Protecting Data in Your Application

DATA AT REST & IN TRANSIT · S3 ACCESS & PROTECTION · CSE VS SSE · AWS KMS · TLS & ACM · SECRETS MANAGER & MACIE

STUDY & REVIEW

01 MUST KNOW

if you remember five things, remember these

01 **Two states, two jobs:** protect **data at rest** (stored in S3, EBS, and the like) and **data in transit** (moving across networks). Encrypting at rest adds a layer of protection even if an endpoint is compromised; data in transit is considered less secure while it moves, so protect it with TLS. Both fall in the **customer** half of the shared responsibility model.

02 **S3 is private by default:** only the resource owner (the account that created it) can access a bucket or object, and access requires AWS credentials. Two access-control mechanisms grant more: **identity-based** policies (attached to an IAM principal) and **resource-based** policies (attached to the resource, e.g. a bucket policy). An explicit **DENY** always overrides an allow. AWS recommends disabling ACLs.

03 **CSE vs SSE:** with **client-side encryption**, your application encrypts data *before* sending it to AWS — keys and algorithms known only to you, and S3 never sees plaintext. With **server-side encryption**, AWS encrypts data at its destination *after* receiving it, transparently. S3 offers three SSE options: **SSE-C**, **SSE-S3**, and **SSE-KMS**.

04 **KMS + envelope encryption: AWS KMS** creates and controls cryptographic keys in **FIPS 140-2** validated HSMs; keys never leave the service and every use is logged in **CloudTrail**. Integrated services use **envelope encryption** — encrypt data with a **data key**, then encrypt that data key with a **root (KMS) key** — which is exactly how SSE-KMS and EBS encryption work.

05 **Layer S3 protections: Block Public Access**, **Versioning**, and **Object Lock** (WORM) defend stored objects; add MFA delete, least privilege, default bucket encryption, enforced SSE on PUT, and **presigned URLs** for time-limited access. **Secrets Manager** keeps credentials out of code, and **Amazon Macie** uses ML to find sensitive data.

02 KEY TERMS

TERM	DEFINITION
Data at rest	Data stored persistently on a device or in a service such as Amazon S3 or an EBS volume
Data in transit	Data actively moving from one location to another, such as across the internet or a private network
Encryption	Using an algorithm and a key to convert plaintext into unreadable ciphertext that only the right key can reverse
Ciphertext	The unreadable output of encryption; it cannot be decrypted without the necessary key
Client-side encryption (CSE)	Your application encrypts data before submitting it to AWS; keys and algorithms are known only to you
Server-side encryption (SSE)	AWS encrypts data at its destination after receiving it; the process is transparent to the user
Envelope encryption	Encrypting plaintext with a data key, then encrypting that data key with a root key for added protection
Data key	The key that encrypts the actual object data; under SSE-KMS it is itself encrypted by a KMS key
AWS KMS	Managed service to create and control cryptographic keys, backed by FIPS 140-2 HSMs and integrated with AWS services
Customer managed key	A KMS key you create and control, with usage policies and a CloudTrail record of every use
Object Lock (WORM)	S3 write-once-read-many protection on object versions, for a fixed period or indefinitely; needs versioning
Presigned URL	A time-limited URL that grants temporary read or write access to a specific S3 object (Bucket, Key, Expires)

03 CLIENT-SIDE VS SERVER-SIDE ENCRYPTION

who encrypts, when, and where

CLIENT-SIDE ENCRYPTION (CSE)

you encrypt before it reaches AWS

- You encrypt data locally *before* it goes to AWS
- Stored already encrypted; S3 sees only ciphertext
- Keys and algorithms known only to you
- Key in AWS KMS or your app; libs: AWS Encryption SDK, S3 clients

SERVER-SIDE ENCRYPTION (SSE)

AWS encrypts after it arrives

- The receiving service encrypts data at its destination
- Transparent to the user; the service handles the keys
- Upload over HTTPS to SSE-capable services
- In S3 it's optional at upload; most SSE integrates with KMS

04 THREE TYPES OF AMAZON S3 SERVER-SIDE ENCRYPTION

mutually exclusive — pick one per object

OPTION	WHO MANAGES KEYS	KEY DETAIL
SSE-C	You provide the keys	S3 encrypts/decrypts with the key you supply per request but never stores it — only a salted **HMAC**. Lose the key → lose the object.
SSE-S3	AWS manages the keys	Each object gets a unique key, itself encrypted by a key the service regularly rotates. Uses **AES-256**.
SSE-KMS	AWS KMS keys	Adds separate key-use permissions and a **CloudTrail** audit trail; customer or AWS managed KMS keys; envelope encryption.

