Amazon S3 and Glacier

Please use the menu below to navigate the article sections:

Hide article menu

Object Lifecycle Management

Encryption

Event Notifications

Object Tags

Amazon S3 CloudWatch Metrics

Cross Region Replication

Same Region replication (SRR)

S3 Analytics

S3 Inventory

Monitoring and Reporting

Logging and Auditing

S3 performance guidelines

Glacier

Amazon S3 is object storage built to store and retrieve any amount of data from anywhere on the Internet.

It's a simple storage service that offers an extremely durable, highly available, and infinitely scalable data storage infrastructure at very low costs.

Amazon S3 is a distributed architecture and objects are redundantly stored on multiple devices across multiple facilities (AZs) in an Amazon S3 region.

Amazon S3 is a simple key-based object store.

Keys can be any string, and they can be constructed to mimic hierarchical attributes.

Alternatively, you can use S3 Object Tagging to organize your data across all your S3 buckets and/or prefixes.

Amazon S3 provides a simple, standards-based REST web services interface that is designed to work with any Internet-development toolkit.

Files can be from 0 bytes to 5TB.

The largest object that can be uploaded in a single PUT is 5 gigabytes.

For objects larger than 100 megabytes use the Multipart Upload capability.

Updates to an object are atomic – when reading an updated object you will either get the new object or the old one, you will never get partial or corrupt data.

There is unlimited storage available.

It is recommended to access S3 through SDKs and APIs (the console uses APIs).

Event notifications for specific actions, can send alerts or trigger actions.

Notifications can be sent to:

- SNS Topics.
- SQS Queue.
- Lambda functions.
- Need to configure SNS/SQS/Lambda before S3.
- No extra charges from S3 but you pay for SNS, SQS and Lambda.

Requester pays function causes the requester to pay (removes anonymous access).

Can provide time-limited access to objects.

Provides read after write consistency for PUTS of new objects.

Provides eventual consistency for overwrite PUTS and DELETES (takes time to propagate).

You can only store files (objects) on S3.

HTTP 200 code indicates a successful write to S3.

S3 data is made up of:

- Key (name).
- Value (data).
- Version ID.
- Metadata.
- Access Control Lists.

Amazon S3 automatically scales to high request rates.

For example, your application can achieve at least 3,500 PUT/POST/DELETE and 5,500 GET requests per second per prefix in a bucket.

There are no limits to the number of prefixes in a bucket.

For read intensive requests you can also use CloudFront edge locations to offload from S3.

Additional Capabilities

Additional capabilities offered by Amazon S3 include:

Additional S3 Capability	How it works
Transfer Acceleration	Speeds up data uploads using CloudFront in reverse
Requester Pays	The requester rather than the bucket owner pays for requests and data transfer
Tags	Assign tags to objects to use in hosting, billing, security etc.
Events	Trigger notifications to SNS, SQS, or Lambda when certain events happen in your bucket
Static Web Hosting	Simple and massively scalable static web hosting

BitTorrent	Use the BitTorrent protocol to retrieve any publicly available object by automatically generating a .torrent file.
Storage Class Analysis	Analyzes storage access patterns to help you decide when to transition the right data to the right storage class.
Storage Lens	Delivers organization-wide visibility into object storage usage, activity trends, and makes actionable recommendations to improve cost-efficiency and apply data protection best practices.
S3 Object Lambda	Add your own code to S3 GET requests to modify and process data as it is returned to an application.

Use Cases

Typical use cases include:

- **Backup and Storage –** Provide data backup and storage services for others.
- **Application Hosting –** Provide services that deploy, install, and manage web applications.
- **Media Hosting –** Build a redundant, scalable, and highly available infrastructure that hosts video, photo, or music uploads and downloads.
- **Software Delivery –** Host your software applications that customers can download.
- **Static Website –** you can configure a static website to run from an S3 bucket.

S3 is a persistent, highly durable data store.

Persistent data stores are non-volatile storage systems that retain data when powered off.

This contrasts with transient data stores and ephemeral data stores which lose the data when powered off.

The following table provides a description of persistent, transient, and ephemeral data stores and which AWS service to use:

Storage Type	Description	Examples
Persistent data store	Data is durable and sticks around after reboots, restarts, or power cycles	S3, Glacier, EBS, EFS
Transient Data Store	Data is just temporarily stored and passed along to another process or persistent store	SQS, SNS

Ephemeral Data Store	Data is lost when the system is stopped	EC2 Instance Store, Memcached (Elasticache)

Buckets

Files are stored in buckets:

- A bucket can be viewed as a container for objects.
- A bucket is a flat container of objects.
- It does not provide a hierarchy of objects.
- You can use an object key name (prefix) to mimic folders.

100 buckets per account by default.

You can store unlimited objects in your buckets.

You can create folders in your buckets (only available through the Console).

You cannot create nested buckets.

Bucket ownership is not transferable.

Bucket names cannot be changed after they have been created.

If a bucket is deleted its name becomes available again.

Bucket names are part of the URL used to access the bucket.

An S3 bucket is region specific.

S3 is a universal namespace so names must be unique globally.

URL is in this format: https://s3-***eu-west-1***.amazonaws.com/***<bucketname>.***

Can backup a bucket to another bucket in another account.

Can enable logging to a bucket.

Bucket naming:

- Bucket names must be at least 3 and no more than 63 characters in length.
- Bucket names must start and end with a lowercase character or a number.

- Bucket names must be a series of one or more labels which are separated by a period.
- Bucket names can contain lowercase letters, numbers, and hyphens.
- Bucket names cannot be formatted as an IP address.

For better performance, lower latency, and lower cost, create the bucket closer to your clients.

Objects

Each object is stored and retrieved by a unique key (ID or name).

An object in S3 is uniquely identified and addressed through:

- Service endpoint.
- Bucket name.
- Object key (name).
- Optionally, an object version.

