**AWS Certified Solutions Architect Associate - SAA-C03**

**Exam Blue Print:**

* 130 Minutes in Length
* 65 Questions (this can change)
* Multiple Choice
* 70% to pass. Results are between 100 and 1000 with 720 of passing score.
* Certification valid for 3 years.
* Scenario based questions.

**Exam Tips:**

* Don’t spend too much time in hard questions first. Flag them and review later.
* Watch for keyword indicators on certain scenarios.
* Remember the Well-Architected Framework pillars.

**Exam Guide:**

* **Scenario Based:** Large majority of the questions based on real-world scenarios.
  + Pay attention to key words and requirements.
* **Multiple choice questions:** One correct response and three incorrect (distractors)
* **Multiple response:** Two correct responses out of five response options.

**Exam Domains:**

* **Design Resilient Architectures** – 26%
  + Design a multi-tier architecture solution.
    - How to make them scalable
    - Loosely coupled (building flexible, standalone, scalable services.)
  + Multitier (Multiple layers, like front end, backend, etc)
  + Design high available and/or fault tolerant architectures.
    - Know difference between both.
  + Design decoupling mechanisms using AWS services.
    - Message Queues
    - Notification topics.
  + Choose appropriate resilient storage.
    - What are the requirements for the storage
    - How often is accessed
    - etc
* **Design High-Performing Architectures 24%**
  + Identify elastic and scalable compute solutions for a workload.
    - Autoscaling groups
    - Load Balancers for handling or dispersing load
  + Select best high-performing and scalable **storage** solutions for a workload.
  + Select best high-performing and scalable **network** solutions for a workload.
    - Do you need VPN
    - Or Direct connect?
    - Etc
  + Choose best high-performing database solutions for a workload.
    - You need relational database? Like Dynamodb
    - Or Amazon Redshift?
    - etc
* **Design Secure Architectures 30%**
  + Design secure access to AWS resources
    - IAM
    - AWS single sign-on
    - Etc
  + Design secure application tiers
  + Select appropriate data security options
    - Encryption at rest
    - Encryption in transit
    - etc
* **Design Cost-Optimized Architectures 20%**
  + Identify cost-effective storage solutions
    - Different tiers in S3 storage
  + Identity most cost-effective compute and database services
    - Aurora serverless
    - Spot instances for EC2
    - Etc
  + Designing cost-optimized network architectures

**AWS Fundamentals**

**The building Blocks of AWS: Availability Zones and Regions**

**Region:** Is a physical location, like London, New York, Sidney, etc

**Availability Zone (AZ):**

* Is a datacenter withing a region. Each region must have at least 3 or more AZ.
* An AZ could be several datacenters if they are close together, then will be grouped in an AZ.
* Each AZ must be separated between them by many kilometres, although all are within 100 km of each other.

**Edge Locations:**

* Endpoint used for caching content.
* There are many more Edge Locations than Regions.
* Typically consists of CloudFront, Amazon’s CDN

**Who Owns What in the Cloud**

**The Shared Responsibility Model:**

**Customer:**

Responsible for security **IN** the Cloud:

A close-up of a sign

Description automatically generated

**AWS:**

Responsible for security **OF** the Cloud:

A purple rectangular object with white text

Description automatically generated

**Exam Tips:**

To know who is responsible (me or Amazon) ask:

Can I do it myself in the AWS management console?

**If yes**, you are likely responsible:

Security Groups, IAM users, patching EC2 OS, pathing db running in EC2.

**If not:**

Management of datacenters, security cameras, cabling, patching RDS OS, etc.

**Shared responsibility:**

Encryption is shared.

You could want to encrypt a volume, you are responsible of hitting the button and AWS is responsible of doing it.

**Compute, Storage, Databases, and Networking**

To make the exam easier we split things up into four key or core services:

* Compute
* Storage
* Databases
* Networking

**Compute:**

* EC2 🡪 Virtual Machines
* Lambda 🡪 Serverless, no virtual machines
* Elastic Beanstalk 🡪 Provisioning engine, to automate the deployment of my apps in AWS.

