|  |  |
| --- | --- |
| **Name:** | **LIM CHEE SENG** |
| **Email:** | **lim\_chee\_seng@mpa.gov.sg** |

**Question 1**

Amazon’s Simple Storage Service (S3) is an object storage service; objects are stored as key/value pairs. Objects can be added or read from S3 either through the web console or through RESTful API.

Read the following documentation of inserting an object into an S3 bucket

<https://docs.aws.amazon.com/AmazonS3/latest/API/API_PutObject.html>

and answer the following questions

1. What content type is supported by S3 in the put object operation?

S3 is capable of storing diverse and generally unstructured data, but it's also suited for hierarchical data and all kinds of structured information. Features such as metadata support, prefixes, and object tags allow users to organize data according to their needs.

All Standard MIME types listed in the ‘Content-Type’ parameter



1. How does S3 ensure payload integrity viz. the object that is uploaded, especially large object, has not been corrupted.

When you use PutObject to upload objects to Amazon S3, pass the Content-MD5 value as a request header. Amazon S3 checks the object against the provided Content-MD5 value. If the values do not match, you receive an error.

Additionally, you can calculate the MD5 while putting an object to Amazon S3 and compare the returned ETag to the calculated MD5 value.

1. What other algorithms (wrt b) does S3 support? How are these algorithms specified?

x-amz-sdk-checksum-algorithm

Indicates the algorithm used to create the checksum for the object when using the SDK. This header will not provide any additional functionality if not using the SDK. When sending this header, there must be a corresponding x-amz-checksum or x-amz-trailer header sent. Otherwise, Amazon S3 fails the request with the HTTP status code 400 Bad Request. For more information, see Checking object integrity in the Amazon S3 User Guide.

If you provide an individual checksum, Amazon S3 ignores any provided ChecksumAlgorithm parameter.

Valid Values: CRC32 | CRC32C | SHA1 | SHA256

x-amz-sdk-checksum-algorithm: ChecksumAlgorithm

x-amz-checksum-crc32: ChecksumCRC32

x-amz-checksum-crc32c: ChecksumCRC32C

x-amz-checksum-sha1: ChecksumSHA1

x-amz-checksum-sha256: ChecksumSHA256

1. How does S3 ensure content confidentiality?

To make sure your files and Amazon S3 buckets are secure, follow these best practices:

* Restrict access to your S3 resources: When using AWS, [restrict access to your resources](https://docs.aws.amazon.com/AmazonS3/latest/dev/access-control-overview.html) to the people that absolutely need it. Follow the principle of [least privilege](https://docs.aws.amazon.com/IAM/latest/UserGuide/best-practices.html#grant-least-privilege).
* Monitor your S3 resources: Monitor your resources using [AWS CloudTrail logs](https://docs.aws.amazon.com/awscloudtrail/latest/userguide/cloudtrail-getting-started.html), [S3 server access logging](https://docs.aws.amazon.com/AmazonS3/latest/dev/ServerLogs.html), [AWS Config](https://docs.aws.amazon.com/config/latest/developerguide/WhatIsConfig.html), [AWS Identity and Access Management (IAM) Access Analyzer](https://docs.aws.amazon.com/IAM/latest/UserGuide/what-is-access-analyzer.html), [Amazon Macie](https://docs.aws.amazon.com/macie/latest/user/what-is-macie.html), Amazon CloudWatch, or [AWS Trusted Advisor's S3 bucket permissions check](https://aws.amazon.com/premiumsupport/technology/trusted-advisor/best-practice-checklist/#security).
* Use encryption to protect your data: Amazon S3 supports encryption during transmission, [server-side encryption (SSE)](https://docs.aws.amazon.com/AmazonS3/latest/dev/serv-side-encryption.html), and [client-side encryption](https://docs.aws.amazon.com/AmazonS3/latest/dev/UsingClientSideEncryption.html).

**Server-side Encryption**

* You can optionally request server-side encryption. With server-side encryption, Amazon S3 encrypts your data as it writes it to disks in its data centers and decrypts the data when you access it. You have the option to provide your own encryption key or use AWS managed encryption keys (SSE-S3 or SSE-KMS). For more information, see [Using Server-Side Encryption](https://docs.aws.amazon.com/AmazonS3/latest/dev/UsingServerSideEncryption.html).

1. What strategy does this operation use to support S3 features (eg. encryption, storage classes, etc.) when an object is uploaded?

In your opinion, how are new S3 features supported by this operation?

x-amz-server-side-encryption-aws-kms-key-id: *SSEKMSKeyId*

x-amz-request-payer: *RequestPayer*

1. How does the put operation support caching?

[**Cache-Control**](https://docs.aws.amazon.com/AmazonS3/latest/API/API_PutObject.html#API_PutObject_RequestSyntax) **parameter**

Can be used to specify caching behavior along the request/reply chain.

1. How does the operation ensure that all the required parameters (eg. bucket name, encryption key, credentials, etc.) are correct be committing to the put operation?

We do a expect-100-Continue

PUT /my-image.jpg HTTP/1.1

Host: myBucket.s3.<Region>.amazonaws.com

Date: Wed, 12 Oct 2009 17:50:00 GMT

Authorization: authorization string

Content-Type: text/plain

Content-Length: 11434

x-amz-meta-author: Janet

**Expect: 100-continue**

[11434 bytes of object data]

Sample Response: Versioning suspended

This example illustrates one usage of PutObject.

HTTP/1.1 100 Continue

HTTP/1.1 200 OK

x-amz-id-2: LriYPLdmOdAiIfgSm/F1YsViT1LW94/xUQxMsF7xiEb1a0wiIOIxl+zbwZ163pt7

x-amz-request-id: 0A49CE4060975EAC

Date: Wed, 12 Oct 2009 17:50:00 GMT

ETag: "1b2cf535f27731c974343645a3985328"

Content-Length: 0

Connection: close

Server: AmazonS3

1. What are the main differences between this operation (PutObject) and PostObject (<https://docs.aws.amazon.com/AmazonS3/latest/API/RESTObjectPOST.html>)

The POST operation adds an object to a specified bucket by using HTML forms. POST is an alternate form of PUT that enables browser-based uploads as a way of putting objects in buckets. Parameters that are passed to PUT through HTTP Headers are instead passed as form fields to POST in the multipart/form-data encoded message body. To add an object to a bucket, you must have WRITE access on the bucket. Amazon S3 never stores partial objects. If you receive a successful response, you can be confident that the entire object was stored.

PUT – Only put 1 object

Multipart/form data can have many objects

1. S3 charges includes egress, viz. amount of data transferred out from a S3 bucket. If your server is using S3 for data storage, how do you reduce your S3 charges?

Use caching

**Submission**

Copy this Word document to your repository and commit it.

git add .

git commit -m ‘worksheet02’

git push origin master