Objects stored in a bucket will never leave the region in which they are stored unless you move them to another region or enable cross-region replication.

You can define permissions on objects when uploading and at any time afterwards using the AWS Management Console.

Subresources

Sub-resources are subordinate to objects, they do not exist independently but are always associated with another entity such as an object or bucket.

Sub-resources (configuration containers) associated with buckets include:

- Lifecycle – define an object's lifecycle.
- Website – configuration for hosting static websites.
- Versioning – retain multiple versions of objects as they are changed.
- Access Control Lists (ACLs) – control permissions access to the bucket.
- Bucket Policies – control access to the bucket.
- Cross Origin Resource Sharing (CORS).
- Logging.

Sub-resources associated with objects include:

- ACLs – define permissions to access the object.
- Restore – restoring an archive.

Cross-origin-resource-sharing (CORS)

Used to allow requests to a different origin when connected to the main origin.

The request will fail unless the origin allows the requests using CORS headers (e.g. Access-Control-Allow-Origin).

Must enable the correct CORS headers.

Specify a CORS configuration on the S3 bucket.

Storage Classes

There are six S3 storage classes.

- S3 Standard (durable, immediately available, frequently accessed).
- S3 Intelligent-Tiering (automatically moves data to the most cost-effective tier).
- S3 Standard-IA (durable, immediately available, infrequently accessed).
- S3 One Zone-IA (lower cost for infrequently accessed data with less resilience).
- S3 Glacier (archived data, retrieval times in minutes or hours).
- S3 Glacier Deep Archive (lowest cost storage class for long term retention).

The table below provides the details of each Amazon S3 storage class:

	S3 Standard	S3 Intelligent Tiering	S3 Standard-IA	S3 One Zone-IA
Designed for durability	99.999999999% (11 9's)	99.999999999% (11 9's)	99.999999999% (11 9's)	99.999999999% (11 9's)
Designed for availability	99.99%	99.9%	99.9%	99.5%
Availability SLA	99.9%	99%	99%	99%
Availability Zones	≥3	≥3	≥3	1

Minimum capacity charge per object	N/A	N/A	128 KB	128 KB
Minimum storage duration charge	N/A	N/A	30 days	30 days
Retrieval charge	N/A	N/A	per GB retrieved	per GB retrieved
First byte latency	milliseconds	milliseconds	milliseconds	milliseconds
Storage type	Object	Object	Object	Object
Lifecycle transitions	Yes	Yes	Yes	Yes

Objects stored in the S3 One Zone-IA storage class are stored redundantly within a single Availability Zone in the AWS Region you select.

Access and Access Policies

There are four mechanisms for controlling access to Amazon S3 resources:

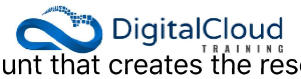
- IAM policies.
- Bucket policies.
- Access Control Lists (ACLs).
- Query string authentication (URL to an Amazon S3 object which is only valid for a limited time).

Access auditing can be configured by configuring an Amazon S3 bucket to create access log records for all requests made against it.

For capturing IAM/user identity information in logs configure AWS CloudTrail Data Events.

By default a bucket, its objects, and related sub-resources are all private.

By default only a resource owner can access a bucket.



The resource owner refers to the AWS account that creates the resource.

With IAM the account owner rather than the IAM user is the owner.

Within an IAM policy you can grant either programmatic access or AWS Management Console access to Amazon S3 resources.

Amazon Resource Names (ARN) are used for specifying resources in a policy.

The format for any resource on AWS is:

arn:partition:service:region:namespace:relative-id.

For S3 resources:

- aws is a common partition name.
- s3 is the service.
- You don't specify Region and namespace.
- For Amazon S3, it can be a bucket-name or a bucket-name/object-key. You can use wild card.

The format for S3 resources is:

- arn:aws:s3:::bucket_name.
- arn:aws:s3:::bucket_name/key_name.

A bucket owner can grant cross-account permissions to another AWS account (or users in an account) to upload objects.

- The AWS account that uploads the objects owns them.
- The bucket owner does not have permissions on objects that other accounts own, however:
  - The bucket owner pays the charges.
  - The bucket owner can deny access to any objects regardless of ownership.
  - The bucket owner can archive any objects or restore archived objects regardless of ownership.

Access to buckets and objects can be granted to:

- Individual users.
- AWS accounts.
- Everyone (public/anonymous).
- All authenticated users (AWS users).

Access policies define access to resources and can be associated with resources (buckets and objects) and users.

You can use the AWS Policy Generator to create a bucket policy for your Amazon S3 bucket.

The categories of policy are resource-based policies and user policies.

Resource-based policies:

- Attached to buckets and objects.
- ACL-based policies define permissions.
- ACLs can be used to grant read/write permissions to other accounts.
- Bucket policies can be used to grant other AWS accounts or IAM users' permission to the bucket and objects.

User policies:

- Can use IAM to manage access to S3 resources.
- Using IAM you can create users, groups and roles and attach access policies to them granting them access to resources.
- You cannot grant anonymous permissions in an IAM user policy as the policy is attached to a user.
- User policies can grant permissions to a bucket and the objects in it.

ACLs:

- S3 ACLs enable you to manage access to buckets and objects.
- Each bucket and object has an ACL attached to it as a subresource.
- Bucket and object permissions are independent of each other.
- The ACL defines which AWS accounts (grantees) or pre-defined S3 groups are granted access and the type of access.
- A grantee can be an AWS account or one of the predefined Amazon S3 groups.
- When you create a bucket or an object, S3 creates a default ACL that grants the resource owner full control over the resource.

Cross account access:

- You grant permission to another AWS account using the email address or the canonical user ID.
- However, if you provide an email address in your grant request, Amazon S3 finds the canonical user ID for that account and adds it to the ACL.
- Grantee accounts can then then delegate the access provided by other accounts to their individual users.

