication can be owned by the same AWS account or by different accounts
* Objects may be replicated to a single destination bucket or multiple destination buckets
* Destination buckets can be in different AWS Regions or within the same Region as the source bucket
* Disabled by default
* To enable object replication, add a replication configuration to your source bucket. Minimum configuration must provide the following -
    1. The destination bucket or buckets where you want Amazon S3 to replicate objects
    2. An AWS Identity and Access Management (IAM) role that Amazon S3 can assume to replicate objects on your behalf
* Why use replication -
    1. Replicate objects while retaining metadata
    2. Replicate objects into different storage classes
    3. Maintain object copies under different ownership
    4. Keep objects stored over multiple AWS Regions
    5. Replicate objects within 15 minutes
* When to use cross-region replication (CRR)
    1. Meet compliance requirements
    2. Minimize latency
    3. Increase operational efficiency
* When to use same-region replication (SRR)
    1. Aggregate logs into a single bucket
    2. Configure live replication between production and test accounts
    3. Abide by data sovereignty laws

Bucket Configurations — Static Website Hosting

* Use Amazon S3 to host a static website
* On a static website, individual web pages include static content and client-side scripts
* A dynamic website relies on server-side processing, including server-side scripts such as PHP, JSP, or ASP.NET
* Amazon S3 does not support server-side scripting, but AWS has other resources for hosting dynamic websites
* After configuring the bucket as a static website, the website is available at the AWS Region-specific website endpoint of the bucket
* Website endpoints are different from the endpoints where you send REST API requests
* Depending on your Region, your Amazon S3 website endpoint follows one of these two formats
  `s3-website dash (-) Region - http://bucket-name.s3-website-Region.amazonaws.com`
  `s3-website dot (.) Region - http://bucket-name.s3-website.Region.amazonaws.com`
* Disabled by default
* Enabling website hosting -
    1. TheAmazon S3 console
    2. The REST API
    3. The AWS SDK
    4. The AWS CLI
* Hosting type
    1. Host a static website
    2. Redirect requests for an object
* Configure and upload an index.html document - a webpage that Amazon S3 returns when a the request is made to the root of a website or any subfolder
* Add an optional error.html document which is returned when an error occurs
* Add optional Redirection rules, written in JSON, automatically redirect webpage requests for specific content

Bucket Configurations

AWS CloudTrail data events

* Retrieve information about a bucket and object-level requests in Amazon S3
* Create a trail manually in CloudTrail to enable this feature
* CloudTrail stores Amazon S3 data event logs in a preconfigured bucket
* It logs Amazon S3 object-level API operations in the CloudTrail console
* To enable a trail:
    1. Trail name (Allows 3-128 characters. Only letters, numbers, periods, underscores, and dashes are allowed)
    2. Choose to 'Enable for all accounts in my organization' (To review accounts in your organization, open AWS Organizations)
    3. Choose a 'Storage location' (Existing S3 bucket or create new S3 bucket)
    4. Trail log bucket and folder (Enter a new S3 bucket name and folder (prefix) to store your logs. Bucket names must be globally unique.)
    5. Enable encryption - Logfile SSE-KMS encryption or Customer managed AWS KMS key
    6. Choose to 'Enable log file validation' - determine whether a log file was modified, deleted, or unchanged after AWS CloudTrail delivered it, use CloudTrail log file integrity validation. This feature is built using industry-standard algorithms: SHA-256 for hashing and SHA-256 with RSA for digital signing.
    7. Choose to 'Send SNS notifications for log file delivery' - enable to be notified each time a log is delivered to a bucket. CloudTrail stores multiple events in a log file. Post enabling this option, Amazon SNS notifications are sent for every log file delivery to an S3 bucket, not for every event.
    8. Choose to 'Send events to CloudWatch Logs' - Configure CloudWatch Logs to monitor trail logs and notify you when specific activity occurs. **Standard CloudWatch and CloudWatch Logs charges apply.
    9. Add tags (optional)
* Turn on a trail across all regions for your AWS account. CloudTrail will deliver log files from all regions to the Amazon S3 bucket and an optional CloudWatch Logs log group specified. Additionally, when AWS launches a new region, CloudTrail will create the same trail in the new region. Receive log files containing API activity for the new region without taking any action.
* To disable object-level logging for the bucket, you must open the CloudTrail console and remove the bucket name from the trail's Data events.

By Dasika Madhu Nimeshika

S3 STORAGE CLASSES

Storage Classes

Storage classes offer a range of options for storing data for various use cases. They are S3 Standard (general-purpose storage of frequently accessed data), S3 Intelligent-Tiering (data with unknown access patterns), S3 Standard - Infrequent Access(IA) and S3 One Zone - Infrequent Access (long-lived, but less frequently accessed data) and S3 Glacier and S3 Glacier Deep Archive (long-term archive and digital preservation). Use S3 Lifecycle policies to automatically transition objects between storage classes without any application changes.

