Amazon S3

Section introduction

- Amazon S3 is one of the main building blocks of AWS
- It's advertised as "infinitely scaling" storage

- Many websites use Amazon S3 as a backbone
- Many AWS services use Amazon S3 as an integration as well

- We'll have a step-by-step approach to S3

Amazon S3 Use cases

- Backup and storage
- Disaster Recovery
- Archive
- Hybrid Cloud storage
- Application hosting
- Media hosting
- Data lakes & big data analytics
- Software delivery
- Static website

Nasdaq

Nasdaq stores 7 years of data into S3 Glacier

Sysco

Sysco runs analytics on its data and gain business insights

Amazon S3 - Buckets

- Amazon S3 allows people to store objects (files) in "buckets" (directories)
- Buckets must have a globally unique name (across all regions all accounts)
- Buckets are defined at the region level
- S3 looks like a global service but buckets are created in a region
- Naming convention
  - No uppercase, No underscore
  - 3-63 characters long
  - Not an IP
  - Must start with lowercase letter or number
  - Must NOT start with the prefix xn--
  - Must NOT end with the suffix -s3alias

S3 Bucket

Amazon S3 - Objects

- Objects (files) have a Key
- The key is the FULL path:
  - s3://my-bucket/my_file.txt
  - s3://my-bucket/my_folder1/another_folder/my_file.txt
- The key is composed of prefix + object name
  - s3://my-bucket/my_folder1/another_folder/my_file.txt
- There's no concept of "directories" within buckets (although the UI will trick you to think otherwise)
- Just keys with very long names that contain slashes ("/")

Object

S3 Bucket
with Objects

Amazon S3 - Objects (cont.)

- Object values are the content of the body:
  - Max. Object Size is 5TB (5000GB)
  - If uploading more than 5GB, must use "multi-part upload"

- Metadata (list of text key / value pairs - system or user metadata)
- Tags (Unicode key / value pair - up to 10) - useful for security / lifecycle
- Version ID (if versioning is enabled)

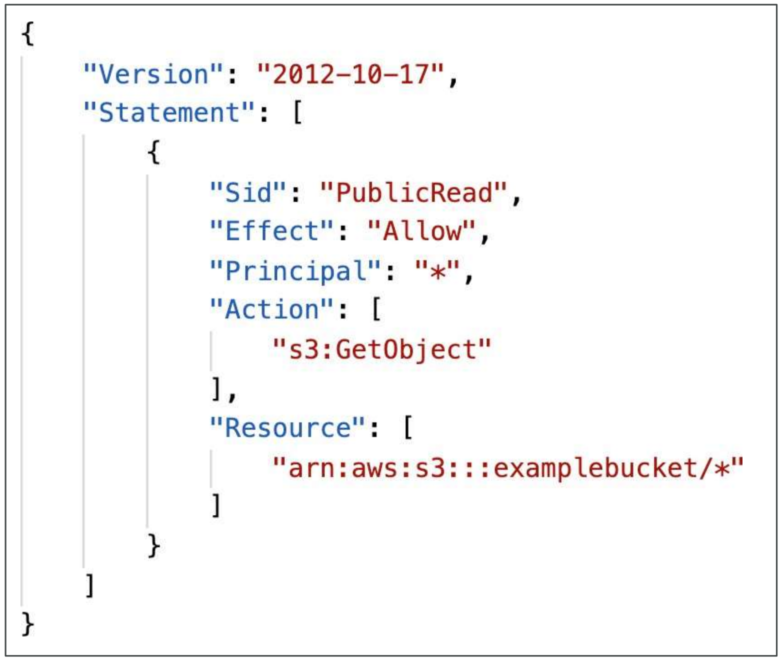
Amazon S3 – Security

- User-Based
  - IAM Policies – which API calls should be allowed for a specific user from IAM
- Resource-Based
  - Bucket Policies – bucket wide rules from the S3 console - allows cross account
  - Object Access Control List (ACL) – finer grain (can be disabled)
  - Bucket Access Control List (ACL) – less common (can be disabled)
- Note: an IAM principal can access an S3 object if
  - The user IAM permissions ALLOW it <u>OR</u> the resource policy ALLOWS it
  - <u>AND</u> there's no explicit DENY
- Encryption: encrypt objects in Amazon S3 using encryption keys

