**S3(Simple storage service)**

**Properties of S3 :**

* S3 is object based, we can only stores image or files in Bucket(like folder). But we cannot install os or any software.
* S3 is a universal namespace. The name should be unique globally.
* The storage capacity is unlimited.
* S3 objects consists of the following

Key(This is simply the name of the object)

Value(This is simply the data)

Version id (Important for versioning)

Metadata (data about the data we are storing)

Access control list( specific access to specific user)

torrent

* Programmatic access to S3 gives 200 http code when we upload any file or delete like PUTS and DELETE request.
* S3 always follow write before read policy.
* You can do version control

**What are the resource policies we have in S3 ?**

Bucket policy, Bucket ACL and Object ACL.

**When Object ACL is useful ?**

If you grant access to account B to write object into your bucket, then Bucket owner that means you cannot access the object . Object owner has to explicitly grant permission to bucket owner.

This kind of permission is called Object ACL.

**Can we grant OBJECT ACL to individual user ?**

No the permission only at account level. The account that we are referring to know as grantee.

Grantee can a predefined S3 groups as well. The account can be referred as email or canonical user id .

**What is canonical user id ?**

The canonical user ID is a long string, such as 79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be unlike account number which is 12 digit id. Go to a S3 bucket and click ACL and can find the canonical id.

You can use the Amazon S3 ListBuckets API with your IAM user credentials to return the AWS account owner ID, which is the canonical user ID.

**What are the predefined s3 groups ?**

Authenticated Users group : This group represents all AWS accounts

All user group : The group represents all users whether aws or other.

Log delivery group : WRITE permission on a bucket enables this group to write server access logs

**Example of Bucket ACL ?**

Log delivery group is an example of bucket ACL. Like object ACL you cannot define acces for an user, should be a account or a grantee. Cross account access also allowed.

Example of a access control list xml

<?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>display-name</DisplayName>

</Owner>

<AccessControlList>

<Grant>

<Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="CanonicalUser">

<ID>Owner-canonical-user-ID</ID>

<DisplayName>display-name</DisplayName>

</Grantee>

<Permission>FULL\_CONTROL</Permission>

</Grant>

<Grant>

<Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="CanonicalUser">

<ID>user1-canonical-user-ID</ID>

<DisplayName>display-name</DisplayName>

</Grantee>

<Permission>WRITE</Permission>

</Grant>

<Grant>

<Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="CanonicalUser">

<ID>user2-canonical-user-ID</ID>

<DisplayName>display-name</DisplayName>

</Grantee>

<Permission>READ</Permission>

</Grant>

<Grant>

<Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="Group">

<URI>http://acs.amazonaws.com/groups/global/AllUsers</URI>

</Grantee>

<Permission>READ</Permission>

</Grant>

<Grant>

<Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="Group">

<URI>http://acs.amazonaws.com/groups/s3/LogDelivery</URI>

</Grantee>

<Permission>WRITE</Permission>

</Grant>

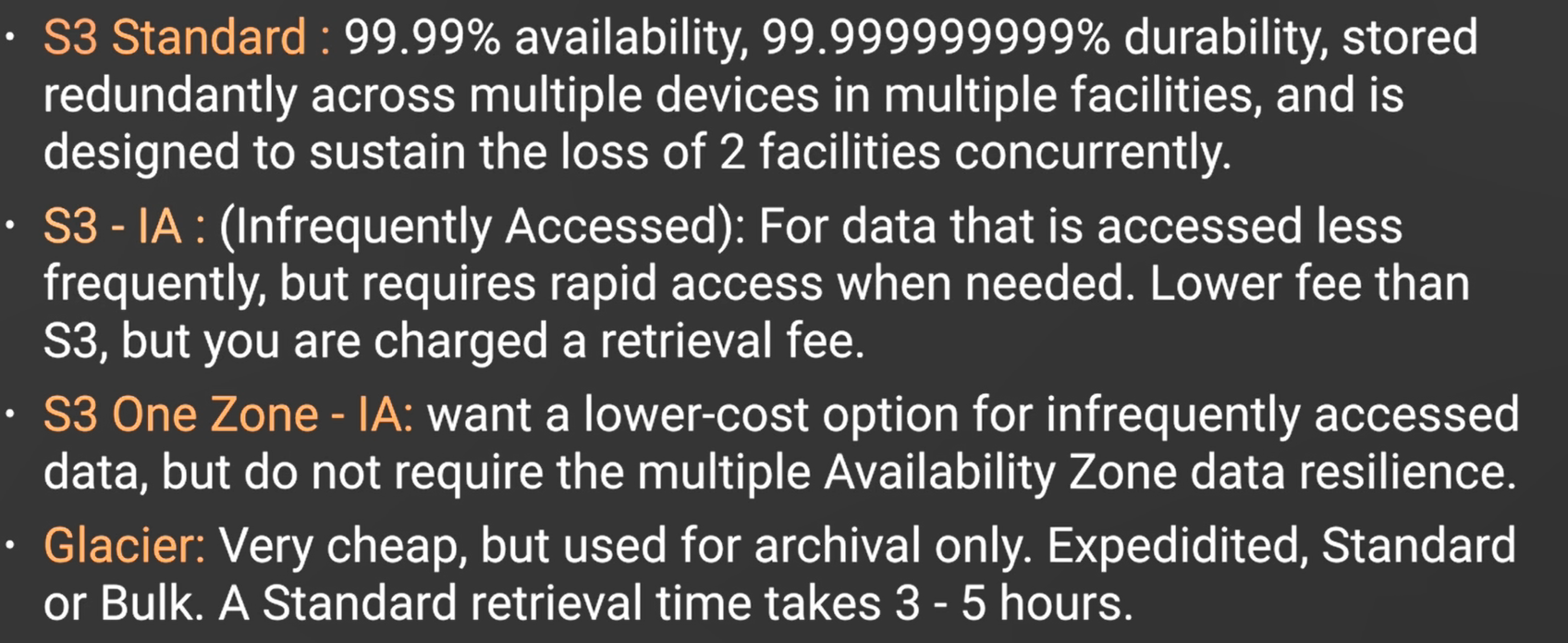
</AccessControlList>

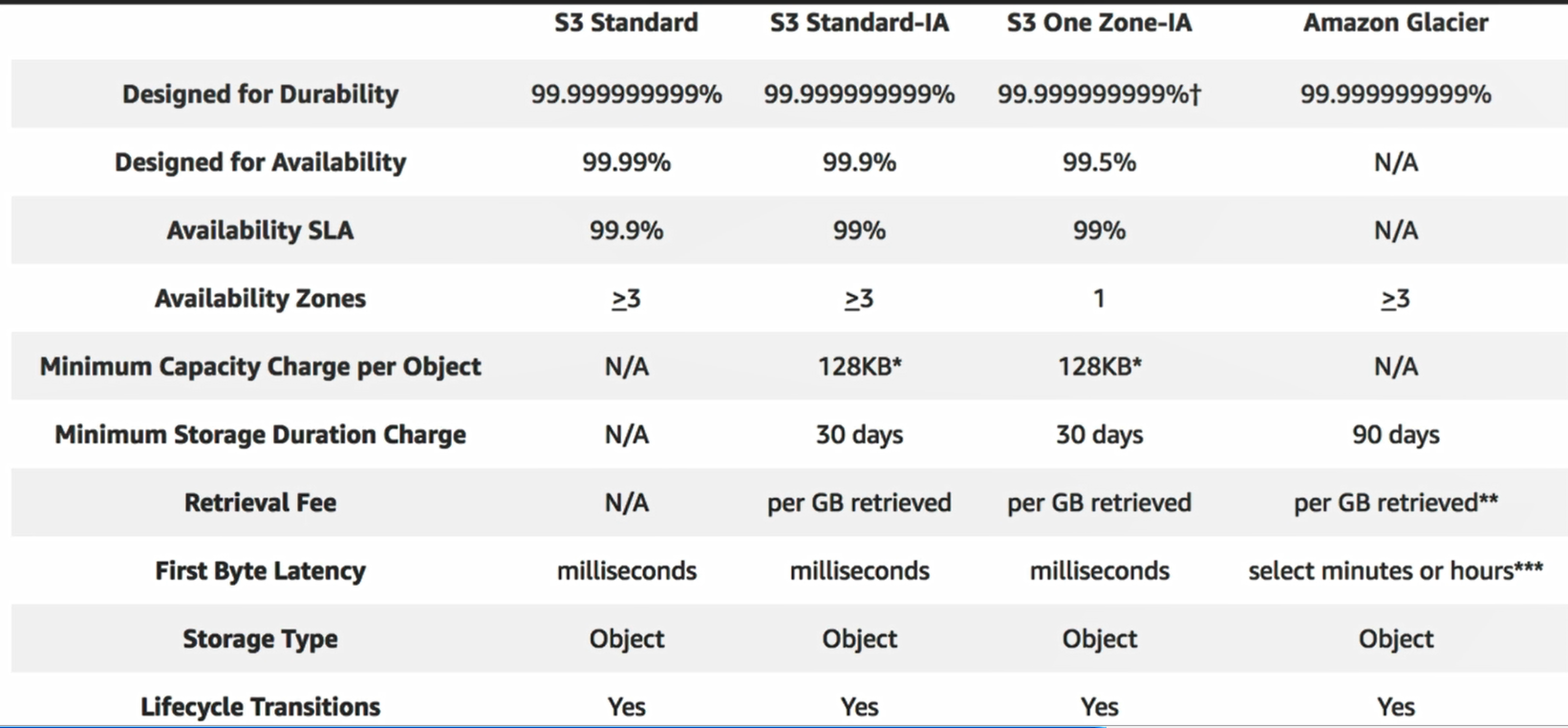
</AccessControlPolicy>

For more infor refere

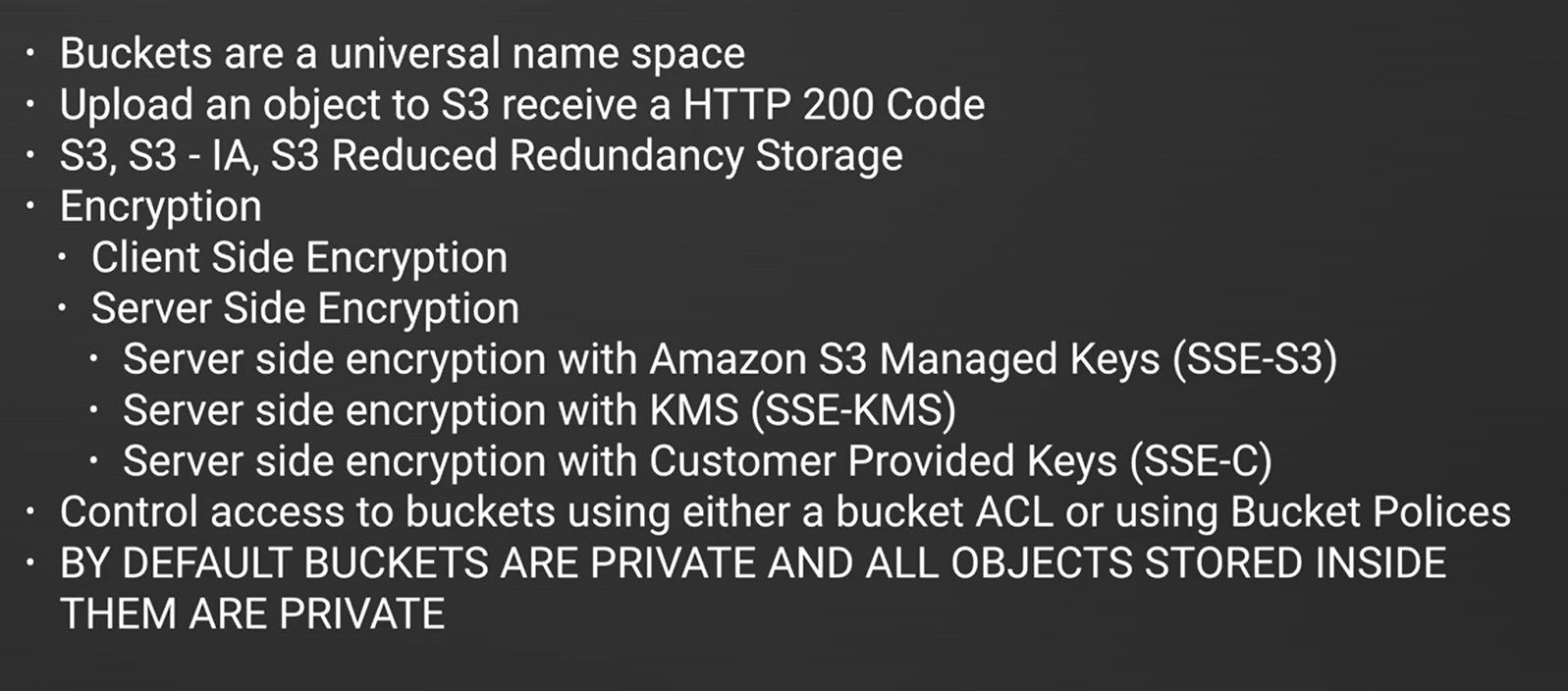
<https://docs.aws.amazon.com/AmazonS3/latest/dev/acl-overview.html>

**S3 storage tiers / classes :**





**Exam tips :**



We can set bucket level as well as object level permission. tags ar not inherited by object but permission is inherited by object.

**Object lifecycle management**

[**https://docs.aws.amazon.com/AmazonS3/latest/dev/object-lifecycle-mgmt.html**](https://docs.aws.amazon.com/AmazonS3/latest/dev/object-lifecycle-mgmt.html)

**How to configure a life cycle ?**

A lifecycle configuration, an XML file, comprises a set of rules with predefined actions that you want Amazon S3 to perform on objects during their lifetime.

Amazon S3 provides a set of API operations for managing lifecycle configuration on a bucket. Amazon S3 stores the configuration as a *lifecycle sub resource* that is attached to your bucket. For details, see the following:

[PUT Bucket lifecycle](https://docs.aws.amazon.com/AmazonS3/latest/API/RESTBucketPUTlifecycle.html)

[GET Bucket lifecycle](https://docs.aws.amazon.com/AmazonS3/latest/API/RESTBucketGETlifecycle.html)

[DELETE Bucket lifecycle](https://docs.aws.amazon.com/AmazonS3/latest/API/RESTBucketDELETElifecycle.html)

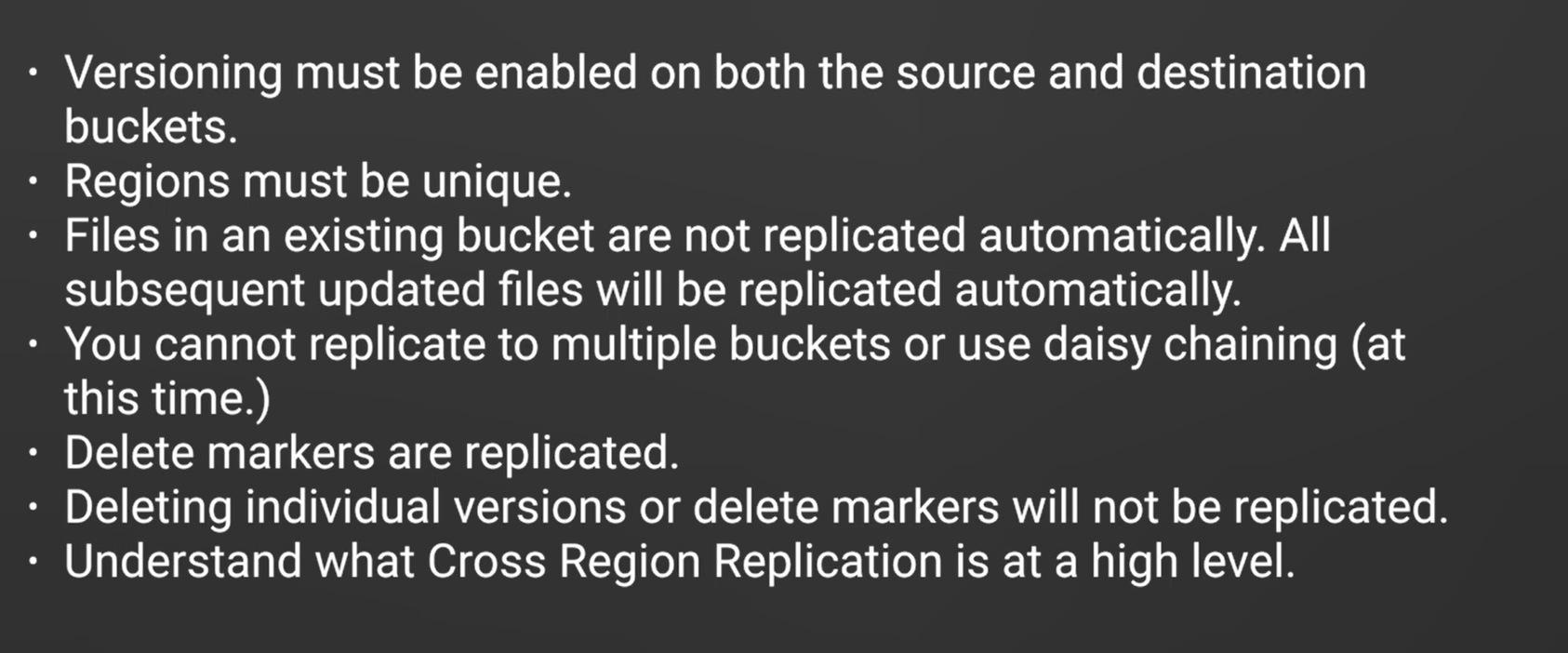
**Versioning control :**

* Version control is applicable at bucket level.
* We can suspend the version once started but cannot stop.
* Each and every bucket has their own versioning
* If versioning is applied on a bucket all the objects has several copies if we upload multiple times. Even if we delete a file the version will be still there unless we delete the version form version control tool.
* Generally versioning is not recommended for large files. For small and essential documents versioning is good option.
* We can integrate with life cycle rules
* We can add MFA delete capability to add one more level of security.

**What is cross region replication ?**

Our buckets are region specific , before creating a bucket we need to select a region. By cross region replication functionality we can apply replication from a source bucket to target bucket.

Any objects, added or updated in a bucket of region 1 will be replicated automatically to the target bucket of region 2 after we create a replication rule. The objects which were added in source bucket are not replicated automatically, we need to copy them with the help of aws client application. Cross region replication follow some rule



Point to be noted, delete markers are replicated, but if we permanently delete a version that will not be deleted in the destination bucket.

**What are not not replicated ?**

* Objects which were added before the replication started.
* Objects encrypted by Customer key.
* You can replicate objects from a source bucket to only one destination bucket.

Suppose that you configure replication where bucket A is the source and bucket B is the destination. Now suppose that you add another replication configuration where bucket B is the source and bucket C is the destination. In this case, objects in bucket B that are replicas of objects in bucket A are not replicated to bucket C.

* if lifecycle configuration is enabled only on your source bucket, Amazon S3 creates delete markers for expired objects but doesn't replicate those markers. If you want the same lifecycle configuration applied to both source and destination buckets, enable the same lifecycle configuration on both.
* Objects that are stored in GLACIER or DEEP\_ARCHIVE storage class.

**What is glacier ?**

Glacier is a low cost storage type used for archiving data. We can store s3 data or directly store the data. Data can be retrieved from glacier valet directly or after restored to s3 bucket.

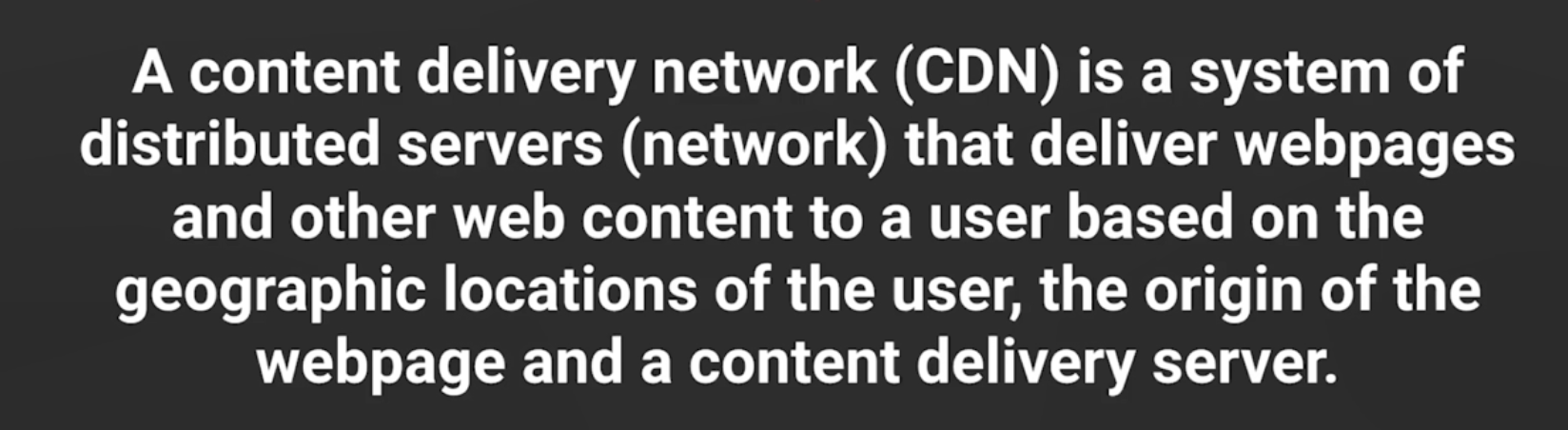
There are 3 data restoration process

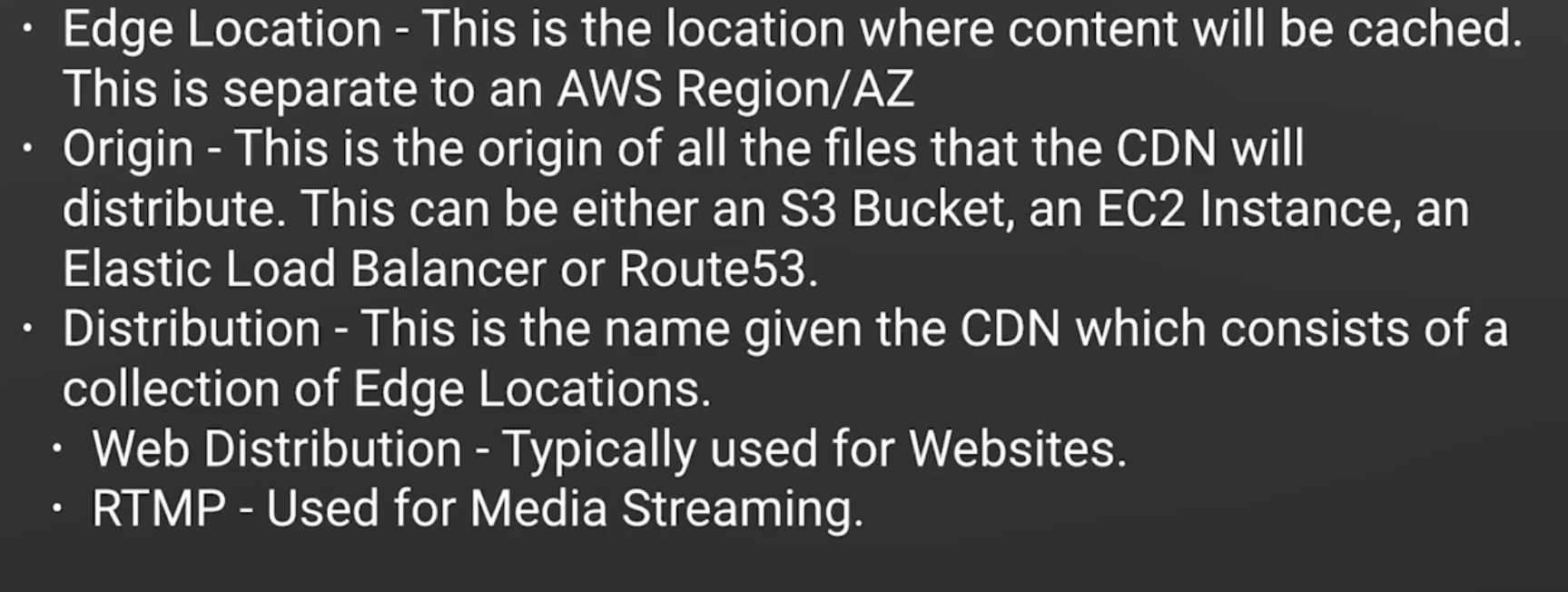
expedite : take 2 to 5 minuets , we can declare the amount of memory to retrieve .

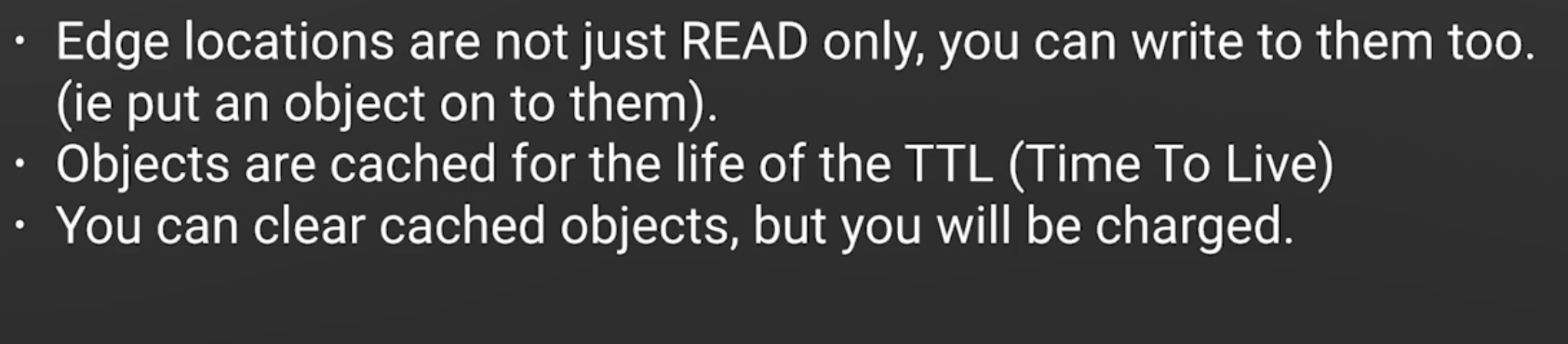
Standard : Takes around 5 hours

Bulk : Lowest cost retrieval , takes 5 to 12 hours but retrieve a huge amount of data in single go.

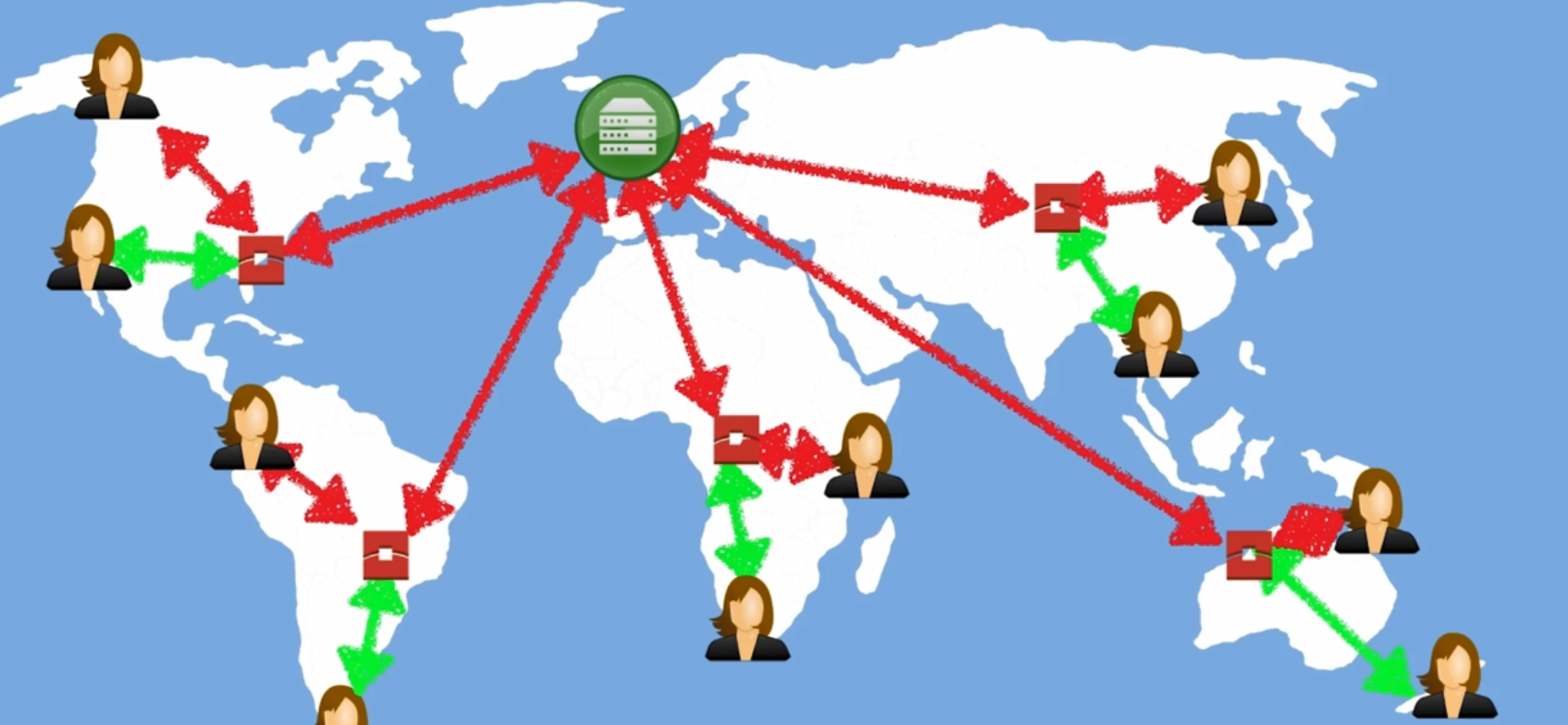
**What is CDN ?**



**Some rules to remember for CDN** 



**How CDN works ?**



If the first user requests any file to aws it will first request to his nearest edge location, if edge location did not find the file it pulls the file for you. So when the second user again query , it will communicate with aws again, as the file is already cached it can deliver the second user upfront with minimum delay. (Edge location is not related to region. means, it is not necessary that each region has one edge location. If any region is small then one edge location can server two regions)

**What is ttl ?**

**Minimum ttl :** The minimum amount of time, in seconds, that you want objects to stay in CloudFront caches before CloudFront forwards another request to your origin to determine whether the object has been updated. Minimum TTL interacts with HTTP headers such as Cache-Control max-age, Cache-Control s-maxage, and Expires and with Default TTL and Maximum TTL.

**Maximum ttl :** The maximum amount of time, in seconds, that you want objects to stay in CloudFront caches before CloudFront forwards another request to your origin to determine whether the object has been updated. The value that you specify applies only when your origin adds HTTP headers such as Cache-Control max-age, Cache-Control s-maxage, and Expires to objects.

**Default ttl :** The default amount of time, in seconds, that you want objects to stay in CloudFront caches before CloudFront forwards another request to your origin to determine whether the object has been updated. The value that you specify applies only when your origin does not add HTTP headers such as Cache-Control max-age, Cache-Control s-maxage, and Expires to objects.

**What is CDN URL?**

Normally for any file in s3 has a simple url like

s3 . {region} . amazonws.com/ {bucketname}/ {prefix name}/{filename}

|  |  |
| --- | --- |
|  | <https://s3.ap-south-1.amazonaws.com/ritbucket123/IMG_0819.JPG> |

prefix name is the folder inside a bucket.

Each CDN creates a unique domain name we can replace the entire url with

{domainname}/{prefix name}/ {filename}

**What is geo restriction ?**

In cdn configuration we can add geographic restriction like

Whitelist: The Countries list contains the countries where you want CloudFront to distribute your content. Blacklist: The Countries list contains the countries where you do not want CloudFront to distribute your content.

If we do not add geo restriction all countries are in whitelist by default.

**CDN is in which level object or bucket ?**

CDN is in bucket level. We can add multiple origin path for a single distribution though, like there can be multiple object path for a single distribution.

**How many origin can be there for one distribution ?**

There can be multiple origin for a single distribution. so origin id should be unique.

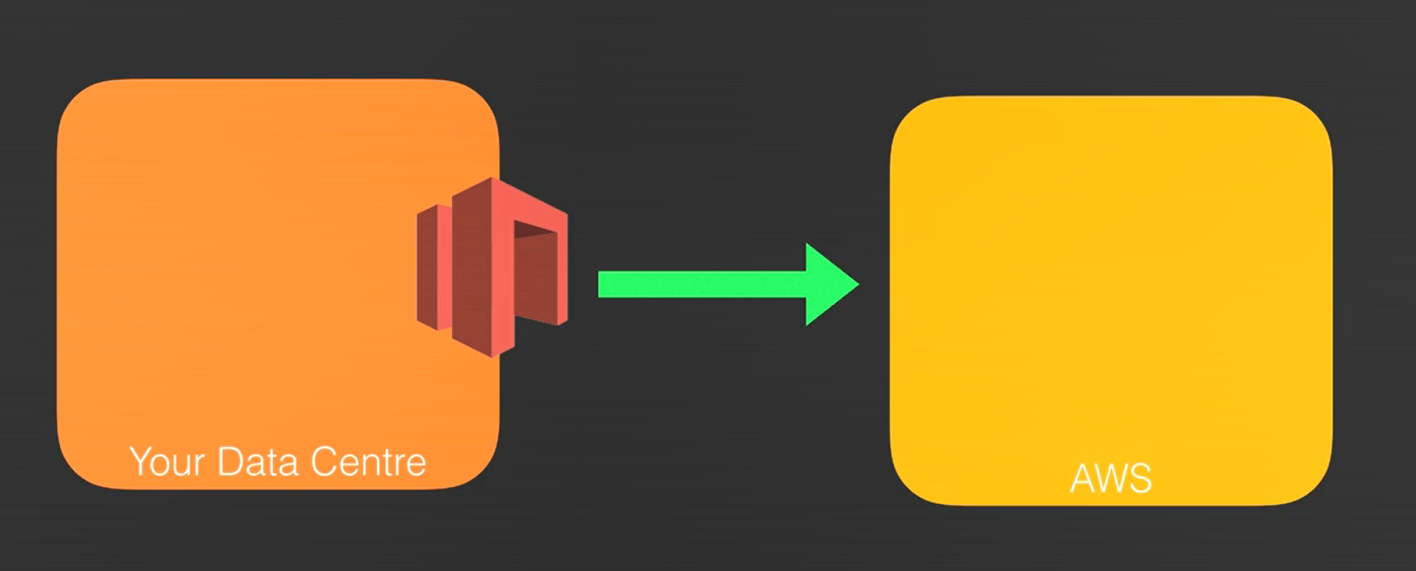
**What are the different encryption types ?**

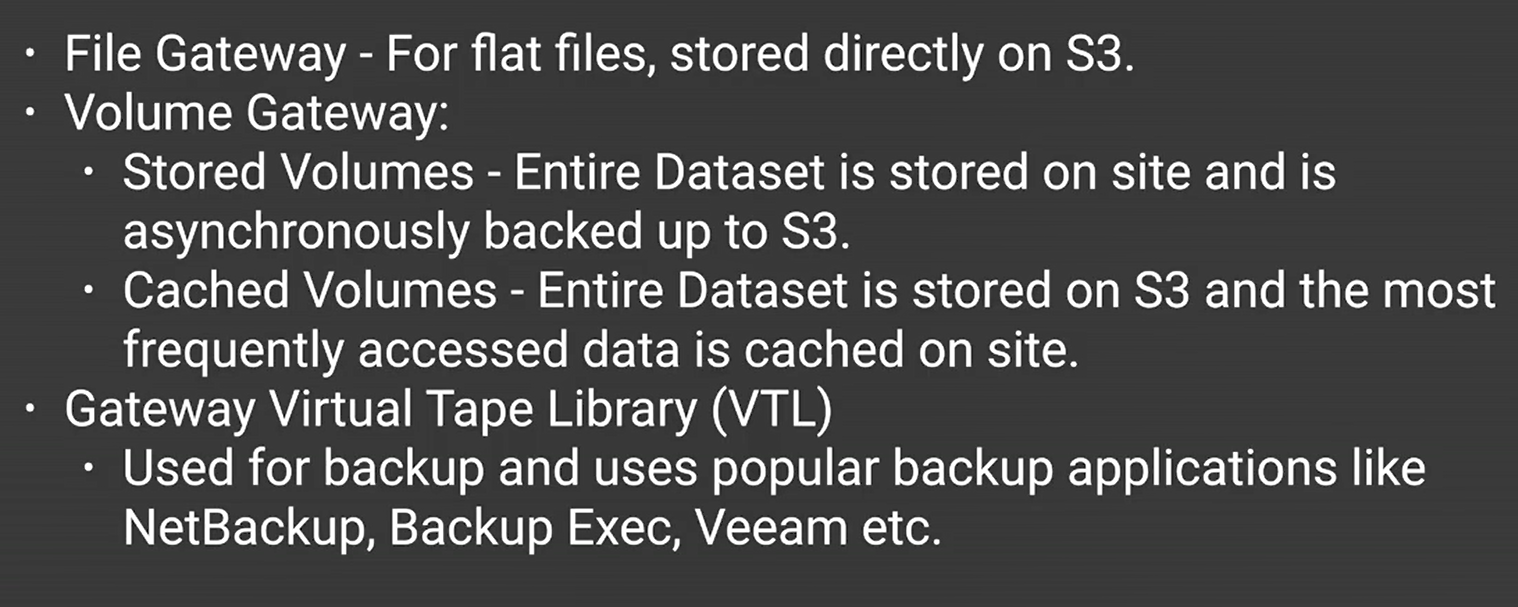
**In transit :** This is a simple upload and download file. It can be done by SSL/TSL . It is applicable at the time of storage lifecycle.

**At Rest :** It has two types server side and client side. Client side means we encrypt the file and upload it to s3. server side there are 4 different types

* S3 managed keys SSE-S3 – This is amazon managed standard encryption
* SSE-KMS = This is aws key management service. This comes with some extra benefits with some extra charges. It provides the facility of envelop key which protects the key used in SSE-S3. It provides a audit trail that who is using which key for which time tenure. this added some more traparency level.
* Server side customer provide key SSE-C

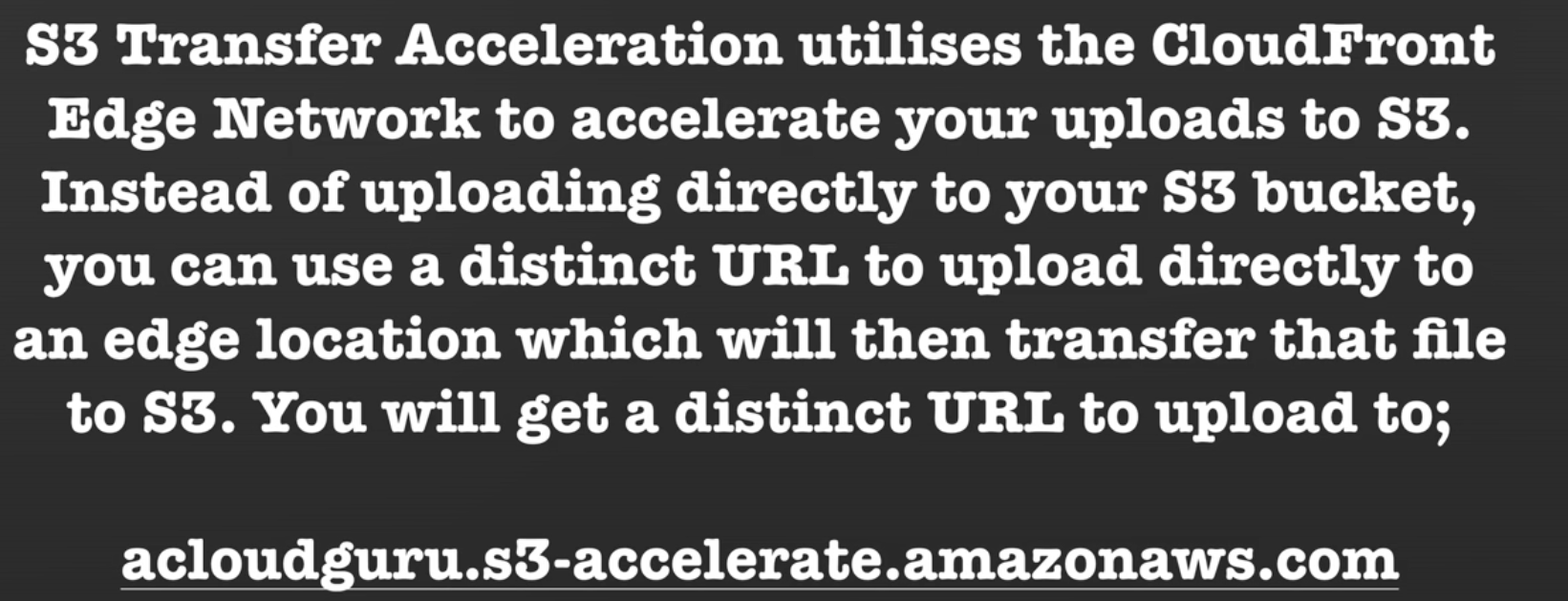
**What is storage gateway ?**





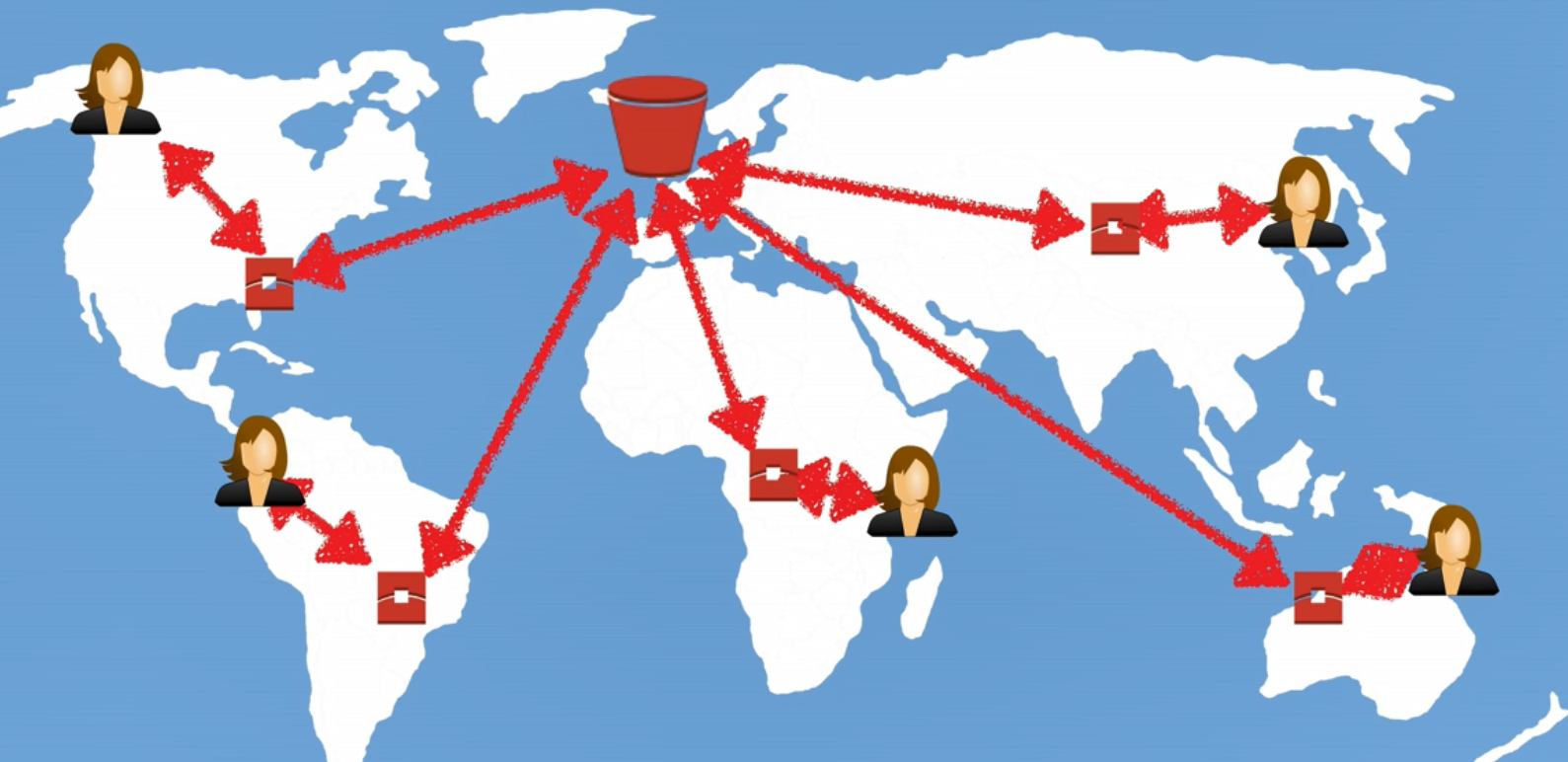
**What is transfer acceleration ?**

Amazon S3 Transfer Acceleration enables fast, easy, and secure transfers of files over long distances between your client and an S3 bucket. Transfer Acceleration takes advantage of Amazon CloudFront’s globally distributed edge locations. As the data arrives at an edge location, data is routed to Amazon S3 over an optimized network path.



The url pattern is

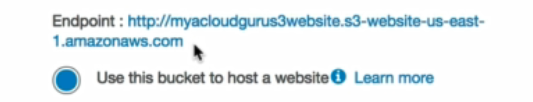
{bucketname} . s3-accelerate.amazon.ws.com



**How to create static website ?**

Static website we can create inside a bucket. It is in the property section of the bucket like other services versioning, encryption, logging, Transfer acceleration. Remember the bucket name used here should be same with Route53 domain name. the url pattern is like

{bucketname} . s3 –website-{datacenter or region}. amazonaws.com



**www.picsee.com website has millions of photos and also thumbnails for each photo. Thumbnails can easily be reproduced from the actual photo. However, a thumbnail takes less space than actual photo. Which of the following is the best solution to store thumbnails?**

Reduced redundancy storage

RRS is a highly available solution for distributing or sharing content that is durably stored elsewhere, or for storing thumbnails, transcoded media, or the processed data that can be easily reproduced