S3 Standard

* General purpose
* Offers high durability, availability, and performance object storage for frequently accessed data
* Delivers low latency and high throughput, S3 Standard is appropriate for a wide variety of use cases
* Configured at the object level and a single bucket can contain objects stored across S3 Standard, S3 Intelligent-Tiering, S3 Standard-IA, and S3 One Zone-IA
* Use S3 Lifecycle policies to automatically transition objects between storage classes without any application changes
* Low latency and high throughput performance
* Designed for durability of 99.999999999% of objects across multiple Availability Zones
* Resilient against events that impact an entire Availability Zone
* Designed for 99.99% availability over a given year - backed with the Amazon S3 Service Level Agreement
* Supports SSL for data in transit and encryption of data at rest
* Use the S3 Lifecycle management for automatic migration of objects to other S3 Storage Classes

S3 Intelligent-Tiering

* Unknown or changing access
* Delivers automatic cost savings by moving objects between four access tiers when objects have infrequent access patterns
* Optimize costs by automatically moving objects to the most cost-effective access tier
* Reduced admin operations overhead
* Stores objects in four access tiers, optimized for frequent, infrequent, archive, and deep archive access
* Data is moved between the Frequently Accessed and Infrequently Accessed tiers (have the same low latency and high throughput performance of S3 Standard)
* If data is not accessed within 30 days, it is moved to the Infrequently Accessed tier
* Once accessed, the object is moved back into the Frequently Accessed tier and the 30-day timer is reset
* Data to be archived is moved between the Archive access and deep Archive access tiers (have the same performance as Glacier and Glacier Deep Archive)
* If data is not accessed for 90 consecutive days, it will be moved to the archive access tier
* After not accessing data for 180 consecutive days, it is automatically moved to the deep archive access tier
* Designed for durability of 99.999999999% of objects across multiple Availability Zones
* Designed for 99.9% availability over a given year
* Incurs a small monthly monitoring and auto-tiering fee
* No retrieval fees, no additional tiering fees apply when objects are moved between access tiers within the S3 Intelligent-Tiering storage class
* Each object must be larger than 128 KB

S3 Standard - Infrequent Access

* For data that is accessed less frequently, but requires rapid access when needed
* Offers high durability, high throughput, and low latency of S3 Standard, with a low per GB storage price and per GB retrieval fee
* Due to the low cost and high performance, it is ideal for long-term storage, backups, and as a data store for disaster recovery files
* Provides the same low latency and high throughput performance of S3 Standard
* Designed for durability of 99.999999999% of objects across multiple Availability Zones
* Designed for 99.9% availability over a given year - backed with the Amazon S3 Service Level Agreement
* Resilient against events that impact an entire Availability Zone and in the event of an entire AZ destruction
* Supports SSL for data in transit and encryption of data at rest
* Use the S3 Lifecycle management for automatic migration of objects to other S3 Storage Classes

S3 One Zone - Infrequent Access

* For data that is accessed less frequently, but requires rapid access when needed
* Stores data in a single AZ and costs 20% less than S3 Standard-IA
* Ideal for customers who want a lower-cost option for infrequently accessed data but do not require the availability and the resilience of S3 Standard or S3 Standard-IA
* Store secondary backup copies of on-premises data or easily re-creatable and non-critical data
* Cost-effective storage for data replicated from another AWS Region using S3 Cross-Region Replication
* Offers the same high durability†, high throughput, and low latency of S3 Standard, with a low per GB storage price and per GB retrieval fee
* Designed for durability of 99.999999999% of objects across multiple Availability Zones
* Resilient against events that impact an entire Availability Zone
* Designed for 99.5% availability over a given year - backed with the Amazon S3 Service Level Agreement
* Supports SSL for data in transit and encryption of data at rest
* Use the S3 Lifecycle management for automatic migration of objects to other S3 Storage Classes
* Data stored in this storage class will be lost in the event of Availability Zone destruction

Storage Classes

S3 Glacier

* For data that requires secure, durable, and low-cost storage class for archiving
* Reliably store any amount of data at costs that are cheaper than on-premises solutions
* S3 Glacier provides three retrieval options that range from a few minutes to hours
    1. Expedited - Access data quickly when occasional urgent requests for a subset of the archives are required. For all the objects other than the largest archived objects, data accessed are typically made available within 1–5 minutes. There are two types of Expedited retrievals -
        1. On-Demand requests - Available most of the time.
        2. Provisioned requests - Guaranteed to be available when required.
    2. Standard - Access any archived objects within several hours. Standard retrievals typically complete within 3–5 hours. This is the default option for retrieval requests that do not specify the retrieval option.
    3. Bulk - The lowest-cost retrieval option, enabling you to retrieve large amounts, even petabytes, of data inexpensively in a day. Bulk retrievals typically take 5–12 hours to retrieve requested data.
* Designed for durability of 99.999999999% of objects across multiple Availability Zones
* Data is resilient in the event of one entire Availability Zone destruction
* Supports SSL for data in transit and encryption of data at rest
* Use the S3 PUT API for direct uploads to S3 Glacier

S3 Glacier Deep Archive

* For lowest-cost storage class and supports long-term retention and digital preservation for data that may be accessed once or twice in a year
* Designed for use cases that retain data sets for 7-10 years or longer to meet regulatory compliance requirements
* Can also be used for backup and disaster recovery use cases, and is a cost-effective and easy-to-manage alternative to magnetic tape systems, whether they are on-premises libraries or off-premises services
* Objects stored are replicated and stored across at least three different geographically Availability Zone and can be restored within 12 hours or less
* Provides bulk data retrieval option which retrieves petabytes of data within 48 hours
* Designed for durability of 99.999999999% of objects across multiple Availability Zones
* Data is resilient in the event of one entire Availability Zone destruction
* Supports SSL for data in transit and encryption of data at rest
* Use the S3 PUT API for direct uploads to S3 Glacier