**Storage:**

* S3 🡪 To store files, etc.
* EBS, Elastic Block Store 🡪 Virtual hard disk attached to our virtual machines.
* EFS, Elastic File Service. 🡪 A way of store our files centrally.
* FSx 🡪 Fileserver storage for windows servers.
* Storage Gateway (connects on prem environment with cloud storage.)

**Databases:** Think about it as a spreadsheet (exel). It’s a reliable way to store and retrieve information.

* RDS, Relational Database Services
* DynamoDB, AWS non-relational database.
* Redshift, it’s a warehousing technology.

**Networking:**

* VPC’s, Virtual Datacenters in the cloud. Where are resources are going to live.
* Direct Connect, to connect our headquarters or on prem dc to AWS
* Route 53, DNS solution, from registering to pointing them out where we want.
* API Gateway, serverless way of replacing our webservers
* AWS Global Accelerator, a way to accelerating your audiences towards your applications within AWS.

**Exam Guide**

Is quite complex, have ~20 pages.

Tell you information about the exam and a lot of things related.

Have a read when you finish the course.

**Well-Architected Framework: Read this white paper before the exam:** <https://docs.aws.amazon.com/wellarchitected/latest/framework/welcome.html>

**Six Pillars of the Well-Architected Framework**

1. **Operational Excellence:**
   1. Running and monitoring systems to deliver business value, and continually improving processes and procedures.
2. **Performance Efficiency:**
   1. Using IT and computing resources efficiently
3. **Security:**
   1. Protecting information and systems
4. **Cost Optimization:**
   1. Avoiding unnecessary costs
5. **Reliability:**
   1. Ensuring a workload performs its intended function correctly and consistently when it’s expected to
6. **Sustainability:**
   1. Minimizing the environmental impacts of running cloud workloads.

**Exam Tips:**

**Region**: Physical location in the world. Consists in 3 or more AZ

**Availability Zone, AZ:**  Is one discrete datacenter, each with redundant power and networking.

**Edge Location:** Enpoints for AWS for caching content. Typically CloudFront (CDN)

**Who is responsible:**

Can you touch it by yourself: You

Is managed by AWS: AWS

**Compute:** EC2, Lambda, Elastic Beanstalk.

**Storage:** S3, EBS, EFS, FSx, Storage Gateway

**Databases:** RDS, DynamoDB, Redshift.

**Networking:** VPC, Direct Connect, Route 53, API Gateway, AWS Global Accelerator

Read the whitepaper of well architected framework.

**Identity and Access Management 101**

101 Means that is an introductory topic for a lecture. This is used in UK and the States.

**Ai** = I

**Ei** = A

**Em** = M

Ai Ei Em

**Root Account:**

* Is the email account you used to sign up for AWS.
* It has full administrative access to AWS.
* Is very important to SECURE this account.

**Exam Tips:**

* Enable Multi-factor authentication on the root account
* Create an admin group for your administrators and assign the appropriate permissions to this group.
* Create user accounts for your administrators.
* Add your users to the admin group.

**How do we control Permissions using IAM:** We assign permissions using policy documents, which are made up of JSON.

You will need to know how to read and understand JSON policy documents in the exam. [HERE](https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies.html#access_policies-json)

**IAM Policy Documents:** You can assign them to:

* **Groups** 🡪 you assign the user to a group and then assign the policy to the group.
* **Users** (is not typical to assign policies to users as makes difficult to manage)
* **Roles**

IAM does work at a Global level, any user, role, etc is created globally in all the regions of AWS.

**Apply or Create Policies:**

There are more than 1000 policy templates, or you can create your own one (which won’t have the orange box if you create it)

Example: AdministratorAccess policy

It grants access to all the services in AWS, the JSON is very simple:

{

"Version": "2012-10-17",

"Statement": [

{

"Effect": "Allow",

"Action": "\*",

"Resource": "\*"

}

]

}

Allow everything to all the resources.