Pre-defined Groups

Authenticated Users group:

- This group represents all AWS accounts.
- Access permission to this group allows any AWS account access to the resource.

- All requests must be signed (authenticated).
- Any authenticated user can access the resource.

All Users group:

- Access permission to this group allows anyone in the world access to the resource.
- The requests can be signed (authenticated) or unsigned (anonymous).
- Unsigned requests omit the authentication header in the request.
- AWS recommends that you never grant the All Users group WRITE, WRITE_ACP, or FULL_CONTROL permissions.

Log Delivery group:

- Providing WRITE permission to this group on a bucket enables S3 to write server access logs.
- Not applicable to objects.

The following table lists the set of permissions that Amazon S3 supports in an ACL.

- The set of ACL permissions is the same for an object ACL and a bucket ACL.
- Depending on the context (bucket ACL or object ACL), these ACL permissions grant permissions for specific buckets or object operations.
- The table lists the permissions and describes what they mean in the context of objects and buckets.

Permission	When granted on a bucket	When granted on an object
READ	Allows grantee to list the objects in the bucket	Allows grantee to read the object data and its metadata
WRITE	Allows grantee to create, overwrite and delete any object in the bucket	N/A
READ_ACP	Allows grantee to read the bucket ACL	Allows grantee to read the object ACL
WRITE_ACP	Allows grantee to write the ACL for the applicable buckets	Allows grantee to write the ACL for the applicable object
FULL_CONTROL	Allows grantee the READ, WRITE, READ_ACP, WRITE_ACP permissions on the bucket	Allows grantee the READ, WRITE, READ_ACP, WRITE_ACP permissions on the object

Note the following:

- Permissions are assigned at the account level for authenticated users.
- You cannot assign permissions to individual IAM users.
- When Read is granted on a bucket it only provides the ability to list the objects in the bucket.
- When Read is granted on an object the data can be read.
- ACP means access control permissions and READ_ACP/WRITE_ACP control who can read/write the ACLs themselves.
- WRITE is only applicable to the bucket level (except for ACP).

Bucket policies are limited to 20 KB in size.

Object ACLs are limited to 100 granted permissions per ACL.

The only recommended use case for the bucket ACL is to grant write permissions to the S3 Log Delivery group.

There are limits to managing permissions using ACLs:

- You cannot grant permissions to individual users.
- You cannot grant conditional permissions.
- You cannot explicitly deny access.

When granting other AWS accounts the permissions to upload objects, permissions to these objects can only be managed by the object owner using object ACLs.

You can use bucket policies for:

- Granting users permissions to a bucket owned by your account.
- Managing object permissions (where the object owner is the same account as the bucket owner).
- Managing cross-account permissions for all Amazon S3 permissions.

You can use user policies for:

- Granting permissions for all Amazon S3 operations.
- Managing permissions for users in your account.
- Granting object permissions to users within the account.

For an IAM user to access resources in another account the following must be provided:

- Permission from the parent account through a user policy.
- Permission from the resource owner to the IAM user through a bucket policy, or the parent account through a bucket policy, bucket ACL or object ACL.

If an AWS account owns a resource it can grant permissions to another account, that account can then delegate those permissions or a subset of them to users in the account (permissions delegation).

An account that receives permissions from another account cannot delegate permissions cross-account to a third AWS account.

Charges

No charge for data transferred between EC2 and S3 in the same region.

Data transfer into S3 is free of charge.

Data transferred to other regions is charged.

Data Retrieval (applies to S3 Standard-IA and S3 One Zone-IA, S3 Glacier and S3 Glacier Deep Archive).

Charges are:

- Per GB/month storage fee.
- Data transfer out of S3.
- Upload requests (PUT and GET).
- Retrieval requests (S3-IA or Glacier).

Requester pays:

- The bucket owner will only pay for object storage fees.
- The requester will pay for requests (uploads/downloads) and data transfers.
- Can only be enabled at the bucket level.

Multipart upload

Can be used to speed up uploads to S3.

Multipart upload uploads objects in parts independently, in parallel and in any order.

Performed using the S3 Multipart upload API.

It is recommended for objects of 100MB or larger.

- Can be used for objects from 5MB up to 5TB.
- Must be used for objects larger than 5GB.

If transmission of any part fails it can be retransmitted.

Improves throughput.

Can pause and resume object uploads.

Can begin upload before you know the final object size.

S3 Copy

You can create a copy of objects up to 5GB in size in a single atomic operation.

For files larger than 5GB you must use the multipart upload API.

Can be performed using the AWS SDKs or REST API.

The copy operation can be used to:

- Generate additional copies of objects.
- Renaming objects.
- Changing the copy's storage class or encryption at rest status.
- Move objects across AWS locations/regions.
- Change object metadata.

Once uploaded to S3 some object metadata cannot be changed, copying the object can allow you to modify this information.

Transfer acceleration

Amazon S3 Transfer Acceleration enables fast, easy, and secure transfers of files over long distances between your client and your Amazon S3 bucket.

S3 Transfer Acceleration leverages Amazon CloudFront's globally distributed AWS Edge Locations.

Used to accelerate object uploads to S3 over long distances (latency).

Transfer acceleration is as secure as a direct upload to S3.

You are charged only if there was a benefit in transfer times.

Need to enable transfer acceleration on the S3 bucket.

Cannot be disabled, can only be suspended.

May take up to 30 minutes to implement.

URL is: <bucketname>.s3-accelerate.amazonaws.com.

Bucket names must be DNS compliance and cannot have periods between labels.

Now HIPAA compliant.

You can use multipart uploads with transfer acceleration.

Must use one of the following endpoints:

- .s3-accelerate.amazonaws.com.
- .s3-accelerate.dualstack.amazonaws.com (dual-stack option).