S3 Outposts

* Amazon S3 on Outposts delivers object storage to an on-premises AWS Outposts environment
* Provides feasibility in storing and retrieving data on an Outpost, as well as secure the data, control access, tag, and report on it
* Outposts provide a single storage class - S3 Outposts
    1. It uses the S3 API
    2. It is designed to durably and redundantly store data across multiple devices and servers on the Outposts
* It is ideal for workloads with local data residency requirements, and to satisfy demanding performance needs by keeping data close to on-premises applications
* S3 Object compatibility and bucket management can be done through the S3 SDK
* Offers data storage durability and redundancy
* Encryption can be added using SSE-S3 and SSE-C
* Authentication and authorization can be added via IAM, and S3 Access Points
* Additional provision to transfer data to AWS Regions using AWS DataSync
* Add S3 Lifecycle expiration actions to delete data automatically after a specific period

2

	*General Purpose	*Optimize storage costs	*Infrequently Accessed Objects		*Archival Purposes		*Not Recommended
Storage Classes	S3 Standard	S3 Intelligent-Tiering	S3 Standard - Infrequent Access	S3 One Zone - Infrequent Access	S3 Glacier	S3 Glacier Deep Archive	Reduced Redundancy Storage (RRS)
Designed for Use Case	Frequently accessed data	Long-lived data with unpredictable accessed patterns	Long-lived, infrequently accessed data	Long-lived, infrequently accessed, non-critical data	Archive infrequently accessed data	Long-term retention of infrequently accessed data	Frequently accessed, non-critical data
Designed for durability	99.999999999% (11 9's)						99.99%
Designed for availability	99.99%	99.9%	99.9%	99.5%	99.99%	99.99%	99.99%
Availability SLA	99.9%	99%	99%	99%	99.9%	99.9%	99.9%
Availability Zones	≥3	≥3	≥3	1	≥3	≥3	≥3
Minimum capacity charge per object	N/A	N/A	128KB	128KB	40KB	40KB	N/A
Minimum storage duration charge	N/A	30 days	30 days	30 days	90 days	180 days	N/A
Retrieval fee	N/A	N/A	per GB retrieved	per GB retrieved	per GB retrieved	per GB retrieved	N/A
First byte latency	milliseconds	milliseconds	milliseconds	milliseconds	select minutes or hours	select hours	milliseconds
Storage type	Object						
Lifecycle transitions	Yes						

S3 OBJECT BASICS

S3 Objects

* Object is a file uploaded to a bucket and any metadata that describes that file
* Each S3 object has data, a key, and metadata
* After uploading the object is in the bucket, it can be opened, downloaded, and moved
* When the object is no longer need an object or a bucket, it can be deleted
* Objects have unique key-values to identify them individually
* Each object can be between 0 and 5 TB in size
* An object consists of the following -
    1. Key - Name that is assigned to an object. An object can be retrieved using the unique key.
    2. Version ID - A key and version ID uniquely identify an object. S3 generates a version ID (string) object to a bucket.
    3. Value - Content that is stored in a bucket. It is a sequence of bytes. Objects can range in size from zero to 5 TB.
    4. Metadata - A set of name-value pairs with which information regarding the object is stored. Assign user-defined metadata to objects. S3 manages objects by assigns system metadata to objects.
    5. Subresources - Stores object-specific information as subresources as they are always associated with some other entity such as an object or a bucket.
    6. Access Control Information - Used to control access to the objects. Objects are private by default. It is a list of grants identifying the grantees and the permissions granted.
* Two kinds of metadata -
    1. System metadata - Metadata assigned to the object by S3
    2. User-defined metadata - A key-value pair provided by the bucket/object owner
* Any file uploaded to S3 is stored as an S3 object
* The number of objects in a bucket is unlimited
* S3 supports a large number of file types - images, backups, data, movies, text, etc
* The maximum size of a file that can be uploaded by using the Amazon S3 console is 160 GB
* To upload a file larger than 160 GB, use the AWS CLI, AWS SDK, or Amazon S3 REST API
* Uploading an object with a key name that already exists in a versioning-enabled bucket creates another version of the object instead of replacing the existing object
* Uploading options -
    1. Upload an object in a single operation using the AWS SDKs, REST API, or AWS CLI
    2. Upload a single object using the S3 Console
    3. Upload an object in parts using the AWS SDKs, REST API, or AWS CLI
* Multipart upload process -
    1. Initiate the upload
    2. Upload the object parts
    3. After uploading all the parts, complete the multipart upload
* The copy operation creates a copy of an object that is already stored in S3
* Create a copy of your object up to 5 GB in a single atomic operation
* To copy an object that is greater than 5 GB, use the multipart upload API
* Download a single object per request using the S3 console
* Data transfer fees apply when you download objects
* Downloading multiple objects can be done via the AWS CLI, AWS SDKs, or REST API
* While downloading an object programmatically, its metadata is returned in the response headers
* Delete one or more objects directly from S3 using the S3 console, AWS SDKs, AWS CLI, or REST API
* API options when deleting an object -
    1. Delete a single object
    2. Delete multiple objects
* Use prefixes and folders to organize buckets
    1. A prefix is a logical grouping of the objects in a bucket
    2. In the S3 console, prefixes are called folders
* Presigned URLs - All objects and buckets are private by default. Use a presigned URL to optionally share objects or enable your customers/users to upload objects to buckets without AWS security credentials or permissions
* Transform objects with Object Lambda using code to S3 GET requests to modify and process data as it is returned to an application - S3 Object Lambda uses AWS Lambda functions to automatically process the output of a standard S3 GET request