**Exam Tips:**

* Assign permissions using IAM policy documents consisting of JSON
* In the exam you will have complicated S3 (could be other service) policy documents and will need to know that they do.
* Pick random policies and look how the JSON works.

**Permanent IAM Credentials:**

**User** = Human being. Should be a physical person.

**Groups**: We should group our users into Groups by Job Function (Admins, Marketing, BE Developers, etc)

**Roles:** For Internal usage within AWS. Allows a part of aws to access another part of aws. Like allowing an EC2 to access a S3 bucket.

**Best Practices:**

Inherit the permissions (Policies) from Groups instead of single users.

You can assign to users directly but is hard to manage as people comes and goes to the organization, or you have contractors, etc.

Is much easier to apply an IAM policy document to a group and the users in that group will inherit those permissions.

Never share user accounts.

**Principle of Least Privilege:**

Assign a user only the minimum number of privileges they need to do their job.

Example:

Sysadmin group will need full access to the AWS platform but not Finances or HR department.

If there is no Allow explicitly this means it’s a Deny.

**Tips when creating a user:**

\* Add tags, tags are key value pairs

Use these ones:

Department: Front End Developer

Employee\_ID: 123456

Now each time this user creates a resource, the resource will be tagged with this information and will be much easy to track who did it.

\* Policy type Job function

AWS already have a few job functions policies set up, like Network Admin, Data Scientist, Support User, etc.

For instance: PowerUserAccess 🡪 Can use everything but can NOT create users and groups.

**Once created the user:**

It will give you the option so send the sign-in instructions via email.

You will see the url the user has to use to access the console (aws gui)

You will see the user and password there too.

And will give you the option to download the password. You won’t be able to retrieve this password again so have this in mind.

**Account Settings:** IAM – Access Management – Account Settings

You manage there the password policies, example:

* How much characters the password has to have.
* Capital Letters and/or lowercase?
* Do you want to use special characters?
* Password expiration
* etc

**Created user without group:** Could be a question for the exam.

If you create a user without assigning him a group the only thing he will be able to do is to change its own password.

IAM – Users – test\_user – Permissions: Permissions policies 🡪 it will have only “

IAMUserChabgePassword” policy applied.

**Programmatic Access:** To use the command line.

IAM – Users – my\_user – Access key 1: Create access key – Command Line Interface (CLI) – Next – add tag if you want – Create

It will create an user and password:

Access key = Username

Secret access key = Password

This access key / secret access key is SEPARATED from the access to the AWS Gui (console).

**Identity providers:** Allow us to connect IAM to things like Active Directory.

IAM –Access Management -- Identity providers

* SAML: To connect AD
* OpenID Connect: To use it with services such as Google, Salesforce, etc.

Possible Exam question: You need to use the same SSO (Single sign-on) that you are using to login in into your windows environment in order to access AWS Management Console, what would you do?

I will add an Identity provider with the provider type SAML.

**Explicit Deny:** When you edit a policy by yourself.

An explicit deny will always override any allow in any other policy.

So if there is an explicit deny, IT WILL ALLWAYS DENY IT, doesn’t matter if there is another policy Allowing the same resource.

Let’s say you have two policies applied to your user:

ec2-admin

all-admin

In both you can see/run/stop ec2 instances, but if you edit manually the policy applied to this user:

IAM – Users – user1 – Permissions – select ec2-admin for example – Edit – (it will redirect you to IAM – User Groups – ec2-admin) and you change the Effect *allow* for *deny*

{

"Version": "2012-10-17",

"Statement": [

{

"Action": [

"ec2:Describe\*",

"ec2:StartInstances",

"ec2:StopInstances"

],

"Resource": "\*",

**"Effect": "Allow" 🡪 before the change.**

},

**"Effect": "Deny" 🡪 after the change.**

Now we have two policies, ec2-admin (deny) and all-admin (allow), both applying to actions over ec2 instances.