S3 Transfer Acceleration supports all bucket level features including multipart uploads.

Static Websites

S3 can be used to host static websites.

Cannot use dynamic content such as PHP, .Net etc.

Automatically scales.

You can use a custom domain name with S3 using a Route 53 Alias record.

When using a custom domain name the bucket name must be the same as the domain name.

Can enable redirection for the whole domain, pages, or specific objects.

URL is: <bucketname>.s3-website-.amazonaws.com.

Requester pays does not work with website endpoints.

Does not support HTTPS/SSL.

Returns an HTML document.

Supports object and bucket level redirects.

Only supports GET and HEAD requests on objects.

Supports publicly readable content only.

To enable website hosting on a bucket, specify:

- An Index document (default web page).
- Error document (optional).

Key Difference	REST API Endpoint	Website Endpoint
Access Control	Supports both public and private content	Supports only publicly readable content
Error message handling	Returns an XML-formatted error response	Returns an HTML document
Redirection support	Not applicable	Supports both object-level and bucket-level redirects
Requests support	Supports all bucket and object operations	Supports only GET and HEAD requests on objects
Responses to GET and HEAD requests at the root of the bucket	Returns a list of the object keys in the bucket	Returns the Index document that is specified in the website configuration
SSL support	Supports SSL connections	Does not support SSL connections

Pre-Signed URLs

Pre-signed URLs can be used to provide temporary access to a specific object to those who do not have AWS credentials.

By default all objects are private and can only be accessed by the owner.

To share an object you can either make it public or generate a pre-signed URL.

Expiration date and time must be configured.

These can be generated using SDKs for Java and .Net and AWS explorer for Visual Studio.

Can be used for downloading and uploading S3 objects.

Versioning

Versioning stores all versions of an object (including all writes and even if an object is deleted).

Versioning protects against accidental object/data deletion or overwrites.

Enables “roll-back” and “un-delete” capabilities.

Versioning can also be used for data retention and archive.

Old versions count as billable size until they are permanently deleted.

Enabling versioning does not replicate existing objects.

Can be used for backup.

Once enabled versioning cannot be disabled only suspended.

Can be integrated with lifecycle rules.

Multi-factor authentication (MFA) delete can be enabled.

MFA delete can also be applied to changing versioning settings.

MFA delete applies to:

- Changing the bucket’s versioning state.
- Permanently deleting an object.

Cross Region Replication requires versioning to be enabled on the source and destination buckets.

Reverting to previous versions isn’t replicated.

By default a HTTP GET retrieves the most recent version.

Only the S3 bucket owner can permanently delete objects once versioning is enabled.

When you try to delete an object with versioning enabled a DELETE marker is placed on the object.

You can delete the DELETE marker and the object will be available again.

Deletion with versioning replicates the delete marker. But deleting the delete marker is not replicated.

Bucket versioning states:

- Enabled.
- Versioned.
- Un-versioned.

Objects that existed before enabling versioning will have a version ID of NULL.

Suspension:

- If you suspend versioning the existing objects remain as they are however new versions will not be created.
- While versioning is suspended new objects will have a version ID of NULL and uploaded objects of the same name will overwrite the existing object.

Object Lifecycle Management

Used to optimize storage costs, adhere to data retention policies and to keep S3 volumes well-maintained.

A *lifecycle configuration* is a set of rules that define actions that Amazon S3 applies to a group of objects. There are two types of actions:

- **Transition actions—**Define when objects transition to another storage class. For example, you might choose to transition objects to the STANDARD_IA storage class 30 days after you created them, or archive objects to the GLACIER storage class one year after creating them.

There are costs associated with the lifecycle transition requests. For pricing information, see Amazon S3 Pricing.

- **Expiration actions—**Define when objects expire. Amazon S3 deletes expired objects on your behalf.

Lifecycle configuration is an XML file applied at the bucket level as a subresource.

Can be used in conjunction with versioning or independently.

Can be applied to current and previous versions.

Can be applied to specific objects within a bucket: objects with a specific tag or objects with a specific prefix.

Supported Transitions and Related Constraints

Amazon S3 supports the following lifecycle transitions between storage classes using a lifecycle configuration:

- You can transition from the STANDARD storage class to any other storage class.

- You can transition from any storage class to the GLACIER or DEEP_ARCHIVE storage classes.
- You can transition from the STANDARD_IA storage class to the INTELLIGENT_TIERING or ONEZONE_IA storage classes.
- You can transition from the INTELLIGENT_TIERING storage class to the ONEZONE_IA storage class.
- You can transition from the GLACIER storage class to the DEEP_ARCHIVE storage class.

The following lifecycle transitions are not supported:

- You can't transition from any storage class to the STANDARD storage class.
- You can't transition from any storage class to the REDUCED_REDUNDANCY storage class.
- You can't transition from the INTELLIGENT_TIERING storage class to the STANDARD_IA storage class.
- You can't transition from the ONEZONE_IA storage class to the STANDARD_IA or INTELLIGENT_TIERING storage classes.
- You can transition from the GLACIER storage class to the DEEP_ARCHIVE storage class only.
- You can't transition from the DEEP_ARCHIVE storage class to any other storage class.