Object Systemr-defined Metadata

Name	Description	Can the user modify the value?
Date	Current date and time	No
Content-Length	Object size in bytes	No
Last-Modified	Object creation date or the last modified date, whichever is the latest	No
Content-MD5	The base64-encoded 128-bit MD digest of the object	No
x-amz-server-side-encryption	Indicates whether server-side encryption is enabled for the object, and whether that encryption is from the AWS Key Management Service (AWS KMS) or from Amazon S3 managed encryption (SSE-S3)	Yes
x-amz-version-id	Object version. When you enable versioning on a bucket, Amazon S3 assigns a version number to objects added to the bucket.	No
x-amz-delete-marker	In a bucket that has versioning enabled, this Boolean marker indicates whether the object is a delete marker.	No
x-amz-storage-class	Storage class used for storing the object	Yes
x-amz-website-redirect-location	Redirects requests for the associated object to another object in the same bucket or an external URL.	Yes
x-amz-server-side-encryption-aws-kms-key-id	If x-amz-server-side-encryption is present and has the value of aws:kms, this indicates the ID of the AWS KMS symmetric customer master key (CMK) that was used for the object.	Yes
x-amz-server-side-encryption-customer-algorithm	Indicates whether server-side encryption with customer-provided encryption keys (SSE-C) is enabled.	Yes

By Dasika Madhu Nimeshika

*AWS Definitions

Object User-defined Metadata

*AWS Definitions

* User-defined metadata is optional information that has to be defined as a key-value pair while sending a PUT or POST request to create the object
* S3 stores user-defined metadata keys in lowercase
* S3 allows arbitrary Unicode characters in your metadata values
* While uploading the objects using REST API, the optional user-defined metadata names must begin with "x-amz-meta-" to distinguish them from other HTTP headers
* While retrieving the object, the prefix is returned as well
* While uploading objects using the SOAP API, the prefix is not required
* While retrieving the object using the SOAP API, the prefix is removed, regardless of which API is used to upload the object
* When metadata is retrieved through the REST API, Amazon S3 combines headers that have the same name (ignoring case) into a comma-delimited list
* If some metadata contains unprintable characters, it is not returned
* The x-amz-missing-meta header is returned with a value of the number of unprintable metadata entries
* The HeadObject action retrieves metadata from an object without returning the object itself
* To avoid issues around the presentation of these metadata values, conform to using US-ASCII characters when using REST and UTF-8 when using SOAP or browser-based uploads via POST
* When using non US-ASCII characters in metadata values, the provided Unicode string is examined for non US-ASCII characters
* If the string contains only US-ASCII characters, it is presented as is
* If the string contains non US-ASCII characters, it is first character-encoded using UTF-8 and then encoded into US-ASCII

By Dasika Madhu Nimeshika

```
PUT /Key HTTP/1.1
Host: awsexamplebucket1.s3.amazonaws.com
x-amz-meta-nonascii: ÄMÄZÕÑ S3

HEAD /Key HTTP/1.1
Host: awsexamplebucket1.s3.amazonaws.com
x-amz-meta-nonascii: =?UTF-8?B?w4PChE3Dg8KEWsODwpXDg8KRIFMz?=

PUT /Key HTTP/1.1
Host: awsexamplebucket1.s3.amazonaws.com
x-amz-meta-ascii: AMAZONS3

HEAD /Key HTTP/1.1
Host: awsexamplebucket1.s3.amazonaws.com
x-amz-meta-ascii: AMAZONS3
```

S3 Data Consistency Model

* Strongly Consistent
    1. The object data is available immediately after an operation is performed
    2. For example, if a PUT operation is performed by one user and another user simultaneously performs a GET operation, S3 returns the latest version of the object
* Eventually Consistent
    1. The object data is available after a few seconds after an operation is executed
    2. For example, if a PUT operation is performed by one user and another user simultaneously performs a GET operation, S3 may return the latest object or the last available version
    3. However corrupt or partial data will never be returned

Object Consistency

* S3 provides strong read-after-write and list consistency automatically for all applications.
* Read operations on S3 Select, S3 Access Control Lists, S3 Object Tags, and object metadata are strongly consistent
* S3 achieves high availability by replicating data across multiple servers within the available data centres
* If an object PUT request is successful, then the object data is stored safely. Any read (GET or LIST) that is initiated following a successful PUT response will return the data written by the PUT.

Bucket Consistency

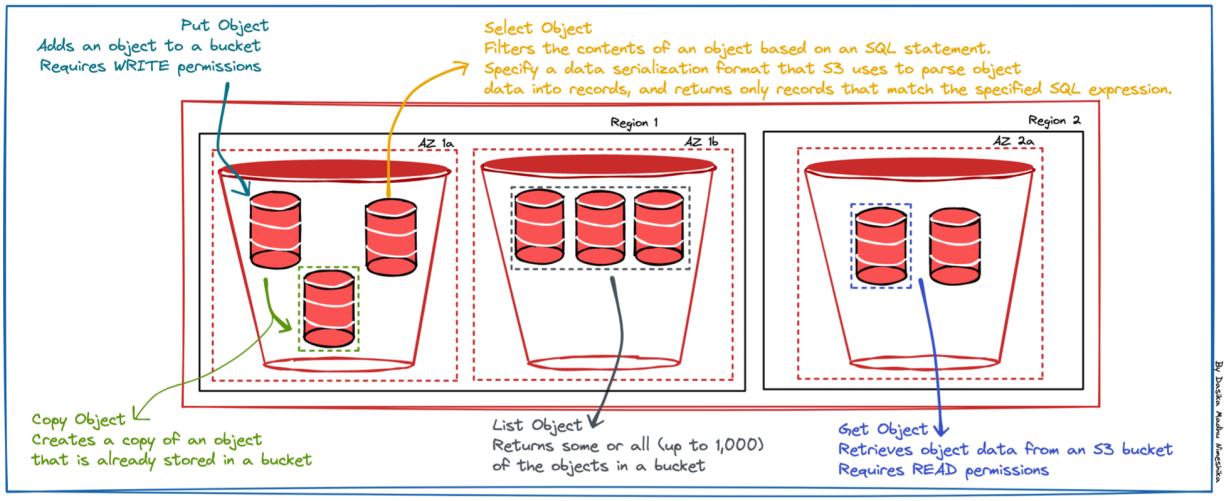
* Bucket configurations follow eventual consistency
    1. Deleting a bucket and immediately listing the buckets may retrieve the deleted bucket name
    2. Enabling versioning on a bucket may take some time to fully propagate all the changes. AWS recommends to wait for 15 minutes after enabling versioning before issuing write operations (PUT or DELETE) on objects in the bucket.

By Dasika Madhu Nimeshika

S3 PRICING

S3 Pricing

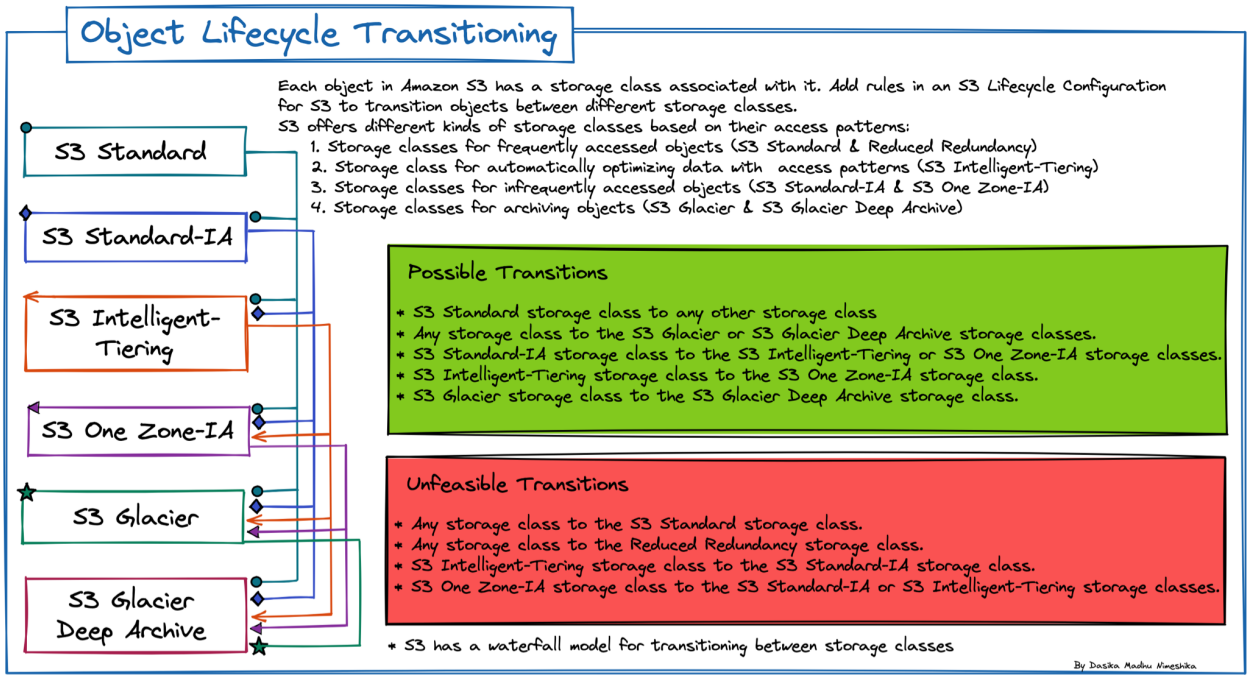
* As per the cloud pricing model in S3, you only pay for the resources you use.
* S3 Pricing is region-dependent - different regions have different costs
* It also depends on the storage class
* The RRS is more expensive than the Standard SC (The former was introduced to lower costs with a tradeoff for lower durability. The latter now provides greater resilience for a lower price.)
* Cost optimal option - Standard IA with 99.9% availability and the same durability as the Standard SC
* Charges incurred are the same despite the mode of accessing the objects including the console, API or SDK
* There is a threshold storage (per GiB) limit which when crossed, reduces the costs