S3 Bucket Policies

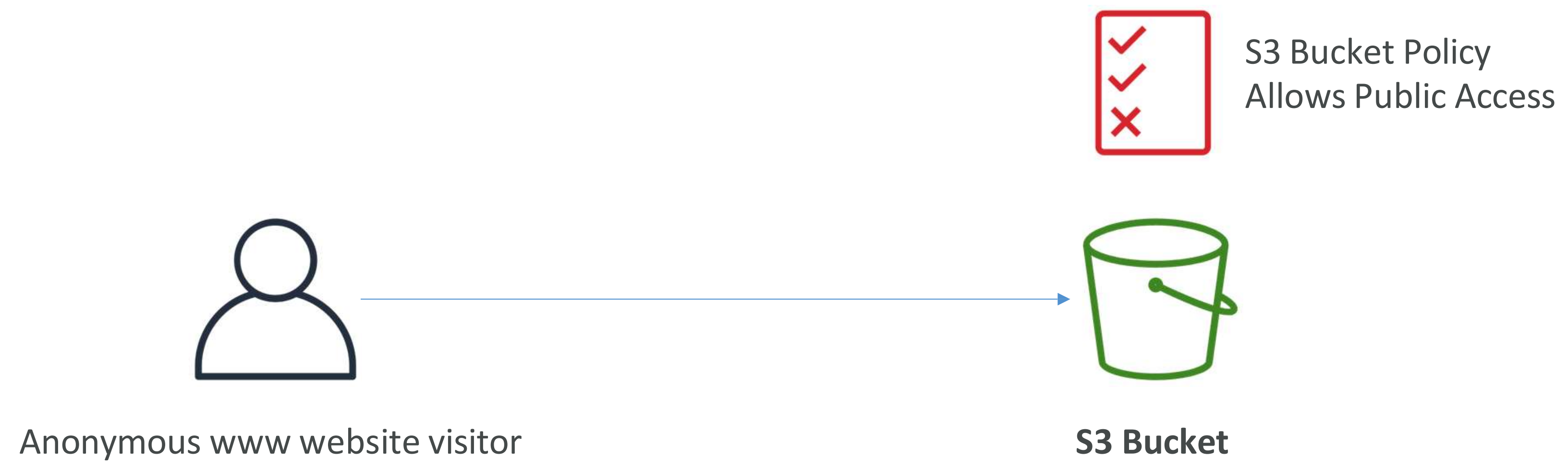
- JSON based policies
  - Resources: buckets and objects
  - Effect: Allow / Deny
  - Actions: Set of API to Allow or Deny
  - Principal: The account or user to apply the policy to

- Use S3 bucket for policy to:
  - Grant public access to the bucket
  - Force objects to be encrypted at upload
  - Grant access to another account (Cross Account)

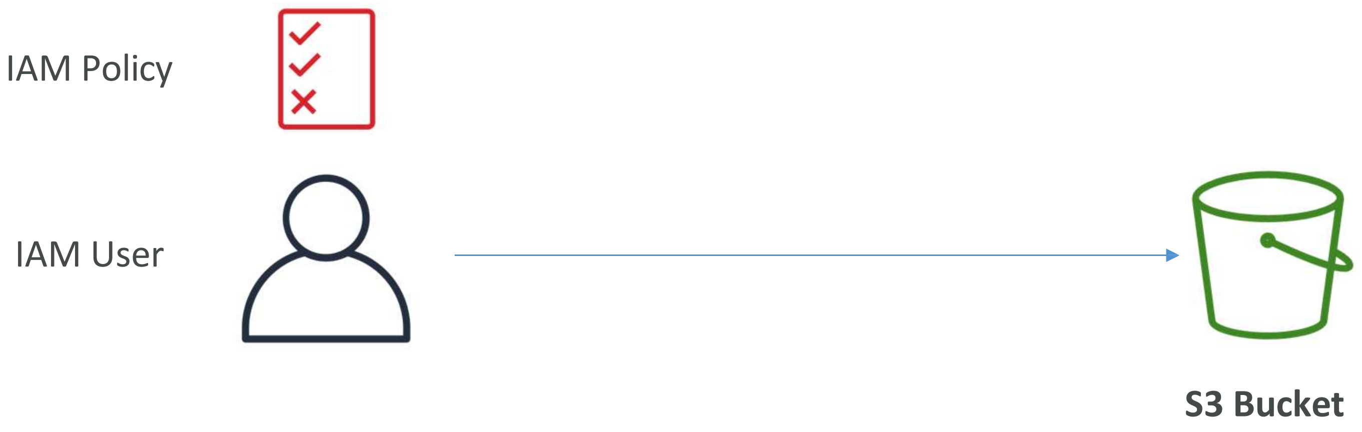


```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PublicRead",
            "Effect": "Allow",
            "Principal": "*",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::examplebucket/*"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

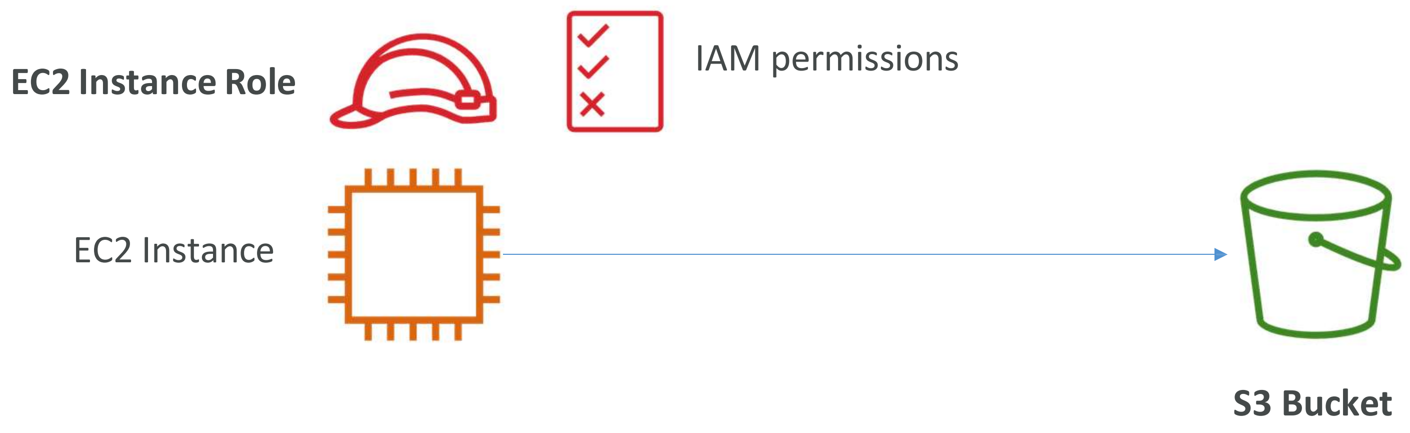
Example: Public Access - Use Bucket Policy



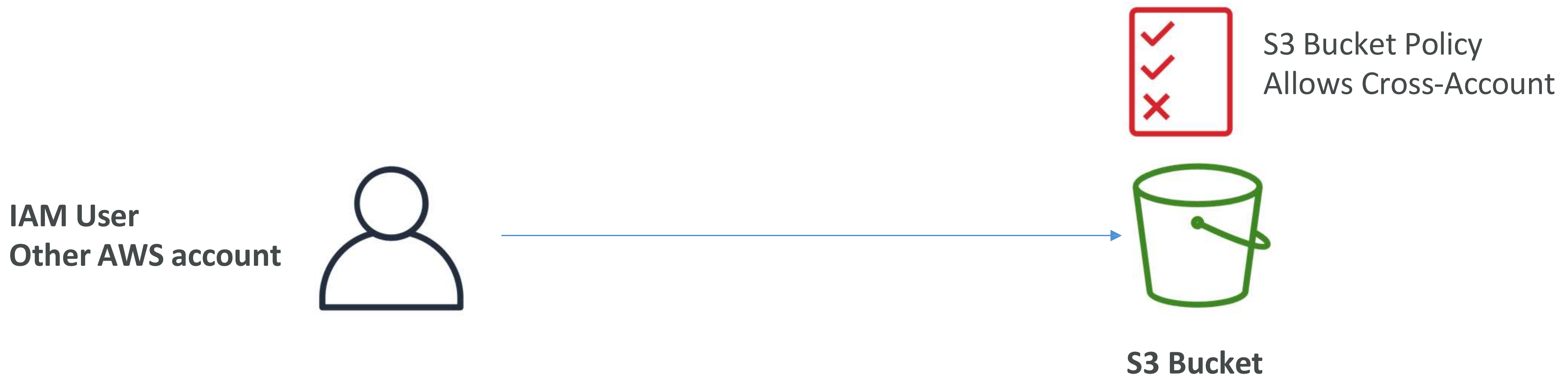
Example: User Access to S3 - IAM permissions



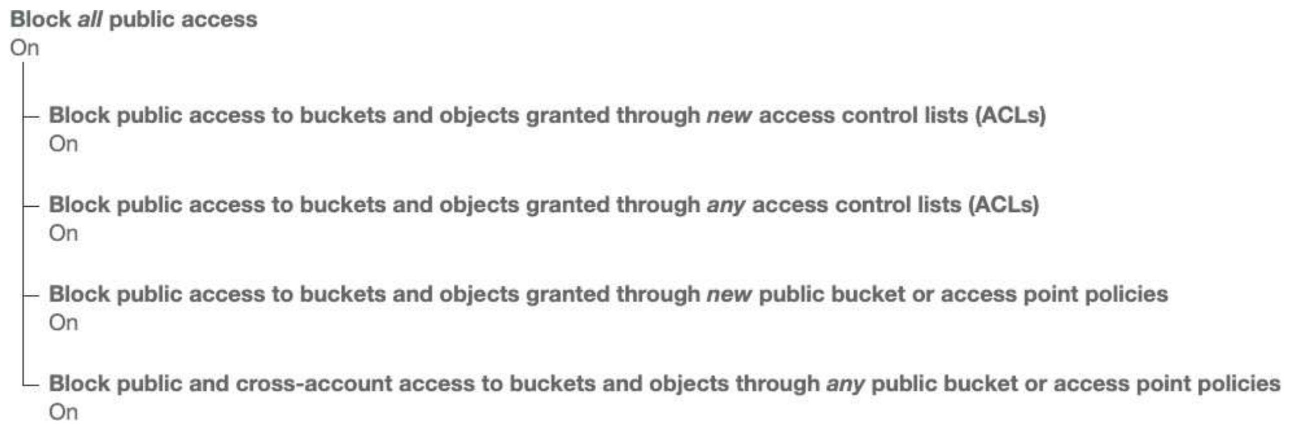
Example: EC2 instance access - Use IAM Roles



Advanced: Cross-Account Access - Use Bucket Policy



Bucket settings for Block Public Access



Block *all* public access
On

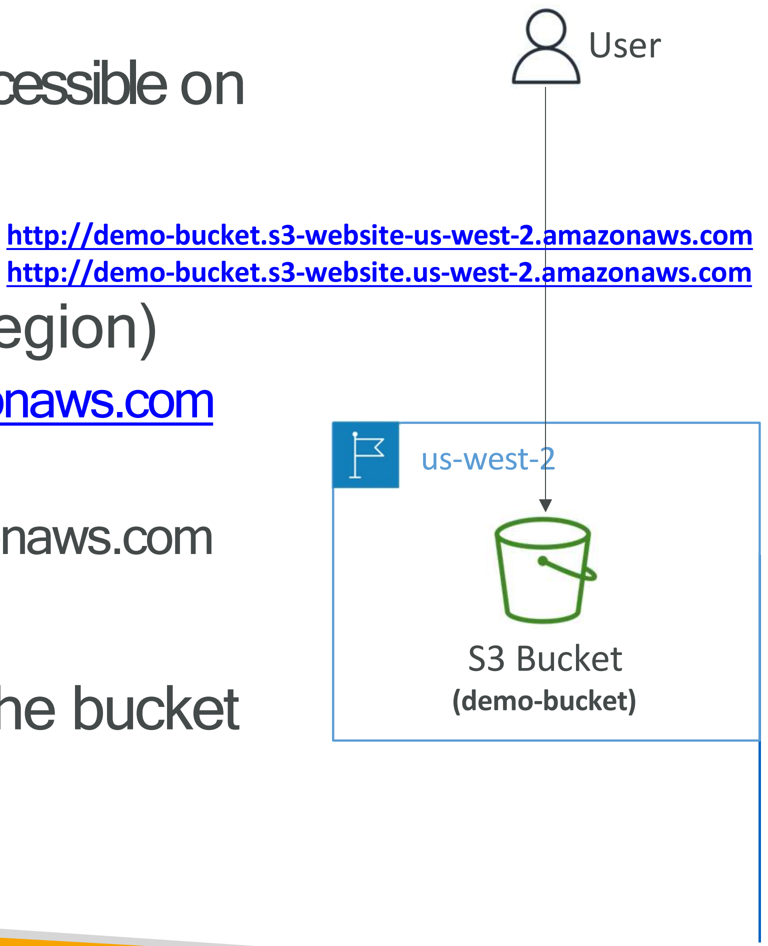
- Block public access to buckets and objects granted through *new* access control lists (ACLs)
  On
- Block public access to buckets and objects granted through *any* access control lists (ACLs)
  On
- Block public access to buckets and objects granted through *new* public bucket or access point policies
  On
- Block public and cross-account access to buckets and objects through *any* public bucket or access point policies
  On

- These settings were created to prevent company data leaks
- If you know your bucket should never be public, leave these on
- Can be set at the account level

Amazon S3 - Static Website Hosting

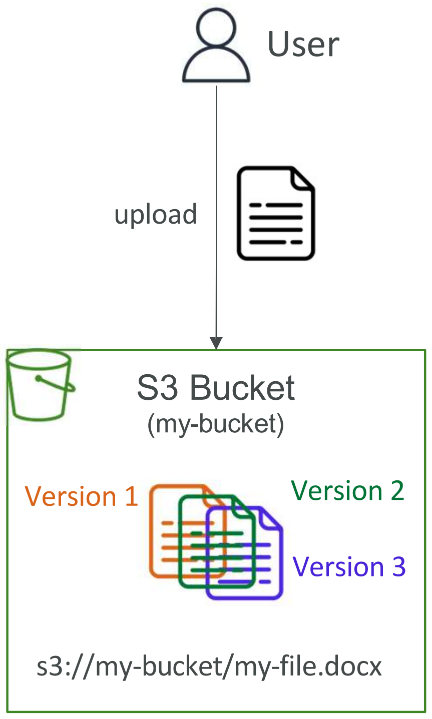
- S3 can host static websites and have them accessible on the Internet
- The website URL will be (depending on the region)
  - http://*bucket-name*.s3-website-*aws-region*.amazonaws.com

  OR
  - http://*bucket-name*.s3-website.*aws-region*.amazonaws.com
- If you get a 403 Forbidden error, make sure the bucket policy allows public reads!


User
http://demo-bucket.s3-website-us-west-2.amazonaws.com
http://demo-bucket.s3-website.us-west-2.amazonaws.com
us-west-2
S3 Bucket
(demo-bucket)

Amazon S3 - Versioning

- You can version your files in Amazon S3
- It is enabled at the bucket level
- Same key overwrite will change the "version": 1, 2, 3….
- It is best practice to version your buckets
  - Protect against unintended deletes (ability to restore a version)
  - Easy roll back to previous version
- Notes:
  - Any file that is not versioned prior to enabling versioning will have version "null"
  - Suspending versioning does not delete the previous versions



Amazon S3 - Object Encryption

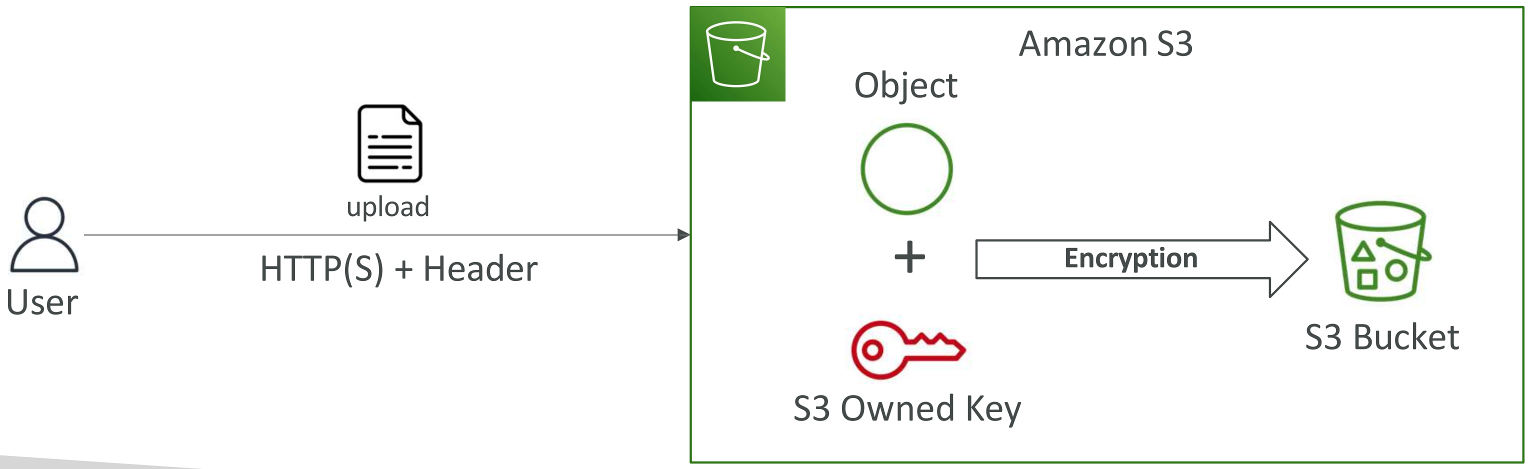
- You can encrypt objects in S3 buckets using one of 4 methods

- Server-Side Encryption (SSE)
  - Server-Side Encryption with Amazon S3-Managed Keys (SSE-S3) - Enabled by Default
    - Encrypts S3 objects using keys handled, managed, and owned by AWS
  - Server-Side Encryption with KMS Keys stored in AWS KMS (SSE-KMS)
    - Leverage AWS Key Management Service (AWS KMS) to manage encryption keys
  - Server-Side Encryption with Customer-Provided Keys (SSE-C)
    - When you want to manage your own encryption keys
- Client-Side Encryption

- It's important to understand which ones are for which situation for the exam

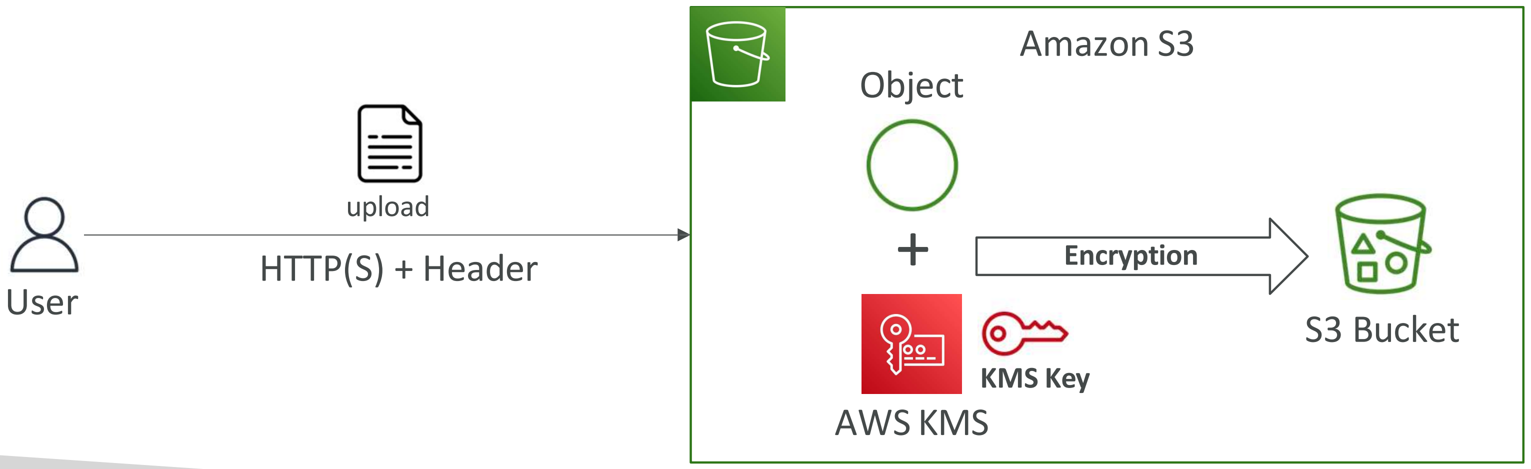
Amazon S3 Encryption – SSE-S3

- Encryption using keys handled, managed, and owned by AWS
- Object is encrypted server-side
- Encryption type is AES-256
- Must set header "x-amz-server-side-encryption": "AES256"
- Enabled by default for new buckets & new objects



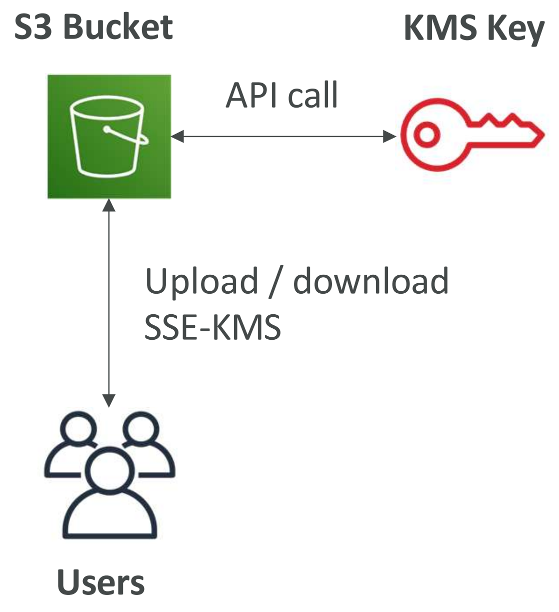
Amazon S3 Encryption – SSE-KMS

- Encryption using keys handled and managed by AWS KMS (Key Management Service)
- KMS advantages: user control + audit key usage using CloudTrail
- Object is encrypted server side
- Must set header "x-amz-server-side-encryption": "aws:kms"



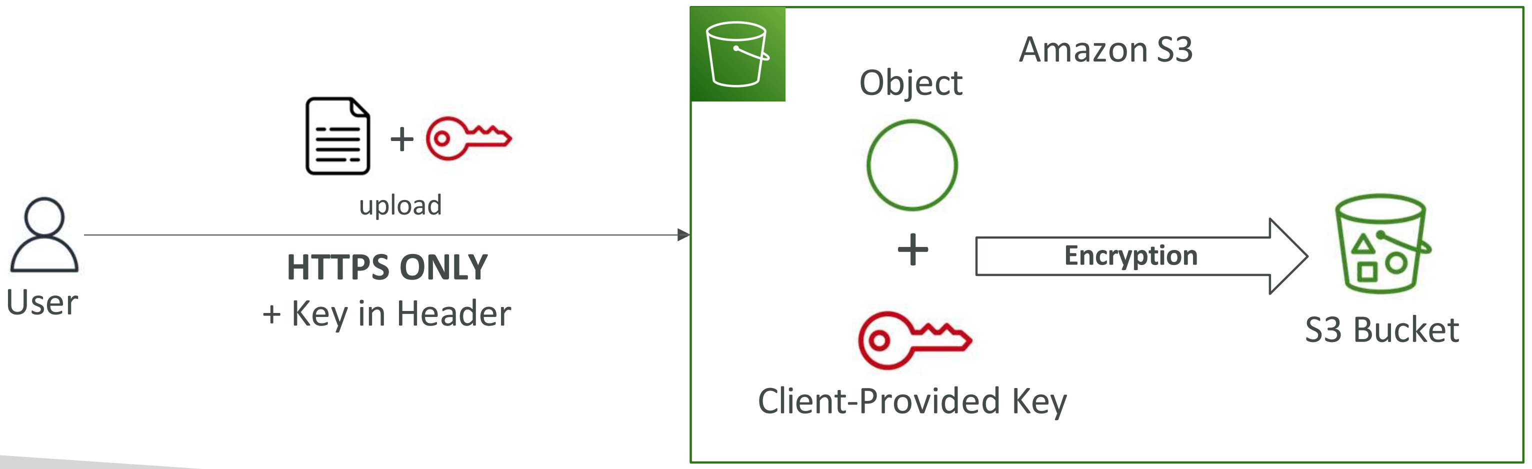
SSE-KMS Limitation

- If you use SSE-KMS, you may be impacted by the KMS limits
- When you upload, it calls the GenerateDataKey KMS API
- When you download, it calls the Decrypt KMS API
- Count towards the KMS quota per second (5500, 10000, 30000 req/s based on region)
- You can request a quota increase using the Service Quotas Console



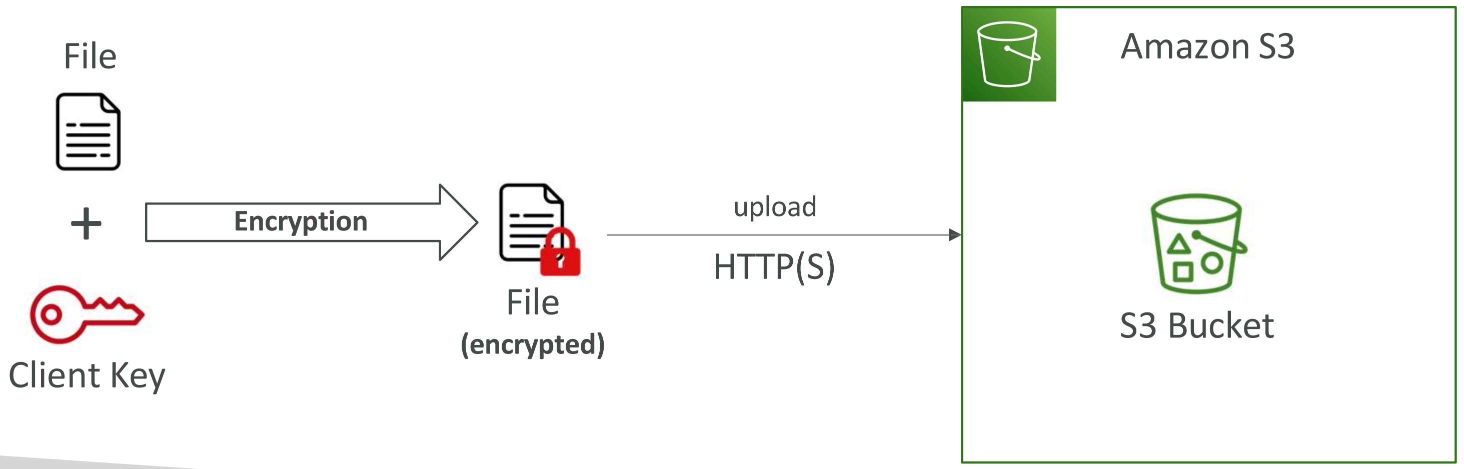
Amazon S3 Encryption – SSE-C

- Server-Side Encryption using keys fully managed by the customer outside of AWS
- Amazon S3 does NOT store the encryption key you provide
- HTTPS must be used
- Encryption key must provided in HTTP headers, for every HTTP request made



Amazon S3 Encryption – Client-Side Encryption

- Use client libraries such as Amazon S3 Client-Side Encryption Library
- Clients must encrypt data themselves before sending to Amazon S3
- Clients must decrypt data themselves when retrieving from Amazon S3
- Customer fully manages the keys and encryption cycle



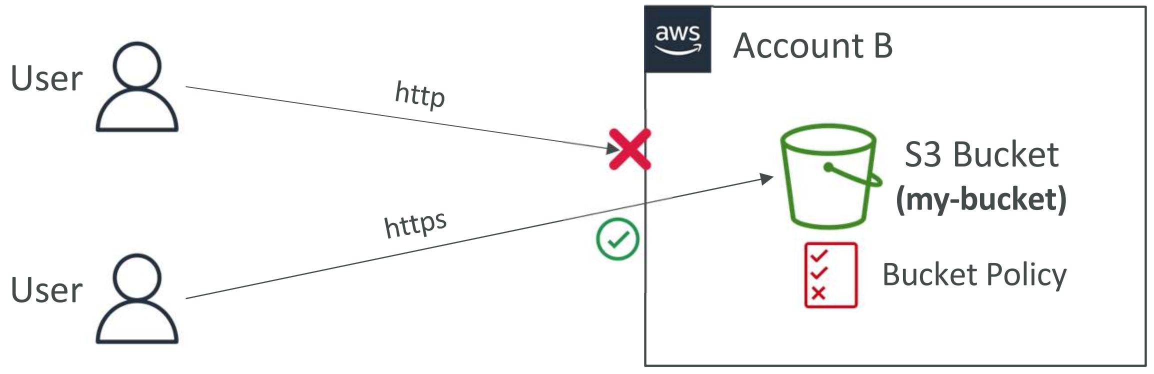
Amazon S3 – Encryption in transit (SSL/TLS)

- Encryption in flight is also called SSL/TLS

- Amazon S3 exposes two endpoints:
  - HTTP Endpoint – non encrypted
  - HTTPS Endpoint – encryption in flight

- HTTPS is recommended
- HTTPS is mandatory for SSE-C
- Most clients would use the HTTPS endpoint by default

Amazon S3 - Force Encryption in Transit aws:SecureTransport



```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Principal": "*",
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::my-bucket/*",
            "Condition": {
                "Bool": {
                    "aws:SecureTransport": "false"
                }
            }
        }
    ]
}
```

Amazon S3 - Default Encryption vs. Bucket Policies

- SSE-S3 encryption is automatically applied to new objects stored in S3 bucket
- Optionally, you can "force encryption" using a bucket policy and refuse any API call to PUT an S3 object without encryption headers (SSE-KMS or SSE-C)

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": "s3:PutObject",
            "Principal": "*",
            "Resource": "arn:aws:s3:::my-bucket/*",
            "Condition": {
                "StringNotEquals": {
                    "s3:x-amz-server-side-encryption": "aws:kms"
                }
            }
        }
    ]
}
```

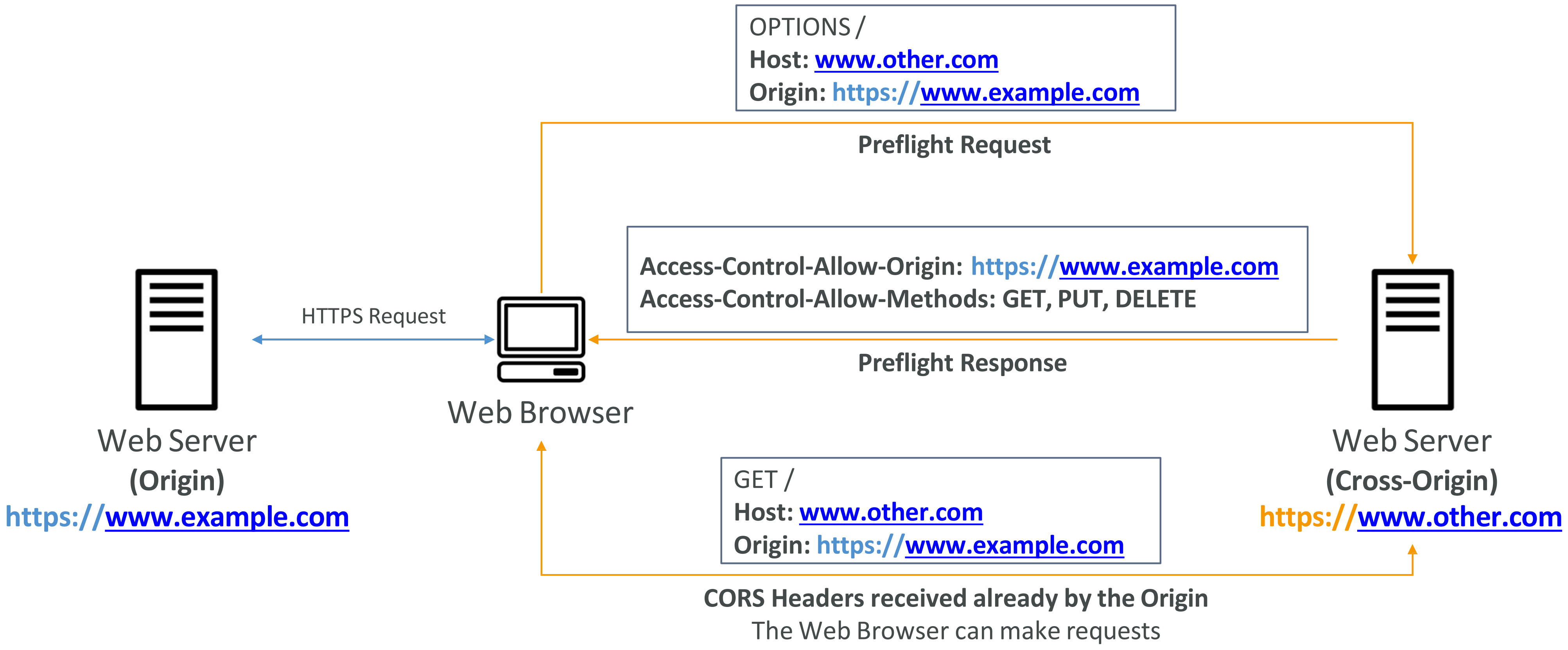
```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": "s3:PutObject",
            "Principal": "*",
            "Resource": "arn:aws:s3:::my-bucket/*",
            "Condition": {
                "Null": {
                    "s3:x-amz-server-side-encryption-customer-algorithm": "true"
                }
            }
        }
    ]
}
```

- Note: Bucket Policies are evaluated before "Default Encryption"

What is CORS?

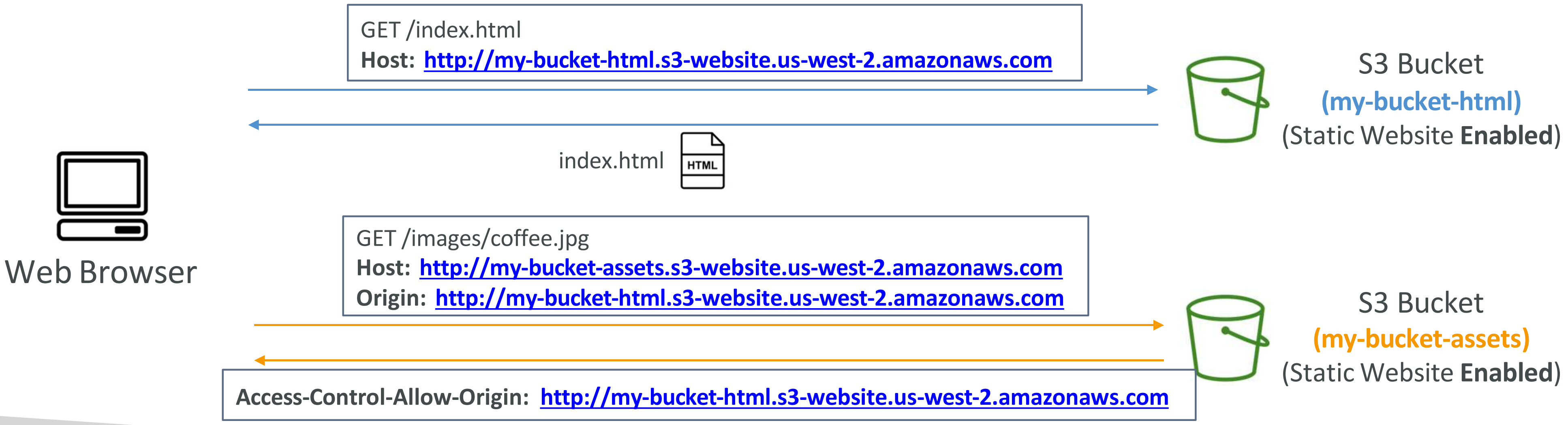
- Cross-Origin Resource Sharing (CORS)
- Origin = scheme (protocol) + host (domain) + port
  - example: https://www.example.com (implied port is 443 for HTTPS, 80 for HTTP)
- Web Browser based mechanism to allow requests to other origins while visiting the main origin
- Same origin: http://example.com/app1 & http://example.com/app2
- Different origins: http://www.example.com & http://other.example.com
- The requests won't be fulfilled unless the other origin allows for the requests, using CORS Headers (example: Access-Control-Allow-Origin)

What is CORS?


OPTIONS /
Host: www.other.com
Origin: https://www.example.com
Preflight Request
Access-Control-Allow-Origin: https://www.example.com
Access-Control-Allow-Methods: GET, PUT, DELETE
HTTPS Request
Preflight Response
Web Browser
Web Server
(Origin)
https://www.example.com
Web Server
(Cross-Origin)
https://www.other.com
GET /
Host: www.other.com
Origin: https://www.example.com
CORS Headers received already by the Origin
The Web Browser can make requests

Amazon S3 - CORS

- If a client makes a cross-origin request on our S3 bucket, we need to enable the correct CORS headers
- It's a popular exam question
- You can allow for a specific origin or for * (all origins)


GET /index.html
Host: http://my-bucket-html.s3-website.us-west-2.amazonaws.com
index.html
HTML
Web Browser
S3 Bucket
(my-bucket-html)
(Static Website Enabled)
GET /images/coffee.jpg
Host: http://my-bucket-assets.s3-website.us-west-2.amazonaws.com
Origin: http://my-bucket-html.s3-website.us-west-2.amazonaws.com
Access-Control-Allow-Origin: http://my-bucket-html.s3-website.us-west-2.amazonaws.com
S3 Bucket
(my-bucket-assets)
(Static Website Enabled)