The lifecycle storage class transitions have the following constraints:

- From the STANDARD or STANDARD_IA storage class to INTELLIGENT_TIERING. The following constraints apply:
  - For larger objects, there is a cost benefit for transitioning to INTELLIGENT_TIERING. Amazon S3 does not transition objects that are smaller than 128 KB to the INTELLIGENT_TIERING storage class because it's not cost effective.
- From the STANDARD storage classes to STANDARD_IA or ONEZONE_IA. The following constraints apply:
  - For larger objects, there is a cost benefit for transitioning to STANDARD_IA or ONEZONE_IA. Amazon S3 does not transition objects that are smaller than 128 KB to the STANDARD_IA or ONEZONE_IA storage classes because it's not cost effective.
  - Objects must be stored at least 30 days in the current storage class before you can transition them to STANDARD_IA or ONEZONE_IA. For example, you cannot create a lifecycle rule to transition objects to the STANDARD_IA storage class one day after you create them.
  - Amazon S3 doesn't transition objects within the first 30 days because newer objects are often accessed more frequently or deleted sooner than is suitable for STANDARD_IA or ONEZONE_IA storage.
  - If you are transitioning noncurrent objects (in versioned buckets), you can transition only objects that are at least 30 days noncurrent to STANDARD_IA or ONEZONE_IA storage.
- From the STANDARD_IA storage class to ONEZONE_IA. The following constraints apply:

Objects must be stored at least 30 days in the STANDARD_IA storage class before you can transition them to the ONEZONE_IA class.

Encryption

You can securely upload/download your data to Amazon S3 via SSL endpoints using the HTTPS protocol (In Transit – SSL/TLS).

Encryption options:

Encryption Option	How It Works
SSE-S3	Use S3's existing encryption key for AES-256
SSE-C	Upload your own AES-256 encryption key which S3 uses when it writes objects
SSE-KMS	Use a key generated and managed by AWS KMS
Client Side	Encrypt objects using your own local encryption process before uploading to S3

Server-side encryption options

Server-side encryption protects data at rest.

Amazon S3 encrypts each object with a unique key.

As an additional safeguard, it encrypts the key itself with a master key that it rotates regularly.

Amazon S3 server-side encryption uses one of the strongest block ciphers available to encrypt your data, 256-bit Advanced Encryption Standard (AES-256).

If you need server-side encryption for all the objects that are stored in a bucket, use a bucket policy.

To request server-side encryption using the object creation REST APIs, provide the x-amz-server-side-encryption request header.

Note: You need the kms:Decrypt permission when you upload or download an Amazon S3 object encrypted with an AWS Key Management Service (AWS KMS) customer master key (CMK), and that is in addition to kms:ReEncrypt, kms:GenerateDataKey, and kms:DescribeKey permissions.

There are three options for using server-side encryption: SSE-S3, SSE-KMS and SSE-C. These are detailed below,

SSE-S3 – Server-Side Encryption with S3 managed keys

When you use Server-Side Encryption with Amazon S3-Managed Keys (SSE-S3), each object is encrypted with a unique key.

As an additional safeguard, it encrypts the key itself with a master key that it regularly rotates.

Amazon S3 server-side encryption uses one of the strongest block ciphers available, 256-bit Advanced Encryption Standard (AES-256), to encrypt your data.

- Each object is encrypted with a unique key.
- Encryption key is encrypted with a master key.
- AWS regularly rotate the master key.
- Uses AES 256.

SSE-KMS – Server-Side Encryption with AWS KMS keys

Server-Side Encryption with Customer Master Keys (CMKs) Stored in AWS Key Management Service (SSE-KMS) is like SSE-S3, but with some additional benefits and charges for using this service.

There are separate permissions for the use of a CMK that provides added protection against unauthorized access of your objects in Amazon S3.

SSE-KMS also provides you with an audit trail that shows when your CMK was used and by whom.

Additionally, you can create and manage customer managed CMKs or use AWS managed CMKs that are unique to you, your service, and your Region.

- KMS uses Customer Master Keys (CMKs) to encrypt.
- Can use the automatically created CMK key.
- OR you can select your own key (gives you control for management of keys).
- An envelope key protects your keys.
- Chargeable.

SSE-C – Server-Side Encryption with client provided keys

With Server-Side Encryption with Customer-Provided Keys (SSE-C), you manage the encryption keys and Amazon S3 manages the encryption, as it writes to disks, and decryption, when you access your objects.

- Client manages the keys, S3 manages encryption.
- AWS does not store the encryption keys.
- If keys are lost data cannot be decrypted.

When using server-side encryption with customer-provided encryption keys (SSE-C), you must provide encryption key information using the following request headers:

x-amz-server-side-encryption-customer-algorithm – Use this header to specify the encryption algorithm. The header value must be "AES256".

x-amz-server-side-encryption-customer-key – Use this header to provide the 256-bit, base64-encoded encryption key for Amazon S3 to use to encrypt or decrypt your data.

x-amz-server-side-encryption-customer-key-MD5 – Use this header to provide the base64-encoded 128-bit MD5 digest of the encryption key according to RFC 1321. Amazon S3 uses this header for a message integrity check to ensure that the encryption key was transmitted without error.

Client-side encryption

This is the act of encrypting data before sending it to Amazon S3.

To enable client-side encryption, you have the following options:

1. Use a customer master key (CMK) stored in AWS Key Management Service (AWS KMS).
2. Use a master key you store within your application.

Option 1. Use a customer master key (CMK) stored in AWS Key Management Service (AWS KMS)

When uploading an object—Using the customer master key (CMK) ID, the client first sends a request to AWS KMS for a CMK that it can use to encrypt your object data. AWS KMS returns two versions of a randomly generated data key:

- A plaintext version of the data key that the client uses to encrypt the object data.
- A cipher blob of the same data key that the client uploads to Amazon S3 as object metadata.

When downloading an object—The client downloads the encrypted object from Amazon S3 along with the cipher blob version of the data key stored as object metadata. The client then sends the cipher blob to AWS KMS to get the plaintext version of the data key so that it can decrypt the object data.