05 SSE-KMS UPLOAD (ENVELOPE ENCRYPTION)

how S3 encrypts an object with a KMS key

01 You upload a file to store as an encrypted object → **02** S3 requests a **data key** from AWS KMS → **03** KMS returns a plaintext data key **and** a copy encrypted under your customer managed key → **04** S3 encrypts the object with the plaintext data key, then **deletes** it → **05** Only the **encrypted** data key is kept in the object metadata

06 DATA-PROTECTION TOOLBOX AT A GLANCE

one-line job of each — exam gold

TOOL	WHAT IT DOES	REMEMBER
S3 Block Public Access	Overrides ACLs and policies to keep buckets/objects private	4 settings; most restrictive of account/bucket/access point wins
S3 Versioning	Keeps every version; recover from delete or overwrite	Can only be suspended, never turned off
S3 Object Lock	WORM immutability for a period or indefinitely	Versioned buckets only; governance vs compliance
AWS KMS	Create and control keys in FIPS 140-2 HSMs	Keys never leave; CloudTrail logs all use
AWS Certificate Manager	Provision, deploy, auto-renew SSL/TLS certificates	In transit; LB, CloudFront, API Gateway
ACM Private CA	Private PKI with root and subordinate CAs	Trusted only inside your org
Secrets Manager	Store and auto-rotate credentials and secrets	No hardcoded secrets; AWSPENDING/AWSCURRENT
Amazon Macie	ML discovery/classification of sensitive data	S3 only; flags public/unencrypted/shared

07 EXAM TRAPS

where the knowledge check gets you

- × 'An explicit ALLOW in my identity policy guarantees access.' — No. An explicit **DENY** in a resource-based policy overrides it: in the deck's example, Bob's identity policy allows PUT on bucket X, but the bucket policy denies PUT, so he cannot PUT.
- × 'S3 Versioning can be turned back off.' — Once enabled, versioning can only be **suspended**, never returned to an unversioned state.
- × 'Object Lock works on any bucket.' — It works **only in versioned buckets**, because it protects object versions using the WORM model.
- × 'Governance and compliance retention modes are basically the same.' — In **compliance** mode no one — not even the account **root user** — can overwrite or delete a protected version; governance mode allows users with special permissions.
- × 'With SSE-C, AWS keeps a copy of my key.' — S3 stores only a salted **HMAC** of the key, never the key itself. Provide the key with each request; lose it and the object is unrecoverable.
- × 'SSE-S3 and SSE-KMS are equivalent.' — SSE-KMS adds separate key-use **permissions**, a **CloudTrail** audit trail of who used the key and when, and customer/AWS managed KMS keys; SSE-S3 keys are fully AWS-managed with no separate audit trail.
- × 'The plaintext data key is stored alongside the object.' — Under SSE-KMS, S3 encrypts the object with the plaintext data key and then **deletes** it; only the **encrypted** data key stays in object metadata.
- × 'SSL/TLS protects my data at rest.' — TLS (HTTPS) protects data **in transit**; encryption such as SSE or CSE protects data **at rest**. On the sample exam question, server-side encryption in S3 is what protects data at rest.

08 SELF-QUIZ

answers are upside-down below

Q1 Which S3 access-control mechanism is attached directly to the resource rather than to an IAM identity?

Q2 Which encryption approach has your application encrypt data before it is ever sent to AWS, with keys known only to you?

Q3 Of the three S3 SSE options, which one has you supply the key on each request while S3 keeps only a salted HMAC of it?

Q4 Which S3 SSE option uses AWS KMS keys and adds a CloudTrail audit trail of key usage?

Q5 What is the practice of encrypting data with a data key and then encrypting that data key with a root key called?

Q6 S3 Object Lock can be used only in buckets that have which feature enabled first?

Q7 Which Object Lock retention mode blocks overwrite and deletion by every user, including the account root user?

Q8 Which service issues, deploys, and automatically renews public and private SSL/TLS certificates so expirations don't cause outages?

Q9 Which service stores database credentials and API keys so they aren't hardcoded, and rotates them on a schedule?

Q10 Which service uses machine learning and pattern matching to discover and classify sensitive data such as PII in S3?

ANSWER KEY (rotate the page)

Q1 A resource-based policy (e.g. a bucket policy) **Q2** Client-side encryption (CSE) **Q3** SSE-C **Q4** SSE-KMS **Q5** Envelope encryption **Q6** Versioning **Q7** Compliance mode **Q8** AWS Certificate Manager (ACM) **Q9** AWS Secrets Manager **Q10** Amazon Macie