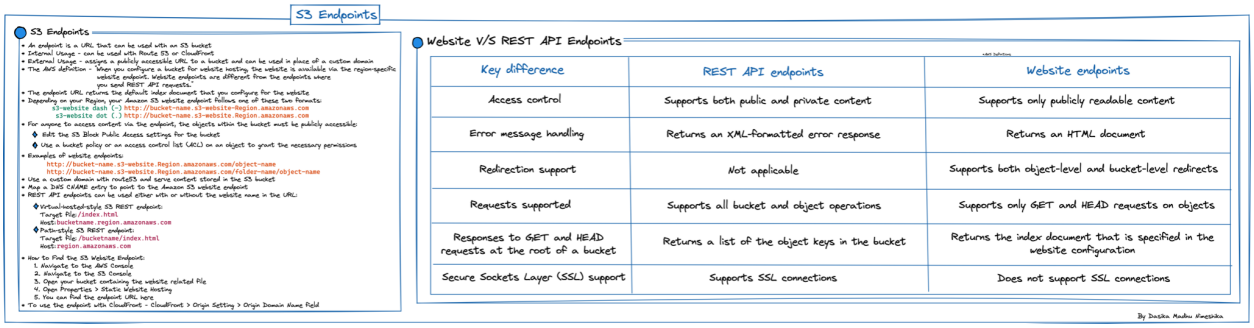
* Costs in S3 are categorized under -

  - Storage - Pay for storing objects in S3. The cost incurred depends on the objects' size, the storage duration, and the storage class. Storing objects in the S3 Intelligent Storage Class incur an additional monthly monitoring and automation fee.
  - Requests and Retrievals - Request Costs (PUT, COPY, POST, LIST, GET, SELECT, Lifecycle Transition, and Data Retrievals) are charged on a per 1,000 requests basis. The cost of requests depends on the storage class. DELETE and CANCEL requests are free. PUT, COPY, and POST requests vary from one storage class to another. LIST operations are the same as S3 Standard PUT/COPY/POST operations. S3 Standard – Infrequent Access, S3 One Zone – Infrequent Access, S3 Glacier, and S3 Glacier Deep Archive storage charge for data retrieval.
  - Data Transfer - Data Transfer in from the internet, out to CloudFront and between S3 buckets in the same AWS Region is free. Data transferred out to an EC2 instance when the instance is in the same AWS Region as the S3 bucket (including to a different account in the same AWS region) is free. S3 Transfer Acceleration pricing is added on top of the Data Transfer pricing.
  - Management and Analytics - Storage management features and analytics in S3 include S3 Inventory, S3 Storage Class Analysis, S3 Storage Lens, and S3 Object Tagging. AWS CloudTrail data events and CloudWatch metrics incur additional costs respectively.
  - Replication - In S3 Replication, both Cross-Region Replication and Same Region Replication, S3 charges for storage in the selected destination S3 storage classes, the storage charges for the primary copy, replication PUT requests, and applicable infrequent access storage retrieval fees. For CRR, inter-region Data Transfer OUT from S3 to each destination region is also charged. While using S3 Replication Time Control, cost accrued includes Replication Time Control Data Transfer fee. Amazon S3 Replication Time Control Data Transfer pricing is the same in all AWS Regions. S3 Replication Metrics charges that are billed at the same rate as CloudWatch custom metrics. Storage and PUT request pricing for the replicated copy is based on the selected destination AWS Regions, while pricing for inter-region data transfers are based on the source AWS Region.
  - Object Lambda - Add custom code to GET requests to modify and process data as it is returned to an application. Can be executed on standard S3 GET requests to filter rows, dynamically resize images, redact confidential data, and much more. The cost incurred depends on the S3 request and Lambda function - the duration and memory allocated to the Lambda function. S3 request and Lambda prices depend on the AWS Region.

S3 OBJECT LIFECYCLE TRANSITIONING



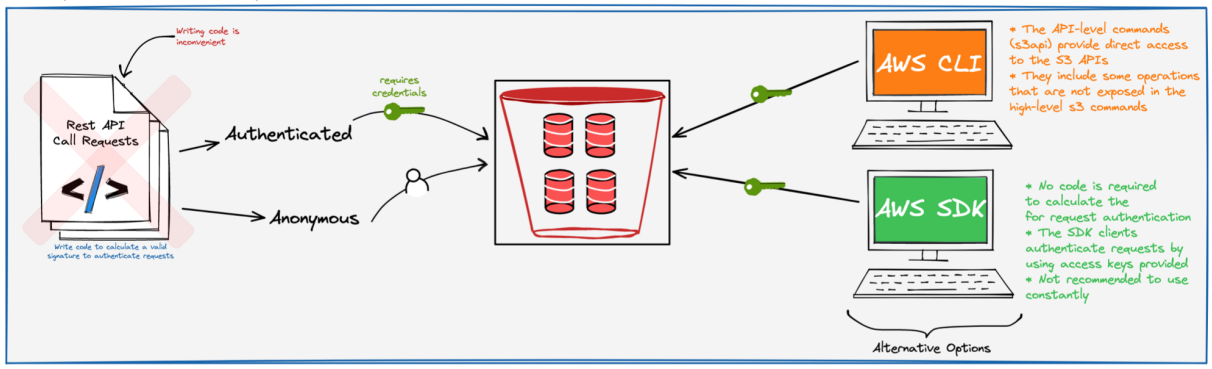
S3 ENDPOINTS



S3 API

S3 API

* The S3 application programming interface (API) describes various API operations, related request and response structures, and error codes
* Current version of the Amazon S3 API is 2006-03-01
* S3 supports REST API
* Support for SOAP is deprecated (still available over HTTPS but not for the newer features)
* S3 APIs are grouped into two sets with no specific functional distinction: Amazon Simple Storage Service and AWS S3 Control
  - Bucket-level & Object-level Actions
  - Account-level Actions
* Having valid credentials to log in is not sufficient. To be able to access or perform operations of resources, the user requires additional permissions.



* All S3 requests are either authenticated or anonymous
* Authentication with AWS Signature Version 4 depends on the request signature:
  1. Verification of the identity of the requester via their access keys (access key ID, secret access key).
  2. In-transit data protection is done by using the request elements to calculate the request signature pre-request and post-request. If both are not the same, S3 rejects the request.
  3. Signed portions of the request are valid for 15 minutes of the timestamp in the request. An unauthorized party who has access to a signed request can modify the unsigned portions of the request without affecting the request's validity in the 15-minute window.
* Authentication Methods:
  1. HTTP Authorization header is the most common authorization method. All of the Amazon S3 REST operations (except for browser-based uploads using POST requests) require this header.
  2. Query string parameters are used to express a request in a URL format. Use query parameters to provide request information, including the authentication information. Since the request signature is part of the URL, this type of URL is often referred to as a presigned URL.

By Dasika Madhu Nimeshika

S3 API - Supported by S3 Control

ACTIONS

CreateAccessPoint
CreateAccessPointForObjectLambda
CreateBucket
CreateJob
DeleteAccessPoint
DeleteAccessPointForObjectLambda
DeleteAccessPointPolicy
DeleteAccessPointPolicyForObjectLambda
DeleteBucket
DeleteBucketLifecycleConfiguration
DeleteBucketPolicy
DeleteBucketTagging
DeleteJobTagging
DeletePublicAccessBlock
DeleteStorageLensConfiguration
DeleteStorageLensConfigurationTagging
DescribeJob
GetAccessPoint
GetAccessPointConfigurationForObjectLambda
GetAccessPointForObjectLambda
GetAccessPointPolicy
GetAccessPointPolicyForObjectLambda
GetAccessPointPolicyStatus
GetAccessPointPolicyStatusForObjectLambda
GetBucket
GetBucketLifecycleConfiguration
GetBucketPolicy
GetBucketTagging
GetJobTagging
GetPublicAccessBlock
GetStorageLensConfiguration
GetStorageLensConfigurationTagging
ListAccessPoints
ListAccessPointsForObjectLambda
ListJobs
ListRegionalBuckets
ListStorageLensConfigurations
PutAccessPointConfigurationForObjectLambda
PutAccessPointPolicy
PutAccessPointPolicyForObjectLambda
PutBucketLifecycleConfiguration
PutBucketPolicy
PutBucketTagging
PutJobTagging
PutPublicAccessBlock
PutStorageLensConfiguration
PutStorageLensConfigurationTagging
UpdateJobPriority
UpdateJobStatus

DATA TYPES

AbortIncompleteMultipartUpload
AccessPoint
AccountLevel
ActivityMetrics
AwsLambdaTransformation
BucketLevel
CreateBucketConfiguration
Exclude
Include
JobDescriptor
JobFailure
JobListDescriptor
JobManifest
JobManifestLocation
JobManifestSpec
JobOperation
JobProgressSummary
JobReport
LambdaInvokeOperation
LifecycleConfiguration
LifecycleExpiration
LifecycleRule
LifecycleRuleAndOperator
LifecycleRuleFilter
ListStorageLensConfigurationEntry
NoncurrentVersionExpiration
NoncurrentVersionTransition
ObjectLambdaAccessPoint
ObjectLambdaConfiguration
ObjectLambdaContentTransformation
ObjectLambdaTransformationConfiguration
PolicyStatus
PrefixLevel
PrefixLevelStorageMetrics
PublicAccessBlockConfiguration
RegionalBucket
S3AccessControlList
S3AccessControlPolicy
S3BucketDestination
S3CopyObjectOperation
S3DeleteObjectTaggingOperation
S3Grant
S3Grantee
S3InitiateRestoreObjectOperation
S3ObjectLockLegalHold
S3ObjectMetadata
S3ObjectOwner
S3Retention
S3SetObjectAclOperation
S3SetObjectLegalHoldOperation
S3SetObjectRetentionOperation
S3SetObjectTaggingOperation
S3Tag
SelectionCriteria
SSEKMS
SSES3
StorageLensAwsOrg
StorageLensConfiguration
StorageLensDataExport
StorageLensDataExportEncryption
StorageLensTag
Tagging
Transition
VpcConfiguration

By Daskia Madhu Nimeshika

Supported by S3 on Outposts

ACTIONS

CreateEndpoint
DeleteEndpoint
ListEndpoints

DATA TYPES

Endpoint
NetworkInterface

By Dasika Madhu Nimeshika

S3 API - Supported by Amazon S3

ACTIONS

AbortMultipartUpload
CompleteMultipartUpload
CopyObject
CreateBucket
CreateMultipartUpload
DeleteBucket
DeleteBucketAnalyticsConfiguration
DeleteBucketCors
DeleteBucketEncryption
DeleteBucketIntelligentTieringConfiguration
DeleteBucketInventoryConfiguration
DeleteBucketLifecycle
DeleteBucketMetricsConfiguration
DeleteBucketOwnershipControls
DeleteBucketPolicy
DeleteBucketReplication
DeleteBucketTagging
DeleteBucketWebsite
DeleteObject
DeleteObjects
DeleteObjectTagging
DeletePublicAccessBlock
GetBucketAccelerateConfiguration
GetBucketAcl
GetBucketAnalyticsConfiguration
GetBucketCors
GetBucketEncryption
GetBucketIntelligentTieringConfiguration
GetBucketInventoryConfiguration
GetBucketLifecycle
GetBucketLifecycleConfiguration
GetBucketLocation
GetBucketLogging
GetBucketMetricsConfiguration
GetBucketNotification
GetBucketNotificationConfiguration
GetBucketOwnershipControls
GetBucketPolicy
GetBucketPolicyStatus
GetBucketReplication
GetBucketRequestPayment
GetBucketTagging
GetBucketVersioning
GetBucketWebsite
GetObject
GetObjectAcl
GetObjectLegalHold
GetObjectLockConfiguration
GetObjectRetention
GetObjectTagging
GetObjectTorrent
GetPublicAccessBlock
HeadBucket
HeadObject
ListBucketAnalyticsConfigurations
ListBucketIntelligentTieringConfigurations
ListBucketInventoryConfigurations
ListBucketMetricsConfigurations
ListBuckets
ListMultipartUploads
ListObjects
ListObjectsV2
ListObjectVersions
ListParts
PutBucketAccelerateConfiguration
PutBucketAcl
PutBucketAnalyticsConfiguration
PutBucketCors
PutBucketEncryption
PutBucketIntelligentTieringConfiguration
PutBucketInventoryConfiguration
PutBucketLifecycle
PutBucketLifecycleConfiguration
PutBucketLogging
PutBucketMetricsConfiguration
PutBucketNotification
PutBucketNotificationConfiguration
PutBucketOwnershipControls
PutBucketPolicy
PutBucketReplication
PutBucketRequestPayment
PutBucketTagging
PutBucketVersioning
PutBucketWebsite
PutObject
PutObjectAcl
PutObjectLegalHold
PutObjectLockConfiguration
PutObjectRetention
PutObjectTagging
PutPublicAccessBlock
RestoreObject
SelectObjectContent
UploadPart
UploadPartCopy
WriteGetObjectResponse

DATA TYPES

AbortIncompleteMultipartUpload
AccelerateConfiguration
AccessControlPolicy
AccessControlTranslation
AnalyticsAndOperator
AnalyticsConfiguration
AnalyticsExportDestination
AnalyticsFilter
AnalyticsS3BucketDestination
Bucket
BucketLifecycleConfiguration
BucketLoggingStatus
CloudFunctionConfiguration
CommonPrefix
CompletedMultipartUpload
CompletedPart
Condition
ContinuationEvent
CopyObjectResult
CopyPartResult
CORSConfiguration
CORSRule
CreateBucketConfiguration
CSVInput
CSVOutput
DefaultRetention
Delete
DeletedObject
DeleteMarkerEntry
DeleteMarkerReplication
Destination
Encryption
EncryptionConfiguration
EndEvent
Error
ErrorDocument
ExistingObjectReplication
FilterRule
GlacierJobParameters
Grant
Grantee
IndexDocument
Initiator
InputSerialization
IntelligentTieringAndOperator
IntelligentTieringConfiguration
IntelligentTieringFilter
InventoryConfiguration
InventoryDestination
InventoryEncryption
InventoryFilter
InventoryS3BucketDestination
InventorySchedule
JSONInput
JSONOutput
LambdaFunctionConfiguration
LifecycleConfiguration
LifecycleExpiration
LifecycleRule
LifecycleRuleAndOperator
LifecycleRuleFilter
LoggingEnabled
MetadataEntry
Metrics
MetricsAndOperator
MetricsConfiguration
MetricsFilter
MultipartUpload
NoncurrentVersionExpiration
NoncurrentVersionTransition
NotificationConfiguration
NotificationConfigurationDeprecated
NotificationConfigurationFilter
Object
ObjectIdentifier
ObjectLockConfiguration
ObjectLockLegalHold
ObjectLockRetention
ObjectLockRule
ObjectVersion
OutputLocation
OutputSerialization
Owner
OwnershipControls
OwnershipControlsRule
ParquetInput
Part
PolicyStatus
Progress
ProgressEvent
PublicAccessBlockConfiguration
QueueConfiguration
QueueConfigurationDeprecated
RecordsEvent
Redirect
RedirectAllRequestsTo
ReplicaModifications
ReplicationConfiguration
ReplicationRule
ReplicationRuleAndOperator
ReplicationRuleFilter
ReplicationTime
ReplicationTimeValue
RequestPaymentConfiguration
RequestProgress
RestoreRequest
RoutingRule
Rule
S3KeyFilter
S3Location
ScanRange
SelectObjectContentEventStream
SelectParameters
ServerSideEncryptionByDefault
ServerSideEncryptionConfiguration
ServerSideEncryptionRule
SourceSelectionCriteria
SSEKMS
SseKmsEncryptedObjects
SSES3
Stats
StatsEvent
StorageClassAnalysis
StorageClassAnalysisDataExport
Tag
Tagging
TargetGrant
Tiering
TopicConfiguration
TopicConfigurationDeprecated
Transition
VersioningConfiguration
WebsiteConfiguration

By Dasika Madhu Nimeshika

Made By - Dasika Madhu Nimeshika | LinkedIn | Medium
https://dasikamadhu.github.io/AWS-from-A-to-Z/