Option 2. Use a master key you store within your application

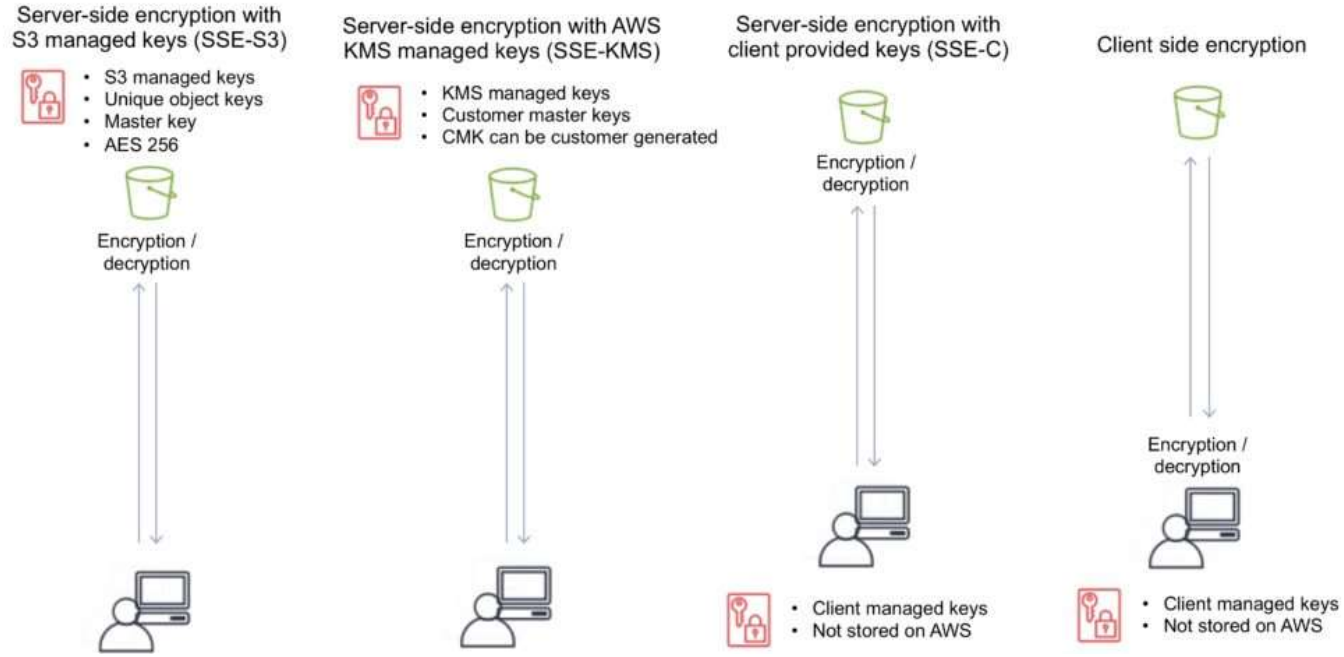
When uploading an object—You provide a client-side master key to the Amazon S3 encryption client. The client uses the master key only to encrypt the data encryption key that it generates randomly. The process works like this:

1. The Amazon S3 encryption client generates a one-time-use symmetric key (also known as a data encryption key or data key) locally. It uses the data key to encrypt the data of a single Amazon S3 object. The client generates a separate data key for each object.
2. The client encrypts the data encryption key using the master key that you provide. The client uploads the encrypted data key and its material description as part of the object metadata. The client uses the material description to determine which client-side master key to use for decryption.

3. The client uploads the encrypted data to Amazon S3 and saves the encrypted data key as object metadata (x-amz-meta-x-amz-key) in Amazon S3.

When downloading an object—The client downloads the encrypted object from Amazon S3. Using the material description from the object's metadata, the client determines which master key to use to decrypt the data key. The client uses that master key to decrypt the data key and then uses the data key to decrypt the object.

The following diagram depicts the options for enabling encryption and shows you where the encryption is applied and where the keys are managed:



Event Notifications

Amazon S3 event notifications can be sent in response to actions in Amazon S3 like PUTs, POSTs, COPYs, or DELETEs.

Amazon S3 event notifications enable you to run workflows, send alerts, or perform other actions in response to changes in your objects stored in S3.

To enable notifications, you must first add a notification configuration that identifies the events you want Amazon S3 to publish and the destinations where you want Amazon S3 to send the notifications.

You can configure notifications to be filtered by the prefix and suffix of the key name of objects.

Amazon S3 can publish notifications for the following events:

- New object created events.
- Object removal events.
- Restore object events.
- Reduced Redundancy Storage (RRS) object lost events.
- Replication events.

Amazon S3 can send event notification messages to the following destinations:

- Publish event messages to an Amazon Simple Notification Service (Amazon SNS) topic.
- Publish event messages to an Amazon Simple Queue Service (Amazon SQS) queue.
- Publish event messages to AWS Lambda by invoking a Lambda function and providing the event message as an argument.

Need to grant Amazon S3 permissions to post messages to an Amazon SNS topic or an Amazon SQS queue.

Need to also grant Amazon S3 permission to invoke an AWS Lambda function on your behalf. For information about granting these permissions.

Object Tags

S3 object tags are key-value pairs applied to S3 objects which can be created, updated, or deleted at any time during the lifetime of the object.

Allow you to create Identity and Access Management (IAM) policies, setup S3 Lifecycle policies, and customize storage metrics.

Up to ten tags can be added to each S3 object and you can use either the AWS Management Console, the REST API, the AWS CLI, or the AWS SDKs to add object tags.

Amazon S3 CloudWatch Metrics

You can use the AWS Management Console to enable the generation of 1-minute CloudWatch request metrics for your S3 bucket or configure filters for the metrics using a prefix or object tag.

Alternatively, you can call the S3 PUT Bucket Metrics API to enable and configure publication of S3 storage metrics.

CloudWatch Request Metrics will be available in CloudWatch within 15 minutes after they are enabled.

CloudWatch Storage Metrics are enabled by default for all buckets and reported once per day.

The S3 metrics that can be monitored include:

- S3 requests.
- Bucket storage.
- Bucket size.
- All requests.
- HTTP 4XX/5XX errors.