If one says allow and the other says deny for the same resource it will DENY the use of that resource.

**Exam Tips:**

* **IAM is universal:** It does not apply to regions, is created across the world globally.
* **The root Account:** The account is created when you first set up your AWS account and which has complete admin access. Secure it as soon as possible and DO NOT use it to log in day to day.
* **New Users:**  No permissions when first created, we must add the permissions manually or add the user to a group. The user will have only the “IAMUserChabgePassword” policy.
* Access key ID and Secret access keys are not the same as usernames and passwords, are used for programmatic access to the AWS console. You only will see the access key and the password once, when you create it, so make sure you saved it.
* **Always set up password rotations:** You can create your own password rotation policies.
* **Identity Federation / IAM Federation:** Identity Providers: To Single Sign-On (SSO) into AWS Management Console using the user/password of another provider, like Active Directory, Salesforce, Google, etc. It uses these Protocols:
  + **SAML:** Security Assertion Markup Language, MS Active Directory
  + **OpenID Connect:** Salesforce, Google, etc
* **Explicit Deny:** If you have two policies applying to the same, full\_access and sys\_admin for instance, if you deny a policy in one of them, it will deny the same in the other.

**S3**

**S3 Is object-based storage:** Manages data as objects rather than in file systems or data blocks.

* Can NOT be used to run an OS or Database
* Upload any file type you can think of to S3
* Examples: Photos, videos, code, documents, and text files.

**S3 Basics:**

* Unlimited storage.
* Objects up to 5 TB in size.
* Buckets: Are the folders of S3, is where the files are stored.

**Working with S3:**

* **Universal Namespace:** The name of the buckets has to be globally unique.
  + **Naming**:
    - https://Bucket-name.s3.Region.amazonaws.com/key-name
    - <https://polla_99.s3.us-east-1.amazonaws.com/my_file.txt>
* **Successful code:** If you upload a file successfully you will receive a code 200.

**Key-Value Store:**

* **Key**: Name of the object: my\_file.txt
* **Value**: The data itself, which is made of a sequence of bytes.
* **Version ID**:To store multiple versions of the same object (file).
* **Metadata**: Is data about data. Examples of metadata in s3:
  + The content type.
  + The last time the file was modified.
  + Etc

**Highly Available and Highly Durable:**

* **Built for Availability:** Built for 99.95% - 99.99% service availability, depending on the s3 tier.
* **Designed for Durability:** Designed for 99.999999999% (11 nines) durability for data stored in s3. You should NOT lose any object stored in S3.

**S3 Standard:** The default version of S3 when you store your objects.

* **High Availability and Durability:**
  + Data stored redundantly across multiple devices in multiple facilities (>= AZs):
    - 99.99% availability
    - 99.999999999% (11 nines) durability
* **Designed for Frequent Access:** Like reading and writing every second, or hour or day for insntance.
* **Suitable for Most Workloads:**
  + The default storage class
  + Use cases include websites, content distribution, mobile and gaming applications, and big data analytics.

**Tiered Storage:**

S3 offers a range of storage classes designed for different use cases.

**Lifecycle Management:**

Define rules to automatically transition objects to a cheaper storage tier or delete objects that are no longer required after a set period of time.

**Versioning:**

All versions of an object can be retrieved, including deleted objects.

**Securing your Data:**

* **Server-Side Encryption:** You can set default encryptions o a bucket to encrypt all the new objects when they are stored in the bucket.
* **ACL:** Define which AWS accounts or goups are granted access and the type of access. You can attach S3 ACLs to individual objects withing a bucket.
* **Bucket Policies:** This specify what actions are allowed or denied, example: Allow user Alice to PUT but not DELETE objects in the bucket. Are JSON policies attached to buckets.

**Data Consistency Model:** Strong Read-After-Write Consistency.

As soon as you written something to S3 it will be immediately available.

* **After a successful** write of a new object (PUT) or an overwrite of an existing one, any subsequent read request immediately receives the latest version of the object.

* **Strong Consistency:** For Lists operations, so after a write you can immediately perform a listing of the objects in a bucket with all changes reflected.

**Exam Tips:**

* Object Based storage that allows to upload files
* Files from 0 bytes to 5 TB.
* Not suitable for OS or DB Storage. Only flat or static files.
* Unlimited Storage.
* Files stored in buckets.
* S3 is a universal namespace (you cannot have two buckets with the same name)
* Successful CLI or API uploads will generate an HTTP 200 status code.
* S3 Object tips:
  + Key: The object name, like my\_file.txt
  + Value: The data itself, which is made up of a sequence of bytes.
  + Version ID: Allows to store multiple versions of the same object.
  + Metadata: Data about the data you are storing, like last modified, etc.

**Securing your bucket with S3 Block Public Access:**

**ACL:** Object ACLs work on an individual object level.

You can make an individual file public or readable, you can adjust the permissions so certain AWS users could access and read that file or download it or write to it, or delete it, and then deny the right of other users to do that.

You can do that on an individual object or file level.

**Bucket Policy:** Work on an entire bucket level.

**Exam Tips:**

* **Buckets are private by default:** When you create an S3 bucket is private by default, including all objects within it. You must allow public access on both the bucket and its objects in order to make the bucket public.
* **Objects ACLs:** You can make individual objects public using object ACLs.
* **Bucket Policies:** You can make entire buckets public using bucket policies.
* **HTTP status code:**  When you upload an object to S3 and it’s successful, you will receive an HTTP 200 code**.**

**Hosting a Static Website using S3:**

**Static websites** don’t change, is basically html.

**Dynamic websites,** such as those that require Databases connections and can not be hosted in S3.

**Example:**

You can use a static S3 website to host a movie trailer and the websites him self’s on S3.

That way you can handle the load, you don’t have to worry about load balancers, capacity planning, etc, S3 will do all for us.

S3 scales automatically, so many enterprises will put static websites on S3 when they think there is going to be a large number of requests, like for a movie preview.

**Exam Tips:**

* **Bucket Policies:** Make entire buckets public using bucket policies.
* **Static Content:** Use S3 to host static content only (not dynamic).
* **Automatic Scaling:** S3 scales automatically with demand.

**Versioning Object in S3:**

**What is versioning:** You can enable versioning in S3 so you can have multiple versions of an object within S3.

**Advantages of versioning:**

* **All versions** of an object are stored in S3. This includes all writes and even if you delete an object.

* **Backup:** Can be a great backup tool.
* **Cannot Be Disabled:** Once enabled, cannot be disabled, only suspended.
* **Lifecycle Rules:** Can be integrated with lifecycle rules.
* **Supports MFA:** Can support MFA authentication.

**Important:** Old versions are not publicly available.

If you have a new version of an html code, for instance, and the code is public, and you go to the old version of the code and hit the link to see the public website, you WON’T be able.

Only the last version is publicly available.

**Delete an object:**

The object will be removed from the GUI but all the versions will be there.

You only have to enable “Show versions” in the bucket and they will appear.

A Type: “Delete marker” will appear showing that you deleted that object, but all the versions will be still available and restorable.

**Restore a deleted object:**

You have to delete the “Delete marker”.

Once this is done, the deleted object will appear again in the GUI, with all the versions.

**Exam Tips:**

* **All versions** of an object are stored in S3. This includes all writes and even if you delete an object.

* **Backup:** Can be a great backup tool.
* **Cannot Be Disabled:** Once enabled, versioning cannot be disabled, only suspended.
* **Lifecycle Rules:** Can be integrated with lifecycle rules.
* **Supports MFA**, typical exam question: How do you protect objects of deletion:
  + Enable Versioning
  + Enable MFA

**S3 Storage Classes**

**S3 Standard:**

* High Availability and Durability
  + Data is stored redundantly across multiple devices in multiple facilities >= 3AZs
    - 99.99% Availability
    - 99.999999999% (11 nines)
  + Designed for frequent accesses data
  + Suitable for most workloads
    - The default storage class
    - Use cases include websites, content distribution, mobile and gaming applications and bit data analytics.

**S3 Standard-IA:** Infrequent Access. Designed for infrequently accessed data.

* **Rapid Access**: Used for data that is accessed less but requires rapid access when needed.

* **You pay to Access the data:** There is a low GB storage price and a low GB retrieval fee.
* **Use Cases:** Great for long term-storage, backups and as data store for disaster recovery files.
* 99.9% Availability
* 99.999999999% (11 nines)

**S3 One Zone-Infrequent Access:** Data is stored redundantly within a single AZ.

For Non-Critical data. To save money.

* Costs 20% less than regular S3 Standard-IA
* Great for long-lived, infrequently accessed, non-critical data.
* 99.5% Availability
* 99.999999999% (11 nines)
* You can afford to lose the data, but you need to minimize costs

**S3 Intelligent-Tiering:** When you don’t know whether you data will be accessed frequently or infrequently

* Uses AI and machine learning to figure out how frequently you access your data and then will store on the storage class that’s going to save you most money based on your access patterns.
* Automatically moves your data to the most cost effective tier based on how frequently you access each object.
* 99.9% availability.
* 99.999999999% (11 nines) durability.
* Good for Optimizing Costs: Monthly fee of $0.0025 per 1000 objects.

**Glacier Options:**

* You pay each time you access your data.
* Use only for archiving data.
* Glacier is cheap storage.
* Optimized for data that is very infrequently accessed.
* 99.99% availability.
* 99.999999999% (11 nines) durability.
* Retrieval fee apply for all archive classes.
* **Glacier Instant Retrieval:** Provides long term data archiving with instant retrieval time for your data, examples like:
  + Recovery of production website or database.
* **Glacier Flexible Retrieval:** No require immediate access but needs the flexibility to retrieve large sets of data at no cost, such:
  + Backup or disaster recovery.
  + Can be minutes or up to 12 hours.
  + You only use this if you can wait up to 12 hours to access the data.
* **Glacier Deep Archive:**
  + Cheapest storage.
  + Designed to retain data sets for 7-10 years or longer to meet regulatory compliance.
  + The standard retrieval time is 12 hours.
  + Bulk retrieval time is 48 hours.

A screenshot of a computer screen

Description automatically generated

**Exam Tips:**

A screen shot of a blue and white screen

Description automatically generated

**Lifecycle Management with S3:**

**What is:** Automates moving your objects between the different storage tiers, thereby maximizing cost effectiveness.

It’s a way of automating moving your objects to save you money.

**Use it with versioning:** Move different versions of objects to different storage tiers.

**Exam Tips:**

* Automates moving objects between different storage tiers.
* Can be used in conjunction with versioning.
* Can be applied to current versions and previous versions.

**S3 Object Lock and Glacier Vault Lock:**

Will be in the exam

**S3 Object Lock:** You can use it to store objects using a write once, read many (WORM) model. It can help prevent objects from being deleted or modified for a fixed amount of time or indefinitely.

You can use it to meet regulatory requirements that require WORM storage or add an extra layer of protection against object changes and deletion.

**Modes:**

**Governance Mode:** Only some users can overwrite or delete.

Users cannot overwrite or delete an object version of alter its lock settings unless they have special permissions.

With governance mode, you protect objects against being deleted by most users, but you can still grant some users permissions to alter the retention settings or delete the object if necessary.

**Compliance Mode:** Nobody can overwrite or delete.

A protected object version cannot be overwritten or deleted by any user, including the root user in your aws account.

When an object is locked in compliance mode, its retention mode can’t be changed and its retention period can’t be shortened.

Compliance mode ensures an object cannot be overwritten or deleted for the duration of the retention period.

**Retention Periods:**

A retention periods protects an object version for a fixed amount of time.

When you place a retention period on an object version, S3 stores a timestamp in the object version’s metadata to indicate when the retention period expires.

After the retention period expires, the object version can be overwritten or deleted unless you also placed a legal hold on the object version.

**Legal Hold:** Can be placed and removed by the user with permissions.

S3 Object Lock also enables you to place a legal hold on an object version. Like a retention period, a legal hold prevents an object version from being overwritten or deleted. However, a legal hold doesn’t have an associated retention period and remains in effect until removed.

Legal holds can be freely placed and removed by any user who has the:

S3:PutObjectLegalHold permission.

**Glacier Vault Lock:**

Allows you to easy deploy and enforce compliance controls for individual S3 Glacier vaults with a vault lock policy.

You can specify controls, such as WORN, in a vault lock policy and lock the policy from future edits.

Once locked, the policy can no longer be changed.

**Exam Tips:**

* Use S3 Object Lock to store objects using write once, read many (WORM) model.
* Object Lock can be on individual objects or applied across the bucket as a whole.
* Object Lock comes in two modes:
  + Governance mode: Some users can modify or delete.
  + Compliance mode: No one can modify or delete, not even the root account.
* S3 Glacier Vault Lock: you can deploy and enforce compliance controls for individual S3 Glacier vaults with a vault policy.
  + You can specify controls, such as WORM, in vault lock policy and lock the policy from future edits. Once locked, the policy can no longer be changed.
* If you see WORM without S3 before it means that is talking about Glacier.

**Encrypting S3 Objects:**

Will be asked in the exam

**Types of Encryptions:**

**Encryption in Transit:** To avoid someone to sniff what you send over internet or other networks you encrypt the communication.

* SSH
* VPN
* SSL/TLS
* HTTPS

**At Rest:** You encrypt your files in your hard drive, database, or cloud storage. Then if someone stole the hard drive its content will be encrypted. This is encryption at rest.

**Encryption in Transit:** You send it encrypted. We do it using:

* SSL/TLS
* HTTPS (443)

**Encryption at Rest: Server-Side Encryption** Is when the object is in your server or in S3 Anything you put in S3 will be encrypted by default.

* **SSE-S3:** S3-managed keys, using AES 256-bit encryption.
  + AWS manages all the encryption and decryption for me.
  + The most common type of encryption and the easiest to use.
* **SSE-KMS:** AWS Key Management Service-managed keys.
  + You work with AWS with your own key management using the key management service (KMS).
  + This gives you more options, like key rotation, audit log, granular access.
* **SSE-C:** Customer-provided keys.

**Encryption at Rest: Client-Side Encryption** You encrypt the files yourself before you upload them to S3.

* You encrypted your objects on your desktop, for example, and then you uploaded them to S3.
* You control the encryption and decryption and all the objects inside S3 are encrypted.

**Encryption by default:**

All objects you put in S3 buckets will be encrypted SSE-S3 by default.

**Enforcing Server-Side Encryption for S3 Uploads:**

You can create bucket policies that deny any S3 PUT request that doesn’t include encrypted objects, it doesn’t include the encryption parameter in the request header.

You can enforce encryption using bucket policies.

**Exam Tips:**

* **Encrypt in Transit:**
  + SSL/TLS
  + HTTPS
* **Encrypt at Rest:** Server Side Encryption (encrypt the hard drive)
  + SSE-S3 (AES 256-bit)
  + SSE-KMS
  + SSE-C (customer)
* **Client-Side Encryption**
  + You encrypt the files yourself before you upload to S3
* **Enforcing Encryption with a Bucket Policy**
  + A bucket policy can deny all PUT requests that don’t include the x-amz-server-side-encryption parameter in the request header.

**Optimizing S3 Performance:**

**S3 Prefixes:** Is the folders before your object and after the bucket name.

my\_documents/**studies/maths**/exam.docx

You have the bucket called: my\_documents

Inside this bucket you have the folder: studies

And inside you have the folder: maths

And inside you have the object (file) exam.docx

The prefix is: studies/maths

Another example:

my\_bucket\_name/folder1/subfolder1/myfile.jpg

folder1/subfolder1/ 🡪 this is the S3 prefix.

**S3 Performance:** S3 has extremely low latency, 100-200 milliseconds.

It has high numbers of requests per second per prefix:

3,500 PUT/COPY/POST/DELETE

5000 GET/HEAD

**Get better performance:** The more S3 Prefixes you have the higher performance you are going to get.

Spread your reads across different prefixes:

* 2 x S3 Prefixes = 11,000 requests per second
  + Folder1 and Folder2
* 4 x S3 Prefixes = 22.000 requests per second
  + Folder1, Folder2, Folder3 and Folder4

**S3 Limitations when using KMS:** KMS comes with built-in limits and therefore will count on my KMS quota.

* If you are using SSE-KMS to encrypt S3 you will have KMS limits.
* When you upload a file, you will call GenerateDataKey in the KMS API.
* When you download a file, you will call Decrypt in the KMS API.

KMS Requests Rates:

* Uploading/Downloading will count in the KMS quota.
* Currently, you cannot request a KMS quota increase.
* Region-specific, however, its either 5,000 – 10,000 or 30,000 requests per second.

Possible exam question:

* If you get an encryption question you may be interested in using the default S3 encryption (SSE-S3)
* If you are asked to troubleshoot why encryption doesn’t work, it may be a KMS reached quota matter.

**Uploads: S3 Multipart Uploads:**

* Recommended for files over 100 MB
* Required for files over 5 GB
* Parallelize uploads (increases efficiency) You split into parts a big file and you upload all the parts at the same time.

Possible exam scenario:

How you can get a better performance in S3 uploads, consider multi part uploads.

**Downloads: S3 Byte-Range Fetches:**

* Parallelize downloads by specifying byte ranges.
* If there is a failure in the download, it’s only for a specific byte range.

Can be use to:

* Speed up downloads
* To download partial amounts of the file (e.g., header information)

**Exam Tips:**

* **Prefix**: my\_documents/**studies/maths**/exam.docx
* You can also achieve a high number of requests per second:
  + 3,500 PUT/COPY/POST/DELETE
  + 5000 GET/HEAD
* You can get better performance by spreading your reads across different prefixes.
* **KMS:** Have in mind KMS limitation quotas.
  + Uploading/downloading will count towards KMS quota
  + Region specific, however, its either 5,000 – 10,000 or 30,000 requests per second.
  + Cannot request KMS quota increase.
* **Multipart Uploads:**
  + Using it increase performance when uploading to S3.
  + Should be used for any file over 100 MB and must be used for over 5 GB.
* **S3** **Byte-Range Fetches** (Downloads)
  + Increases performance downloading files from S3.
  + Split your file in several parts and download them at the same time.

**Backing Up Data with S3 Replication:**

Used to be called Crossed Region Replication but now you can do it in the same region as well, so it was renamed to S3 Replication.

**What is it:**  Way to replicate objects from one bucket to another.

* Versioning must be enabled on both the source and destination buckets.
* Objects in an existing bucket are not replicated automatically.
  + Once replication is turned on, all subsequent updated objects will be replicated automatically.
* Deleted markers are not replicated by default.

Possible exam scenario:

You have S3 buckets in Japan and you have earthquake problems there, what can you do to do a backup of your S3 objects?

Use S3 Replication. You can replicate them into Australia region or something similar.

**Exam Tips:**

* You can replicate objects from one bucket to another.
* Objects in an existing bucket are not replicated automatically, but you have the option to do it.
* Delete markers are not replicated by default.
* You need to enable versioning in source and destination buckets.

**EC2**