Cross Region Replication

CRR is an Amazon S3 feature that automatically replicates data across AWS Regions.

With CRR, every object uploaded to an S3 bucket is automatically replicated to a destination bucket in a different AWS Region that you choose.

Provides automatic, asynchronous copying of objects between buckets in different regions.

CRR is configured at the S3 bucket level.

You enable a CRR configuration on your source bucket by specifying a destination bucket in a different Region for replication.

You can use either the AWS Management Console, the REST API, the AWS CLI, or the AWS SDKs to enable CRR.

Versioning must be enabled for both the source and destination buckets .

Source and destination buckets must be in different regions.

With CRR you can only replication between regions, not within a region (see SRR below for single region replication).

Replication is 1:1 (one source bucket, to one destination bucket).

You can configure separate S3 Lifecycle rules on the source and destination buckets.

You can replicate KMS-encrypted objects by providing a destination KMS key in your replication configuration.

You can set up CRR across AWS accounts to store your replicated data in a different account in the target region.

Provides low latency access for data by copying objects to buckets that are closer to users.

To activate CRR you need to configure the replication on the source bucket:

- Define the bucket in the other region to replicate to.
- Specify to replicate all objects or a subset of objects with specific key name prefixes.

The replicas will be exact replicas and share the same key names and metadata.

You can specify a different storage class (by default the source storage class will be used).

AWS S3 will encrypt data in-transit with SSL.

AWS S3 must have permission to replicate objects.

Bucket owners must have permission to read the object and object ACL.

Can be used across accounts but the source bucket owner must have permission to replicate objects into the destination bucket.

Triggers for replication are:

- Uploading objects to the source bucket.
- DELETE of objects in the source bucket.
- Changes to the object, its metadata, or ACL.

What is replicated:

- New objects created after enabling replication.
- Changes to objects.
- Objects created using SSE-S3 using the AWS managed key.
- Object ACL updates.

What isn't replicated:

- Objects that existed before enabling replication (can use the copy API).
- Objects created with SSE-C and SSE-KMS.
- Objects to which the bucket owner does not have permissions.
- Updates to bucket-level subresources.
- Actions from lifecycle rules are not replicated.
- Objects in the source bucket that are replicated from another region are not replicated.

Deletion behavior:

- If a DELETE request is made without specifying an object version ID a delete marker will be added and replicated.
- If a DELETE request is made specifying an object version ID the object is deleted but the delete marker is not replicated.

Charges:

- Requests for upload.
- Inter-region transfer.
- S3 storage in both regions.

Same Region replication (SRR)

As the name implies you can use SRR to replication objects to a destination bucket within the same region as the source bucket.

This feature was released in September 2018.

Replication is automatic and asynchronous.

New objects uploaded to an Amazon S3 bucket are configured for replication at the bucket, prefix, or object tag levels.

Replicated objects can be owned by the same AWS account as the original copy or by different accounts, to protect from accidental deletion.

Replication can be to any Amazon S3 storage class, including S3 Glacier and S3 Glacier Deep Archive to create backups and long-term archives.

When an S3 object is replicated using SRR, the metadata, Access Control Lists (ACL), and object tags associated with the object are also part of the replication.

Once SRR is configured on a source bucket, any changes to the object, metadata, ACLs, or object tags trigger a new replication to the destination bucket.

S3 Analytics

Can run analytics on data stored on Amazon S3.

This includes data lakes, IoT streaming data, machine learning, and artificial intelligence.

The following strategies can be used:

S3 Analytics Strategies	Service Used
Data Lake Concept	Athena, Redshift Spectrum, QuickSight
IoT Streaming Data Repository	Kinesis Firehose
ML and AI Storage	Rekognition, Lex, MXNet
Storage Class Analysis	S3 Management Analytics

S3 Inventory

You can use S3 Inventory to audit and report on the replication and encryption status of your objects for business, compliance, and regulatory needs.

Amazon S3 inventory provides comma-separated values (CSV), Apache optimized row columnar (ORC) or Apache Parquet (Parquet) output files that list your objects and their corresponding metadata on a daily or weekly basis for an S3 bucket or a shared prefix (that is, objects that have names that begin with a common string).

Monitoring and Reporting

Amazon CloudWatch metrics for Amazon S3 can help you understand and improve the performance of applications that use Amazon S3. There are several ways that you can use CloudWatch with Amazon S3.

- **Daily storage metrics for buckets -** Monitor bucket storage using CloudWatch, which collects and processes storage data from Amazon S3 into readable, daily metrics. These storage metrics for Amazon S3 are reported once per day and are provided to all customers at no additional cost.
- **Request metrics -** Monitor Amazon S3 requests to quickly identify and act on operational issues. The metrics are available at 1-minute intervals after some latency to process. These CloudWatch metrics are billed at the same rate as the Amazon CloudWatch custom metrics.
- **Replication metrics -** Monitor the total number of S3 API operations that are pending replication, the total size of objects pending replication, and the maximum replication time to the destination Region. Only replication rules that have S3 Replication Time Control (S3 RTC) enabled will publish replication metrics.

Logging and Auditing

You can record the actions that are taken by users, roles, or AWS services on Amazon S3 resources and maintain log records for auditing and compliance purposes.

To do this, you can use Amazon S3 server access logging, AWS CloudTrail logs, or a combination of both.

AWS recommend that you use AWS CloudTrail for logging bucket and object-level actions for your Amazon S3 resources.

Server access logging provides detailed records for the requests that are made to a bucket.

This information can be used for auditing.

You must not set the bucket being logged to be the destination for the logs as this creates a logging loop and the bucket will grow exponentially.

S3 performance guidelines

AWS provide some performance guidelines for Amazon S3. These are summarized here:

Measure Performance – When optimizing performance, look at network throughput, CPU, and DRAM requirements. Depending on the mix of demands for these different resources, it might be worth evaluating different Amazon EC2 instance types.

Scale Storage Connections Horizontally – You can achieve the best performance by issuing multiple concurrent requests to Amazon S3. Spread these requests over separate connections to maximize the accessible bandwidth from Amazon S3.

Use Byte-Range Fetches – Using the Range HTTP header in a GET Object request, you can fetch a byte-range from an object, transferring only the specified portion. You can use concurrent connections to Amazon S3 to fetch different byte ranges from within the same object. This helps you achieve higher aggregate throughput versus a single whole-object request. Fetching smaller ranges of a large object also allows your application to improve retry times when requests are interrupted.

Retry Requests for Latency-Sensitive Applications – Aggressive timeouts and retries help drive consistent latency. Given the large scale of Amazon S3, if the first request is slow, a retried request is likely to take a different path and quickly succeed. The AWS SDKs have configurable timeout and retry values that you can tune to the tolerances of your specific application.

Combine Amazon S3 (Storage) and Amazon EC2 (Compute) in the Same AWS Region – Although S3 bucket names are globally unique, each bucket is stored in a Region that you select when you create the bucket. To optimize performance, we recommend that you access the bucket from Amazon EC2 instances in the same AWS Region when possible. This helps reduce network latency and data transfer costs.

Use Amazon S3 Transfer Acceleration to Minimize Latency Caused by Distance – Amazon S3 Transfer Acceleration manages fast, easy, and secure transfers of files over long geographic distances between the client and an S3 bucket. Transfer Acceleration takes advantage of the globally distributed edge locations in Amazon CloudFront. As the data arrives at an edge location, it is routed to Amazon S3 over an optimized network path. Transfer Acceleration is ideal for transferring gigabytes to terabytes of data regularly across continents. It's also useful for clients that upload to a centralized bucket from all over the world.

Glacier

Glacier is an archiving storage solution for infrequently accessed data.

There are three storage tiers:

S3 Glacier Instant Retrieval

- Data retrieval in milliseconds with the same performance as S3 Standard

- Designed for durability of 99.999999999% of objects across multiple Availability Zones
- Data is resilient in the event of the destruction of one entire Availability Zone
- Designed for 99.9% data availability each year
- 128 KB minimum object size
- Backed with the Amazon S3 Service Level Agreement for availability
- S3 PUT API for direct uploads to S3 Glacier Instant Retrieval, and S3 Lifecycle management for automatic migration of objects

S3 Glacier Flexible Retrieval (Formerly S3 Glacier)

- 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
- Ideal for backup and disaster recovery use cases when large sets of data occasionally need to be retrieved in minutes, without concern for costs
- Configurable retrieval times, from minutes to hours, with free bulk retrievals
- S3 PUT API for direct uploads to S3 Glacier Flexible Retrieval, and S3 Lifecycle management for automatic migration of objects

Amazon S3 Glacier Deep Archive (S3 Glacier Deep Archive)

- Designed for durability of 99.999999999% of objects across multiple Availability Zones
- Lowest cost storage class designed for long-term retention of data that will be retained for 7-10 years
- Ideal alternative to magnetic tape libraries
- Retrieval time within 12 hours
- S3 PUT API for direct uploads to S3 Glacier Deep Archive, and S3 Lifecycle management for automatic migration of objects

The key difference between the top tiers is that Deep Archive is lower cost, but retrieval times are much longer (12 hours).

The S3 Glacier tier has configurable retrieval times from minutes to hours (you pay accordingly).

Archived objects are not available for real time access and you need to submit a retrieval request.

Glacier must complete a job before you can get its output.

Requested archival data is copied to S3 One Zone-IA.

Following retrieval you have 24 hours to download your data.

You cannot specify Glacier as the storage class at the time you create an object.

Glacier is designed to sustain the loss of two facilities.

Glacier automatically encrypts data at rest using AES 256 symmetric keys and supports secure transfer of data over SSL.

Glacier may not be available in all AWS regions.

Glacier objects are visible through S3 only (not Glacier directly).

Glacier does not archive object metadata; you need to maintain a client-side database to maintain this information.

Archives can be 1 byte up to 40TB.

Glacier file archives of 1 byte – 4 GB can be performed in a single operation.

Glacier file archives from 100MB up to 40TB can be uploaded to Glacier using the multipart upload API.

Uploading archives is synchronous.

Downloading archives is asynchronous.

The contents of an archive that has been uploaded cannot be modified.

You can upload data to Glacier using the CLI, SDKs or APIs – you cannot use the AWS Console.

Glacier adds 32-40KB (indexing and archive metadata) to each object when transitioning from other classes using lifecycle policies.

AWS recommends that if you have lots of small objects they are combined in an archive (e.g. zip file) before uploading.

A description can be added to archives, no other metadata can be added.

Glacier archive IDs are added upon upload and are unique for each upload.

Archive retrieval:

- Expedited is 1-5 minutes retrieval (most expensive).
- Standard is 3.5 hours retrieval (cheaper, 10GB data retrieval free per month).
- Bulk retrieval is 5-12 hours (cheapest, use for large quantities of data).

You can retrieve parts of an archive.

When data is retrieved it is copied to S3 and the archive remains in Glacier and the storage class therefore does not change.

AWS SNS can send notifications when retrieval jobs are complete.

Retrieved data is available for 24 hours by default (can be changed).

To retrieve specific objects within an archive you can specify the byte range (Range) in the HTTP GET request (need to maintain a DB of byte ranges).

Glacier Charges:

There is no charge for data transfer between EC2 and Glacier in the same region.

There is a charge if you delete data within 90 days.

When you restore you pay for:

- The Glacier archive.
- The requests.
- The restored data on S3.

Related posts: