Understanding AWS KMS (Key Management Service)

This material is entirely my own and has not been reviewed by AWS. It should not be considered canonical and might contain errors. It is provided “as is.” Think of it as my scribbles on the board in the classroom.

www.linkedin.com/in/carel-grove

Agenda

- Data encryption and Hardware Security Modules (HSMs)
- AWS KMS
- AWS KMS Keys
- KMS:GenerateDataKey
- KMS:Decrypt

Data encryption

And hardware security modules (HSMs)

Why isn’t everything encrypted everywhere?

- Because it is hard?
- The encryption part is easy.
- The hard part is the keys
  - Generating keys
    - in software (PRNG)
    - in hardware (HRNG)
  - Storing keys
    - On the server’s disk drive
    - In secure hardware

Key generation and storage in hardware

- Smart Card
- SIM chip (a type of smart card)
- Trusted Platform Module (TPM)
- Apple Secure Enclave & T2 chip
- Hardware Security Module

Some Countermeasures

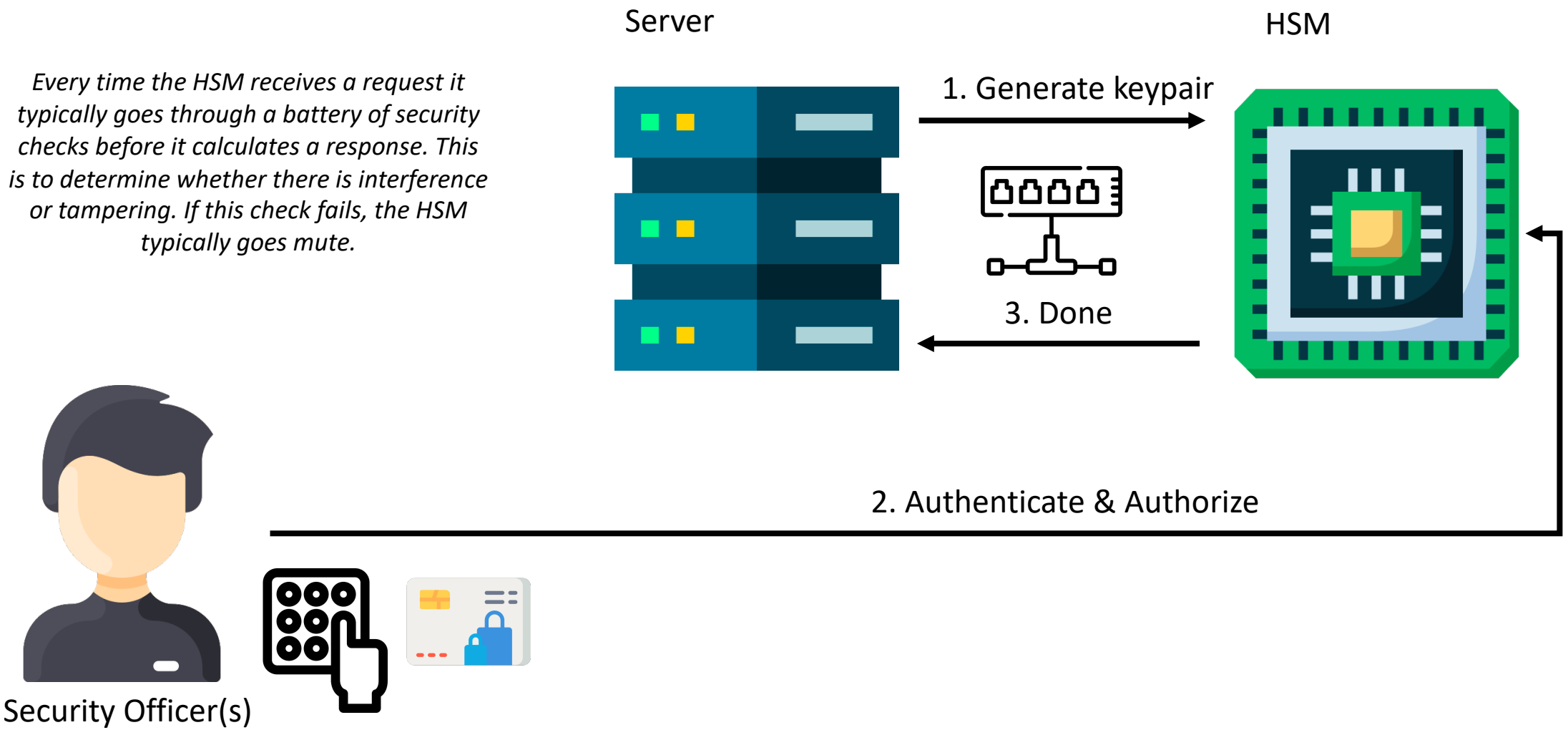
It is a digital safe

- **Tamper-responsive design:** respond to tampering events by erasing sensitive data
- **Environmental Sensors**: These sensors can detect changes in temperature, humidity, light, or pressure that might indicate a tampering attempt. For example, if someone tries to freeze a device to slow down its internal clock, a temperature sensor might detect the change and trigger a defensive response.
- **Active Meshes**: These are circuits or wires that surround the device. If the mesh is cut or broken – as might happen during an attempt to drill into the device – the circuit is broken, and the device can trigger a defensive response.
- **Voltage and Clock Monitoring**: Some devices monitor their own power supply and internal clock. If the power supply or clock signal is interrupted or altered, this might indicate a tampering attempt.
- **Chain of custody** from manufacture to customer handover: Like evidence from a crime scene. Prevent tampering with device between manufacture and when a customer starts using it.

Communication – Request / Response

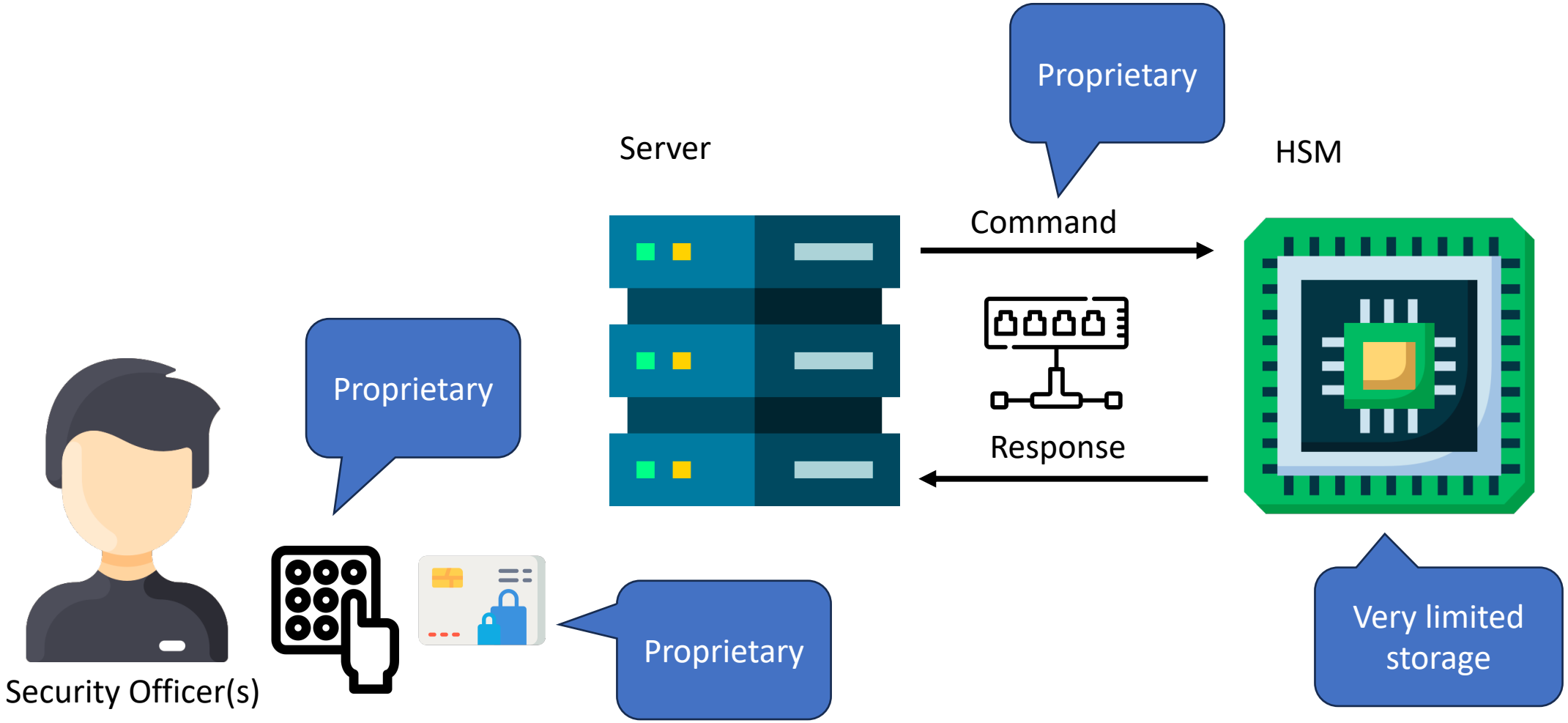
Server and HSM communicate with commands like terminal and mainframe

Every time the HSM receives a request it typically goes through a battery of security checks before it calculates a response. This is to determine whether there is interference or tampering. If this check fails, the HSM typically goes mute.



Challenges with HSMs

Also, deploying this at scale, running for years is a very challenging task



AWS KMS

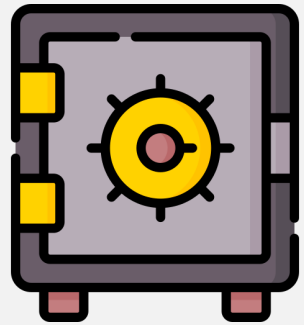
Four of the main actors

Symmetric envelope encryption used by over 120 AWS services

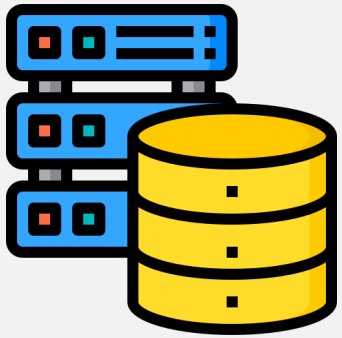
Over 120 AWS services

AWS KMS Service

AWS KMS HSM



AWS Storage



Two worlds

"Normal World"

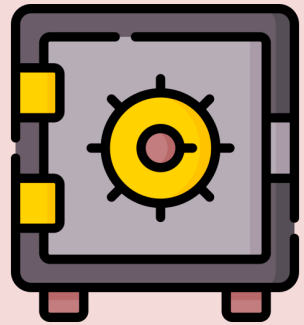
Over 120 AWS services

AWS KMS Service

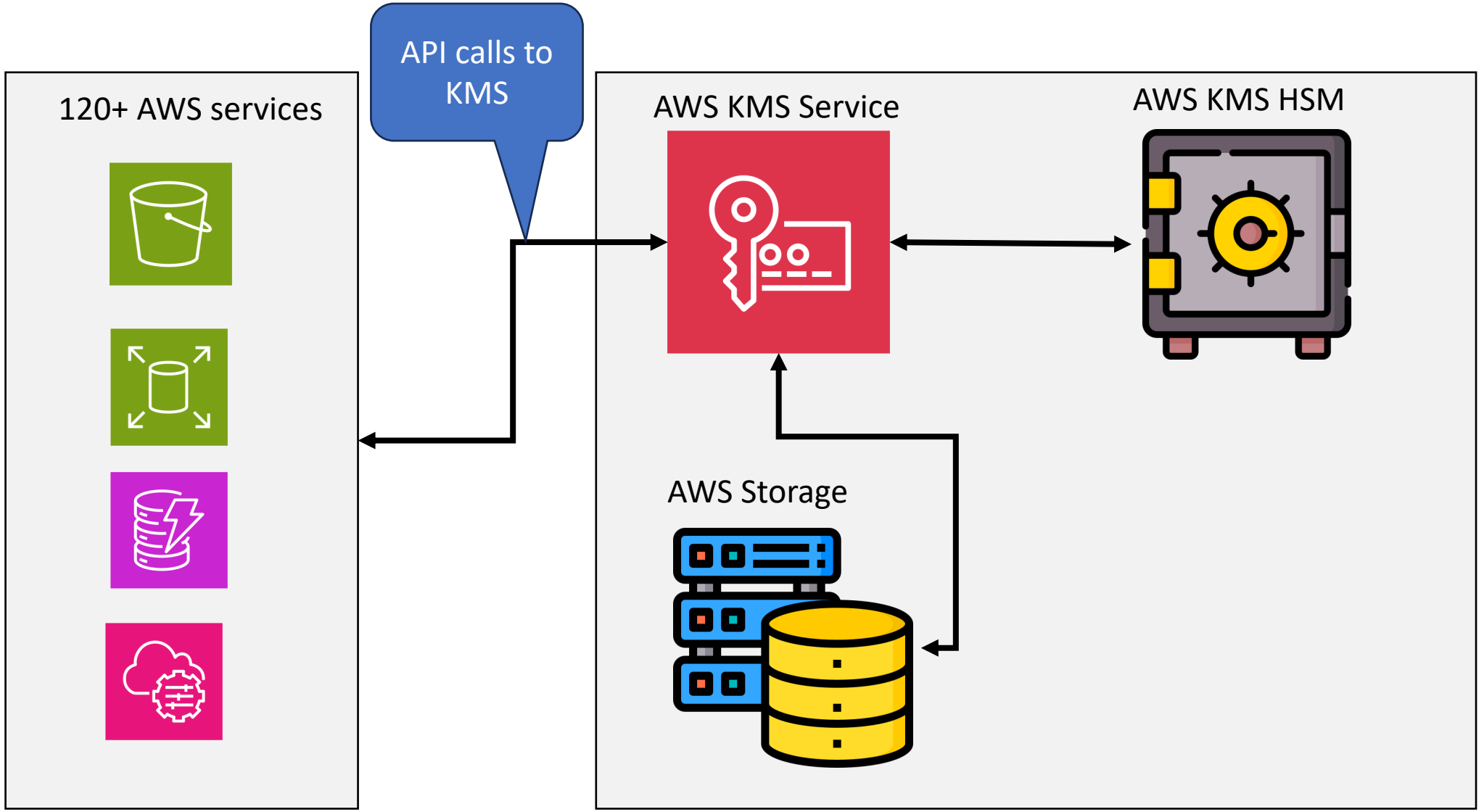
AWS Storage

"Crypto world"

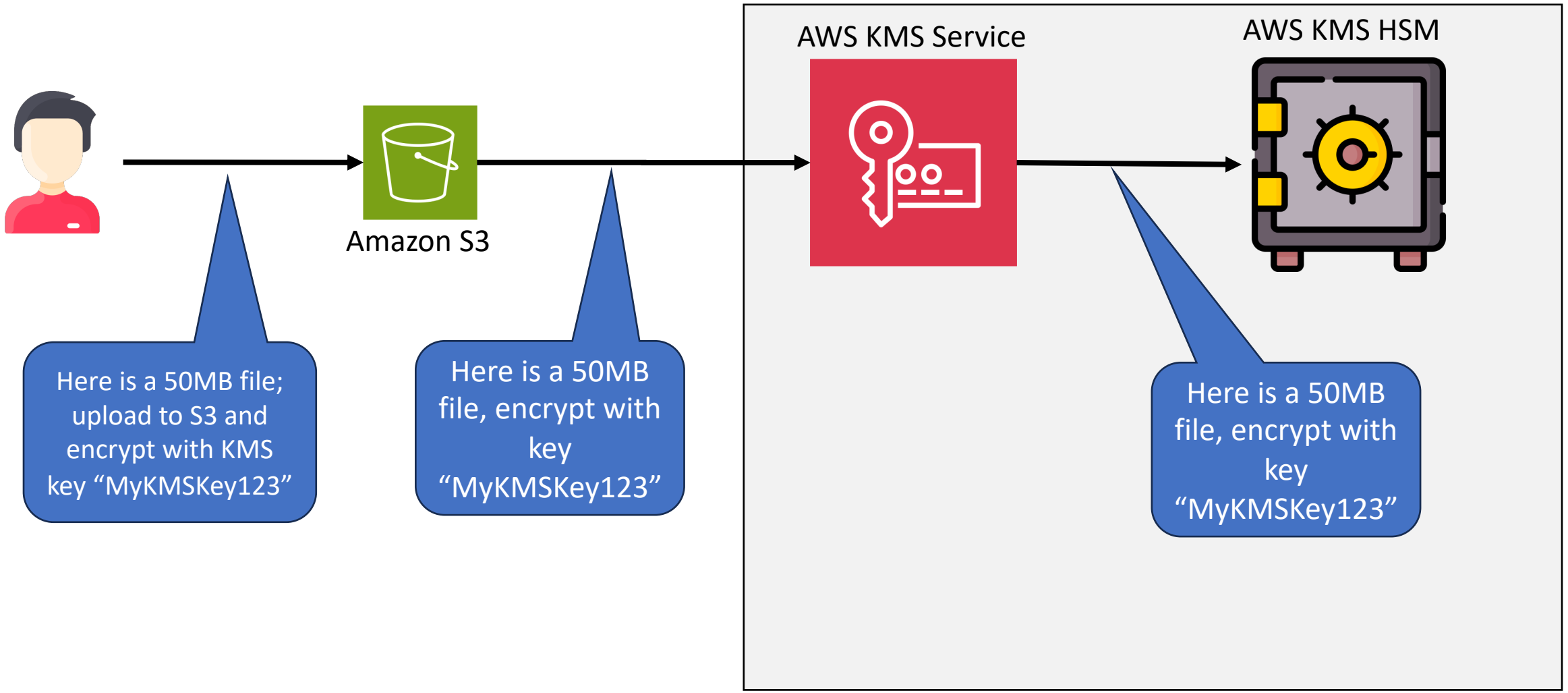
AWS KMS HSM



Comms between the main actors



Here is how is DOESN'T Work 1/3

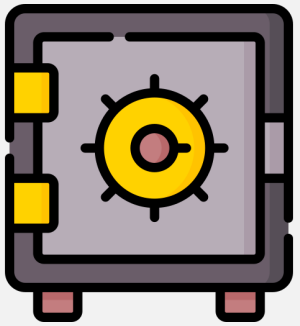


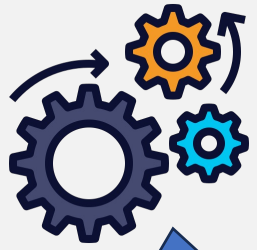
Here is how is DOESN'T Work 2/3

Amazon S3

AWS KMS Service

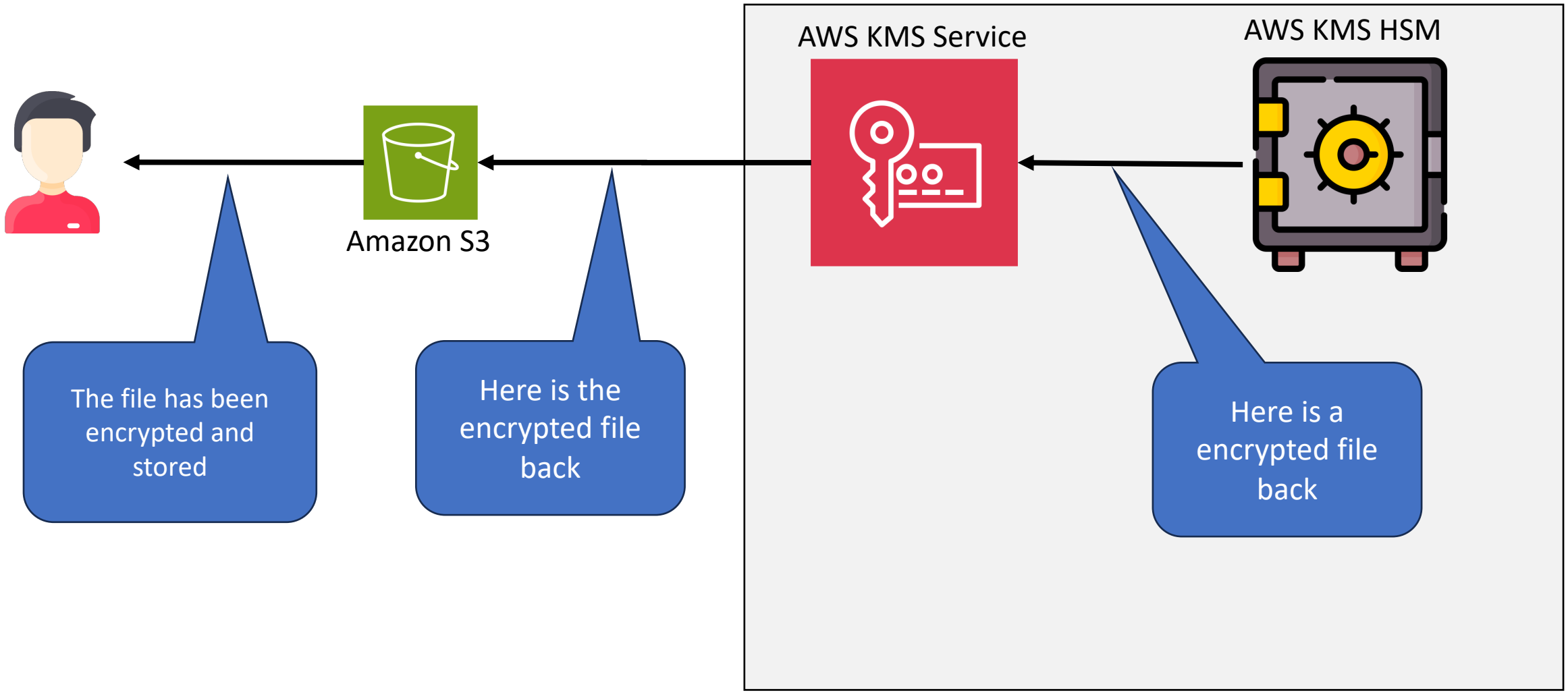
AWS KMS HSM





Encrypting
50MB file

Here is how is DOESN'T Work 3/3



AWS KMS Keys

Well, the main ones anyway…

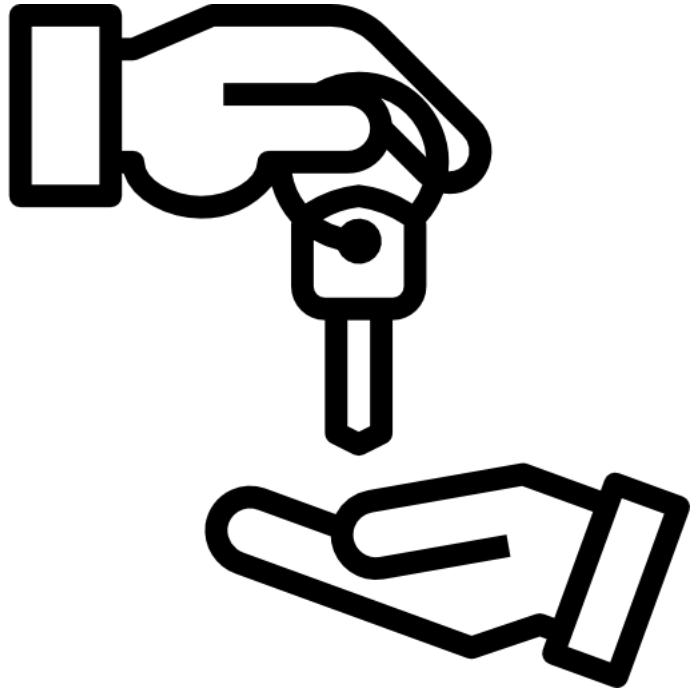
To understand how it actually works, some details on keys

There are more keys but let's keep it as simple as possible

- KMS Key (not a cryptographic key!)
- HBK (HSM-Backing Key – referred to as “key material”)
- EKT (Exported Key token)
- DK (Data Key a.k.a. Customer Data Key)
- DEK (Data Encryption Key)

DEMO

Create a KMS key



What is a KMS Key

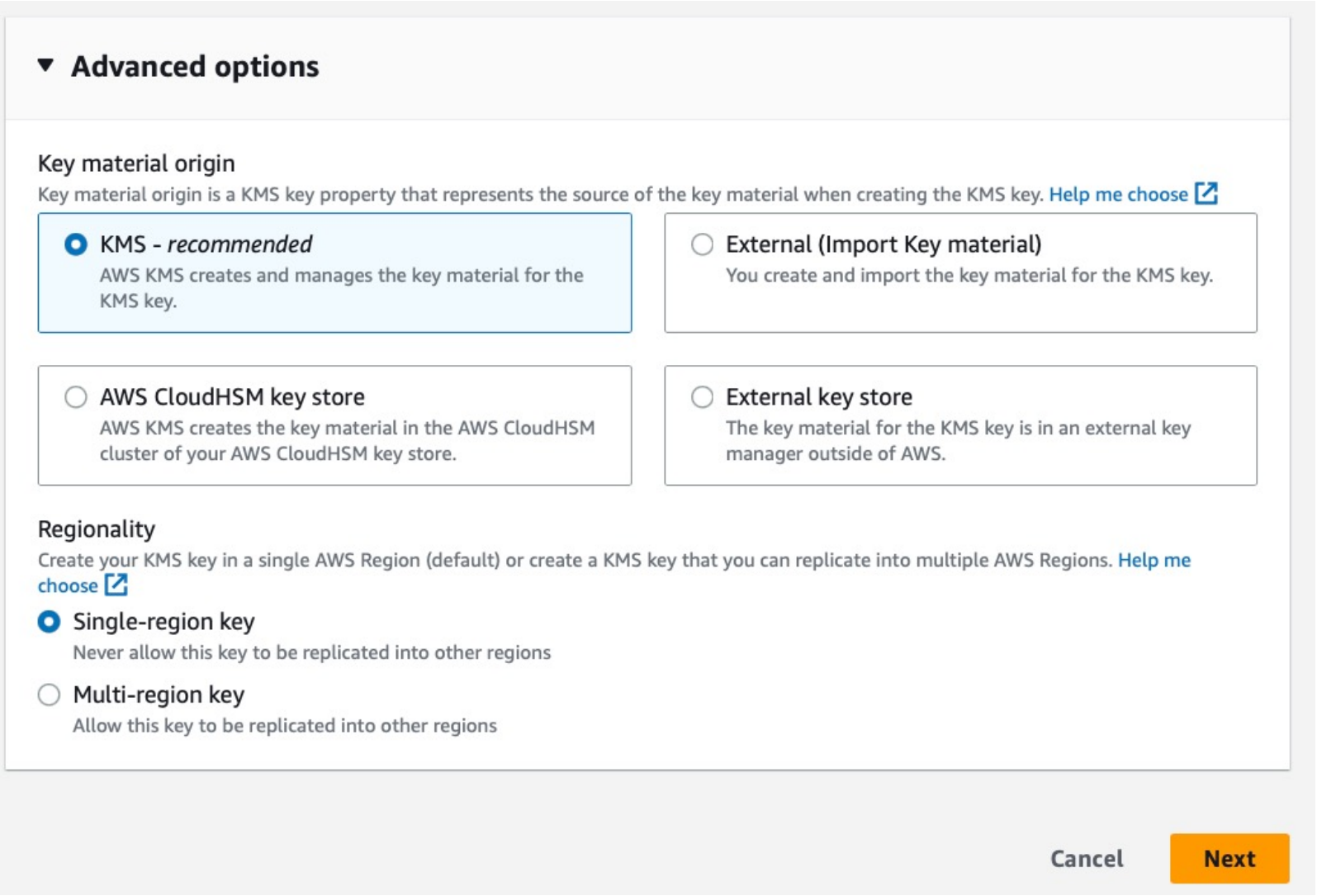
- Not a cryptographic key
- Think of it as an entry in a database in the AWS KMS *Service* (not in the AWS KMS HSM!) with a name (Alias) and an ARN plus metadata (attributes, policy, etc.)
- It *points* to an actual key (EKT and HBK - more later)
- *The KMS "key" is not used to encrypt or decrypt anything*
- (Nor is it used to *derive* another key to encrypt or decrypt anything)

HBK (HSM Backing Key) - the so-called “key material”

- The HBK is an actual symmetric cryptographic key and is associated with the KMS “Key”
- It is generated inside the AWS KMS HSM (as part of the same operation when you create a KMS Key)
- After being generated, the HBK is in turn encrypted by the AWS KMS HSM and *exported* to AWS Storage.
- In its encrypted form in AWS Storage it is called the EKT (Exported Key Token)
- This prevents the AWS KMS HSMs from filling up almost immediately with a gazillion keys.
- When the HBK is needed for an operation, the EKT is imported into the AWS KMS HSM, and decrypted for use.
- It never leaves the AWS KMS HSM in unencrypted form
- Details about the HBK and EKT are stored with the KMS key as metadata attributes of the KMS key

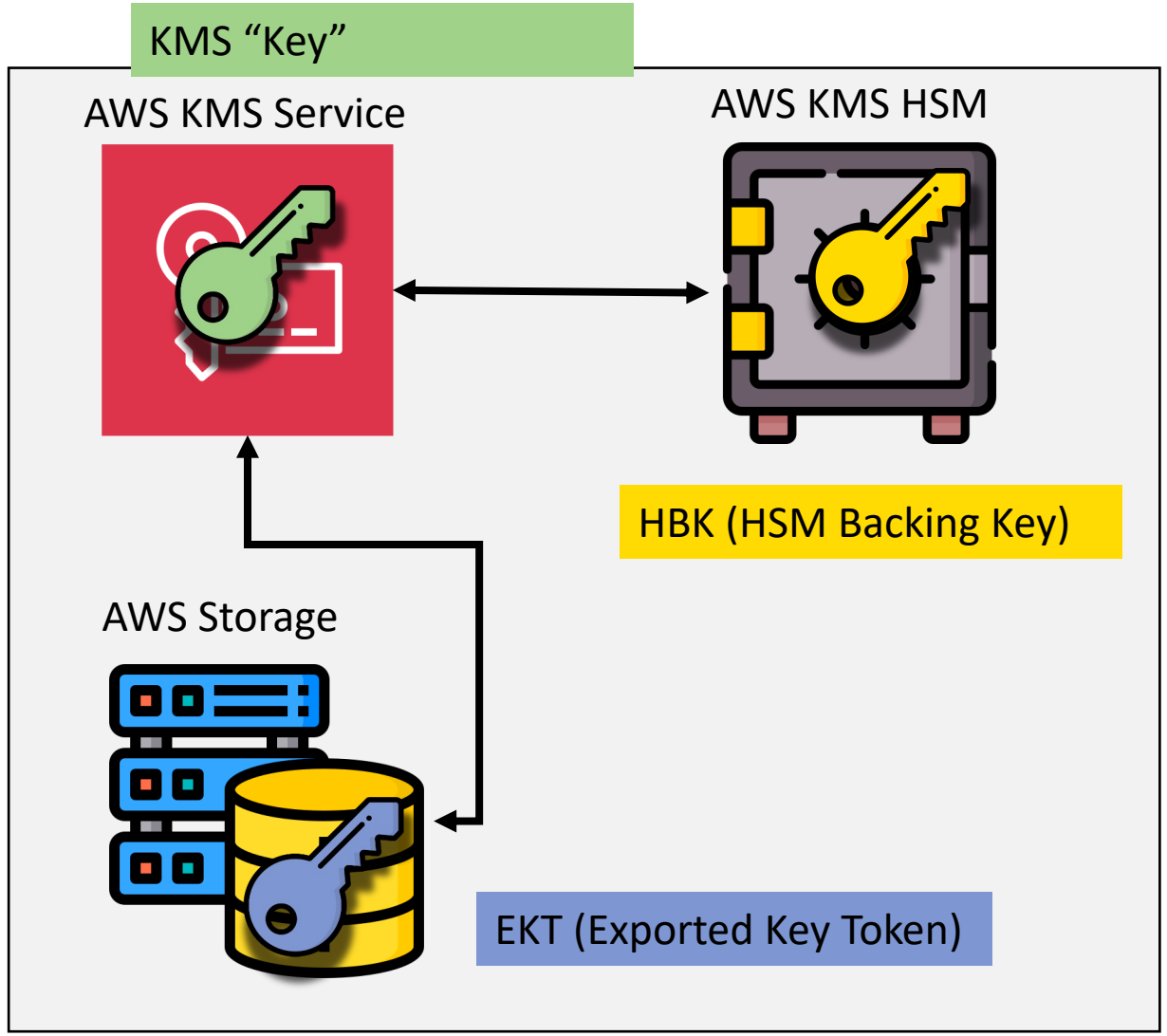
There are more sources of “key material”

But this (AWS KMS HSM) is the recommended use case



When creating a KMS key you can select where you want the corresponding key material for that KMS key to come from

Where the keys “live”

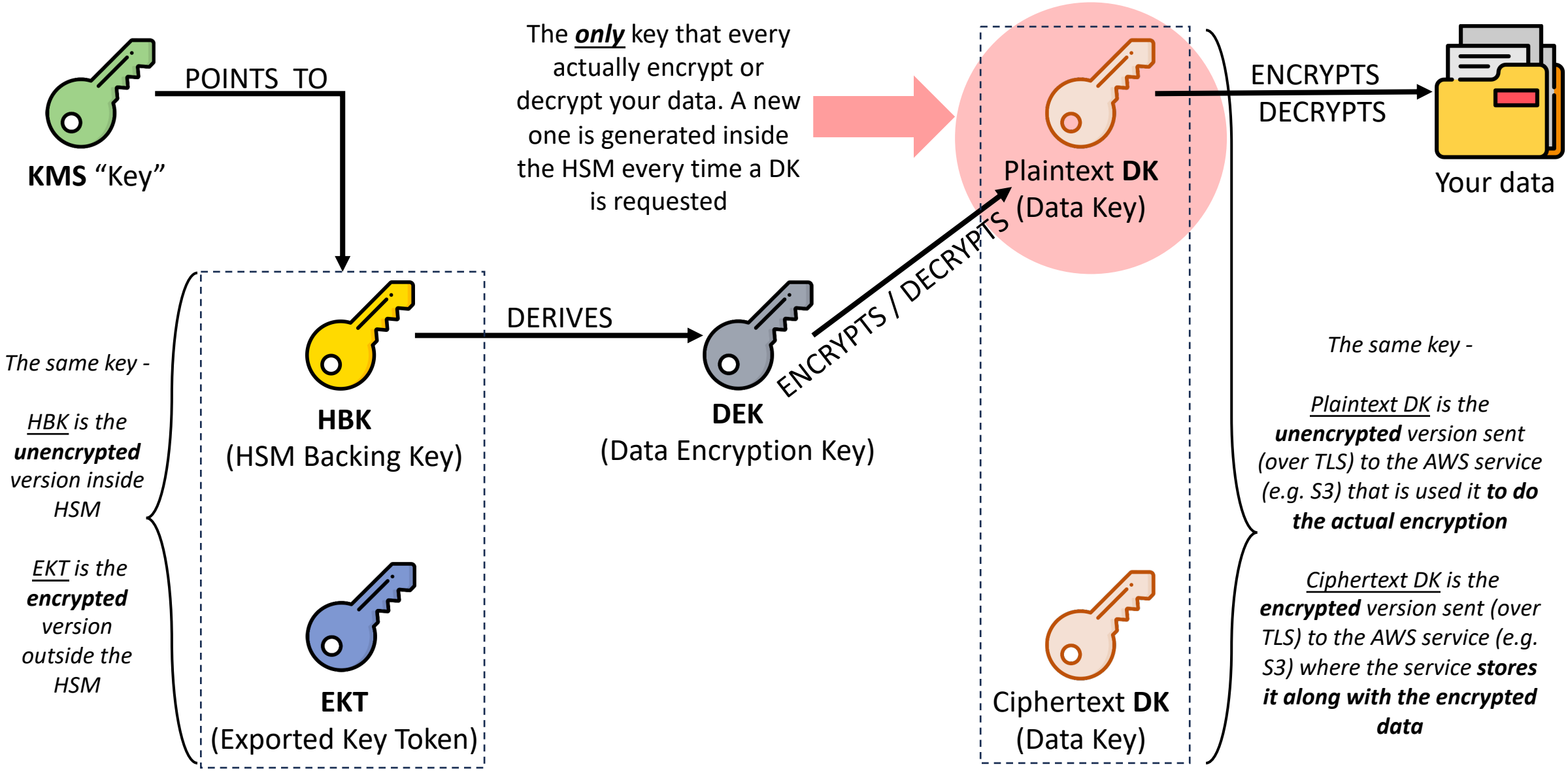


- The HBK is only in the KMS when initially generated or when it needs to be used.
- When the HBK is required for an operation, the AWS KMS Service imports the EKT into the AWS KMS HSM.
- The AWS KMS “Key” has metadata that reference the corresponding EKT / HBK

This might surprise you, but

- None of these keys (KMS “Key”, HBK or EKT) encrypt / decrypt any of your data.
- That is right, these keys *do not encrypt or decrypt your data*
- Your data gets encrypted / decrypted by yet another key - The Data Key (DK) *also known as the CDK (Customer Data Key) – more in a moment.*
- The job of the HBK is NOT to encrypt your data, but to derive a 5th (!) key - the DEK (Data Encryption Key).
- The DEK *does not encrypt your data either*, it encrypts the DK

Thus… Envelope encryption – the full story



The DK (Data Key) a.k.a. CDK (Customer Data Key)

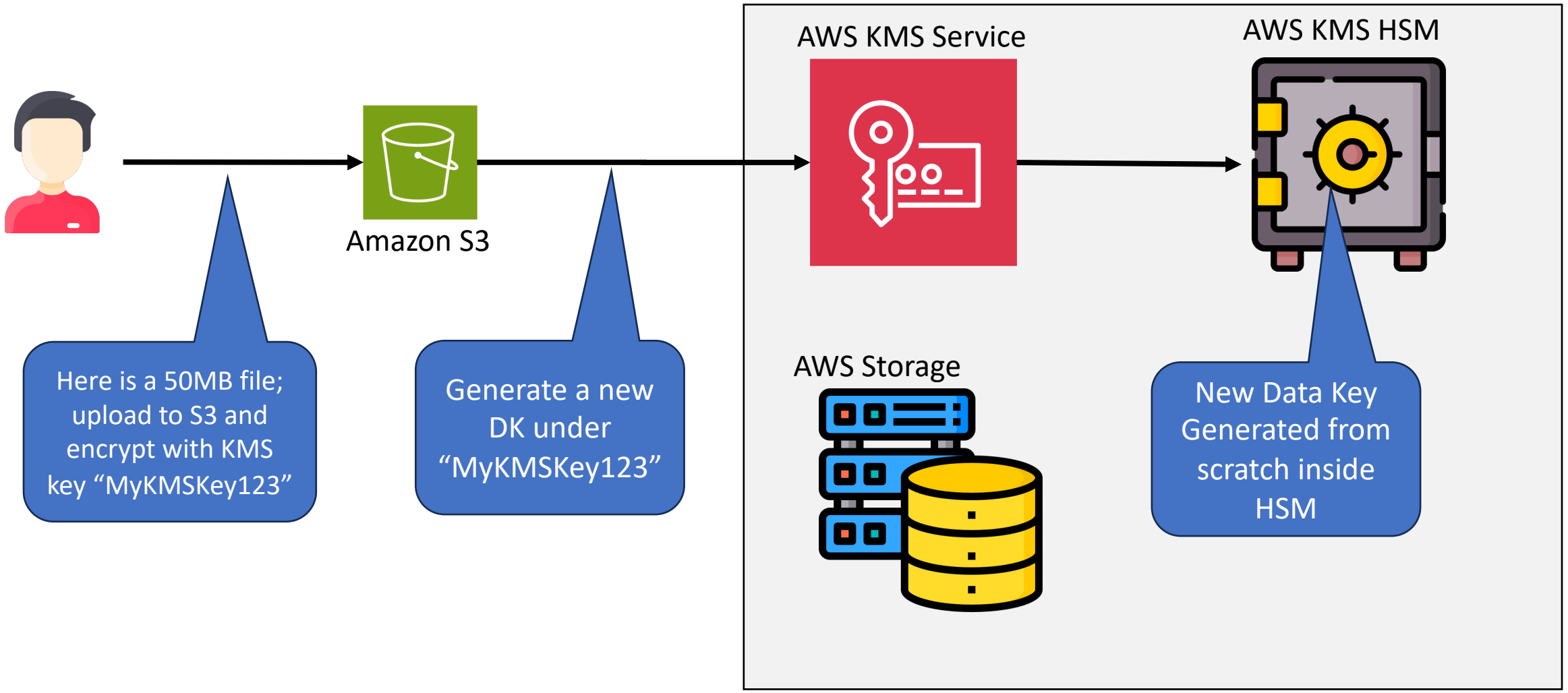
- This is the actual key that encrypts / decrypts your data.
- Every time something needs to be encrypted using KMS, the AWS service that needs to encrypt it (such as S3) calls KMS and asks KMS to generate a new DK
- The DK is generated inside the AWS KMS HSM from scratch every time it receives a new request to generate a DK.
- The DK has no cryptographic relationship with any other the other keys, you get a brand new unique one every time KMS is asked to generate one.
- The *actual* encryption / decryption happens on the AWS service that requested the key (such as Amazon S3 or 120 + other AWS services that work with KMS)

SSE-KMS Demo at an AWS service level (EC2, S3)

- Show how KMS Key is used to encrypt EBS volumes in EC2
- Create S3 bucket and set bucket to encrypt by default with SSE-KMS
- Upload an object to the bucket

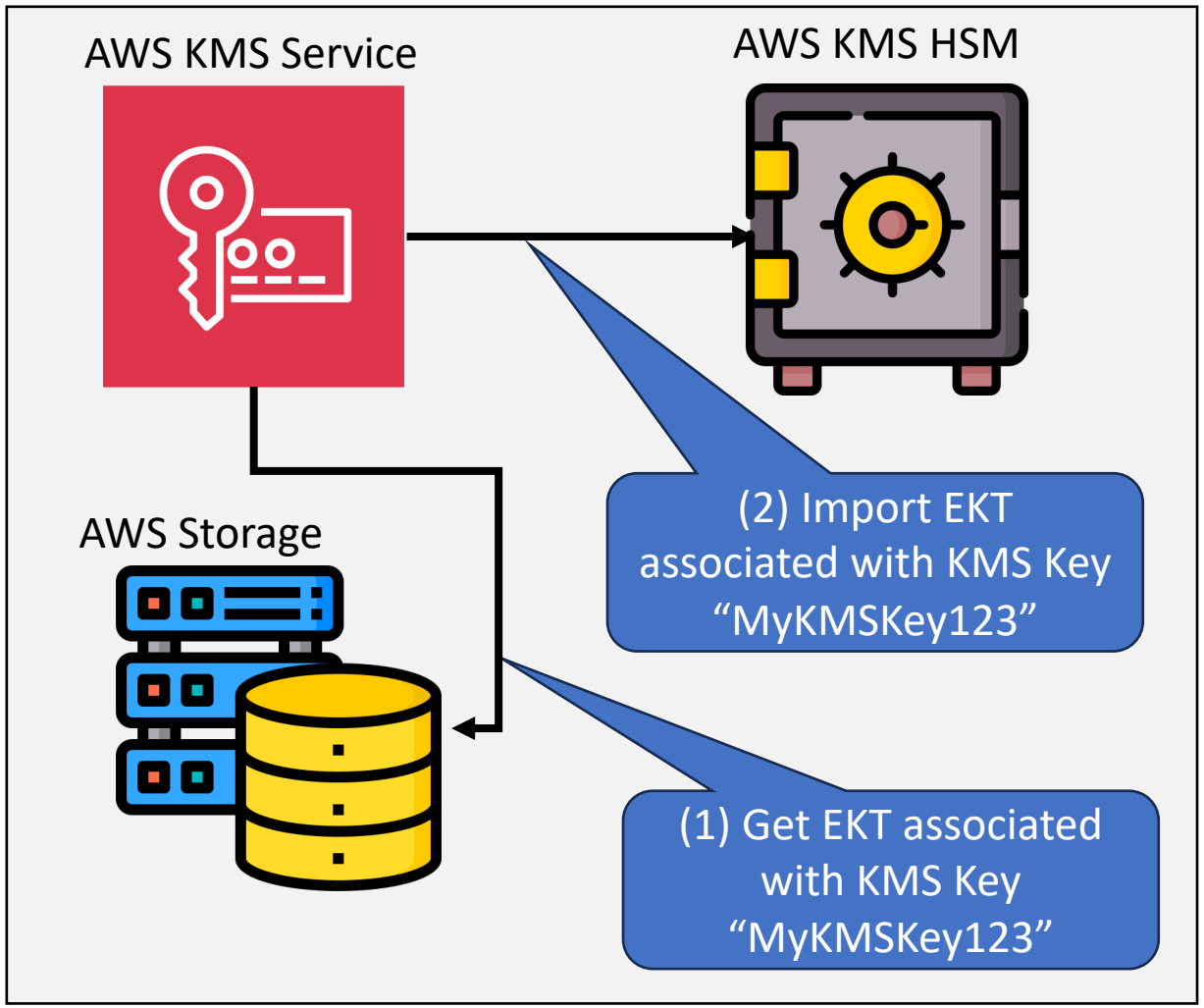
KMS:GenerateDataKey

Here is how is encryption ACTUALLY Works 1

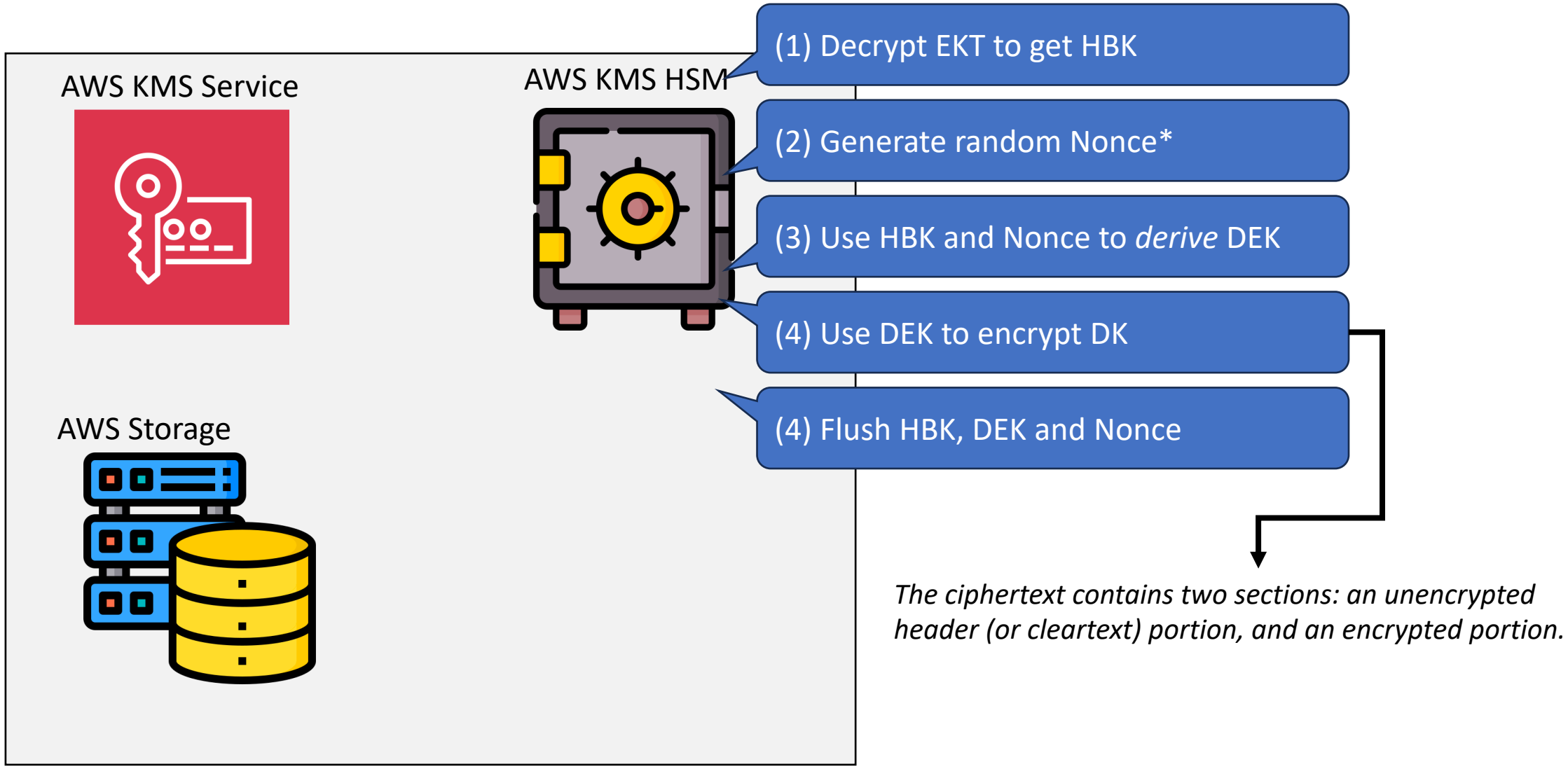


Here is how is encryption ACTUALLY Works 2

Amazon S3



Here is how is encryption ACTUALLY Works 3

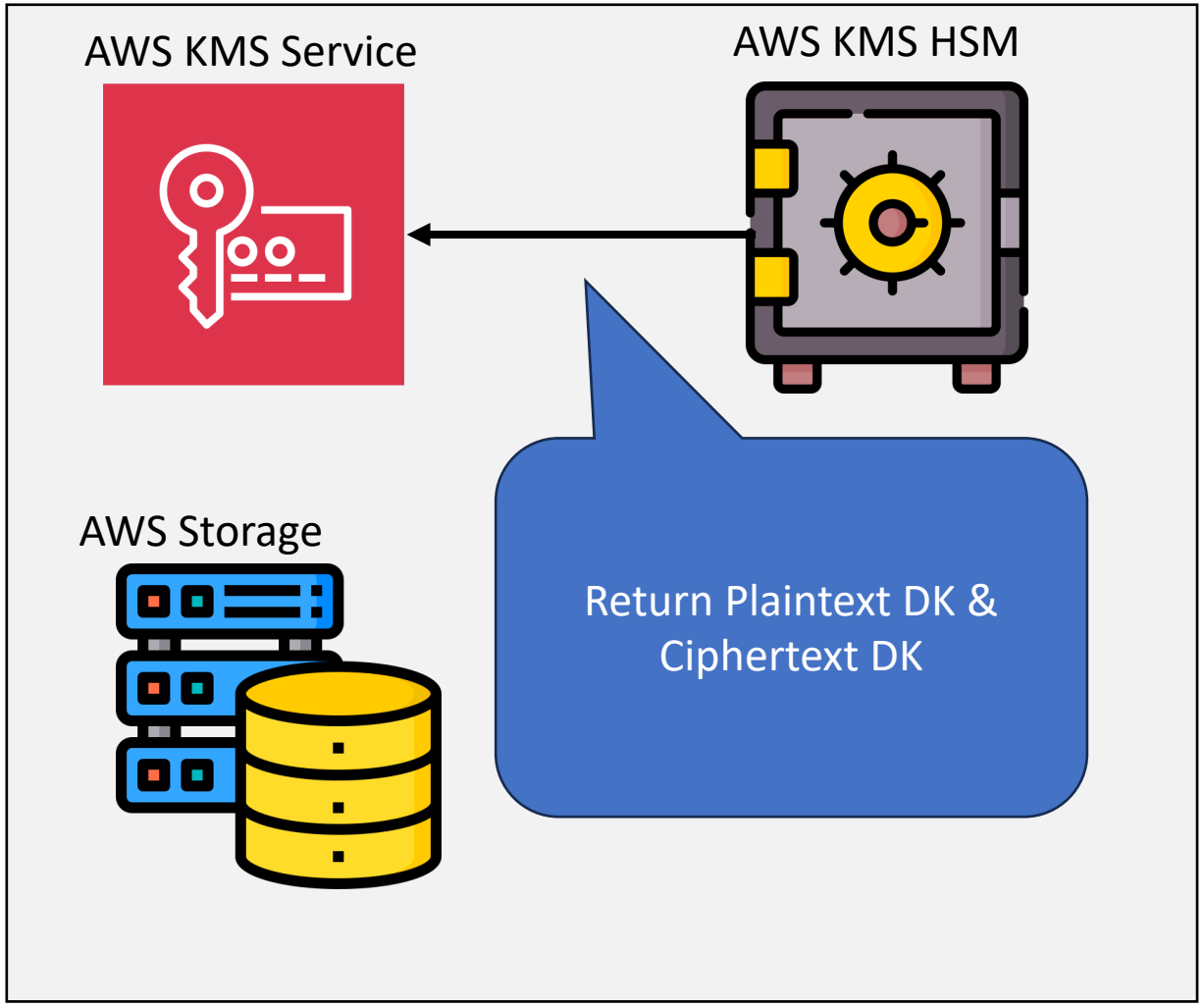


The ciphertext contains two sections: an unencrypted header (or cleartext) portion, and an encrypted portion.

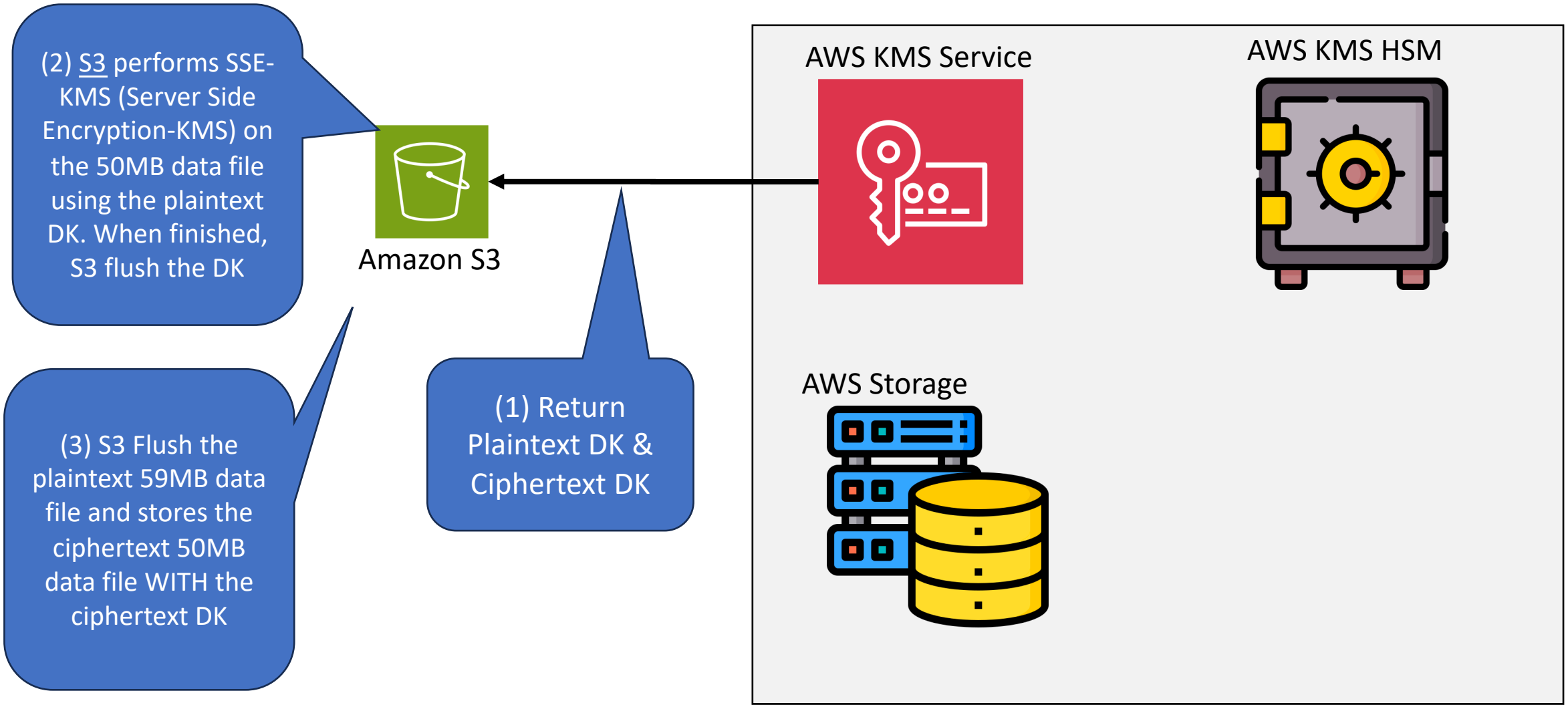
* Nonce = a number that is used only *once*

Here is how is encryption ACTUALLY Works 4

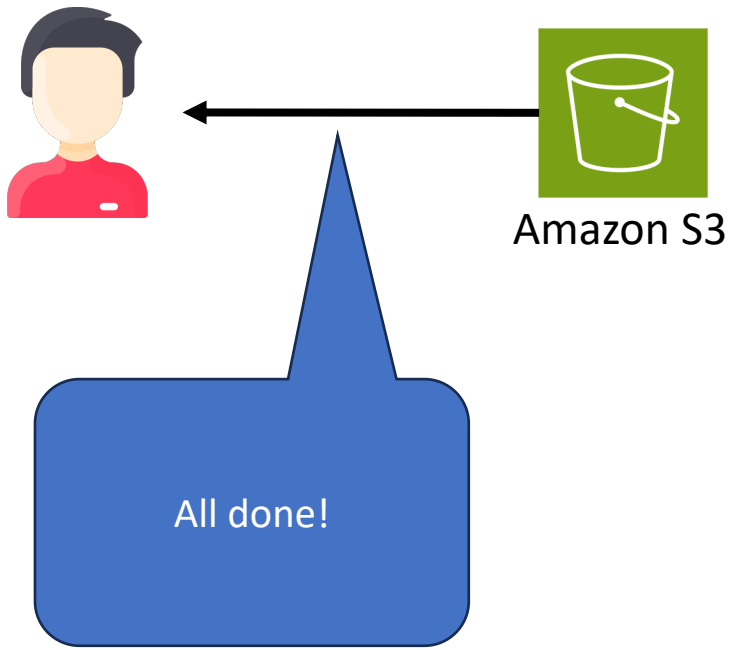
Amazon S3

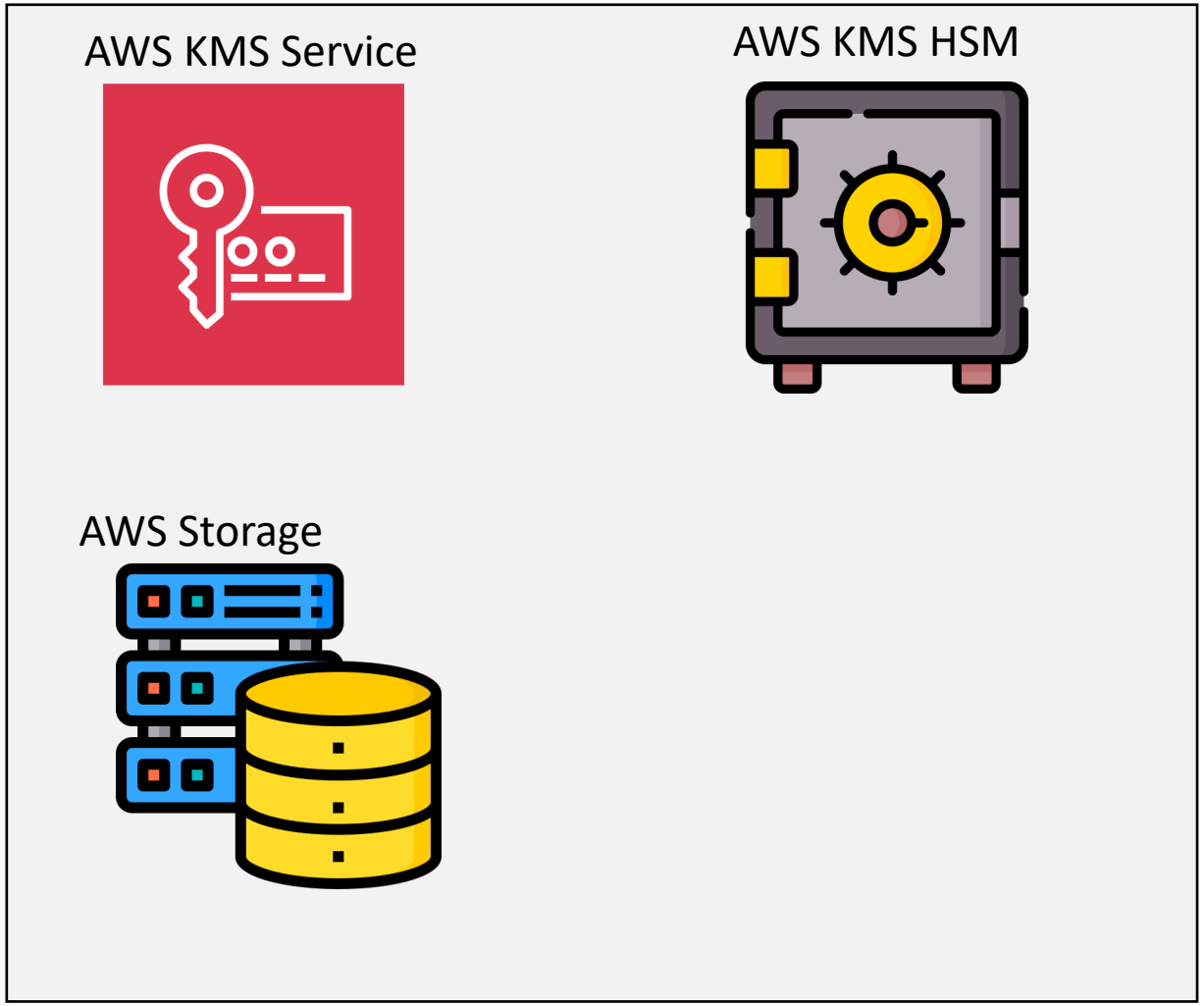


Here is how is encryption ACTUALLY Works 5



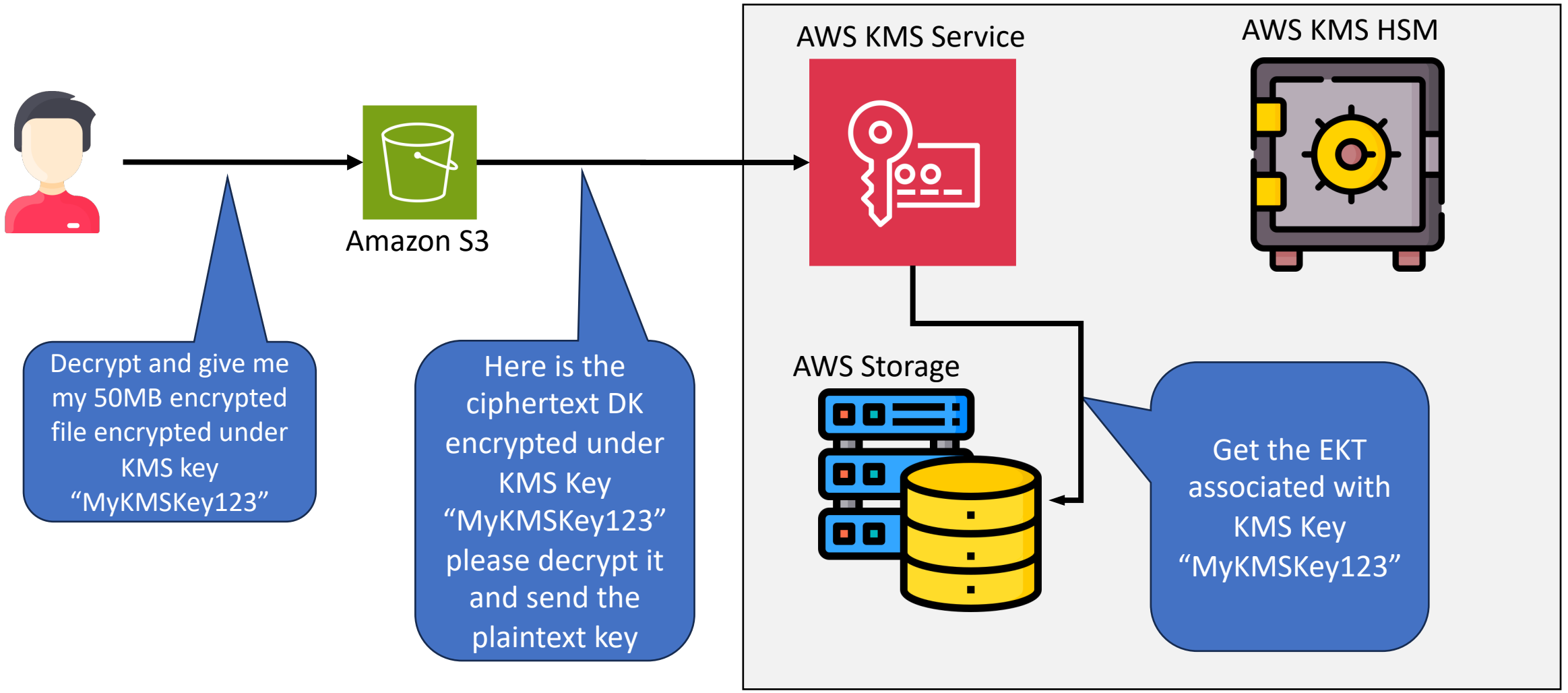
Here is how is encryption ACTUALLY Works 6





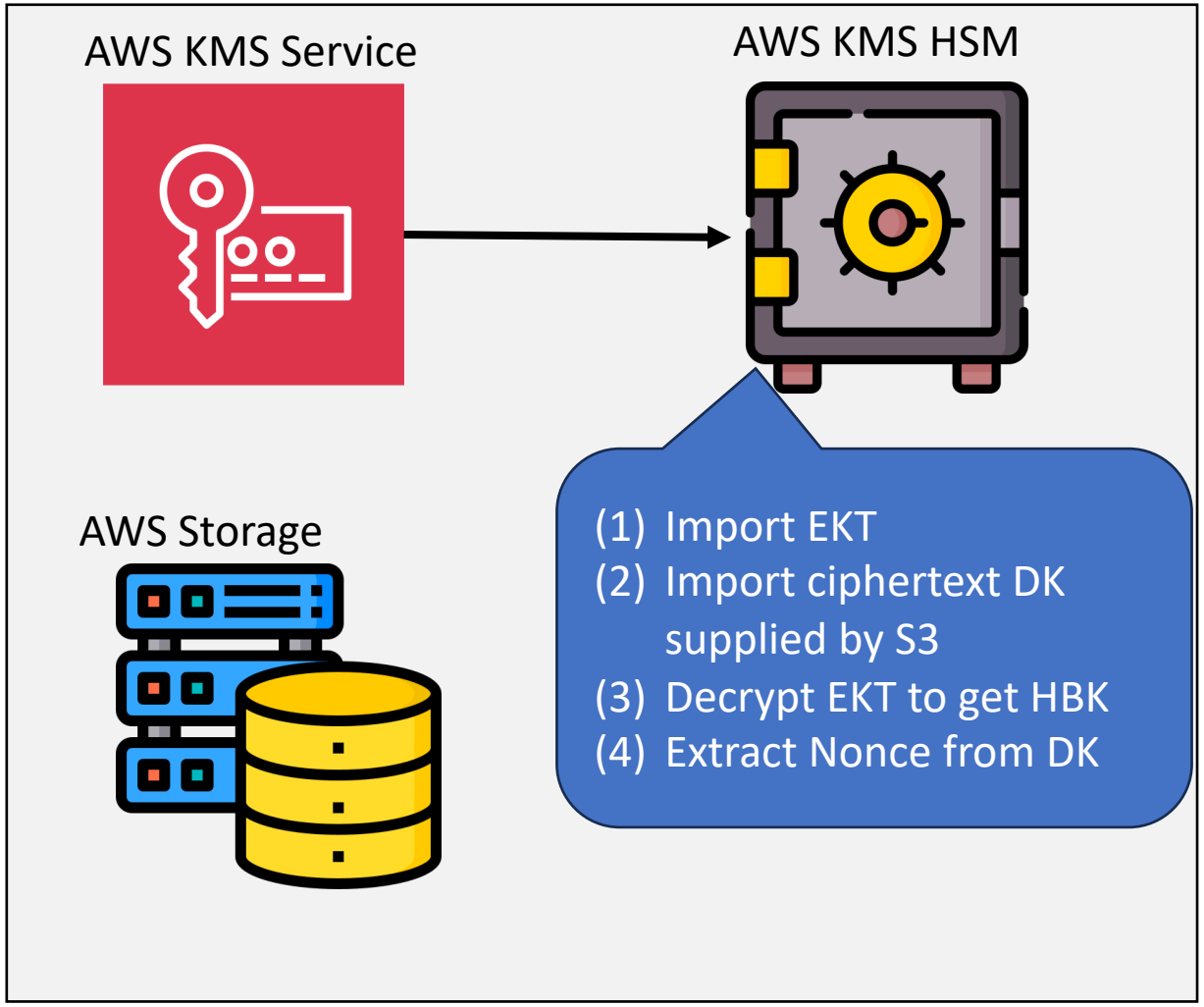
KMS:Decrypt

Here is how is decryption ACTUALLY Works 1



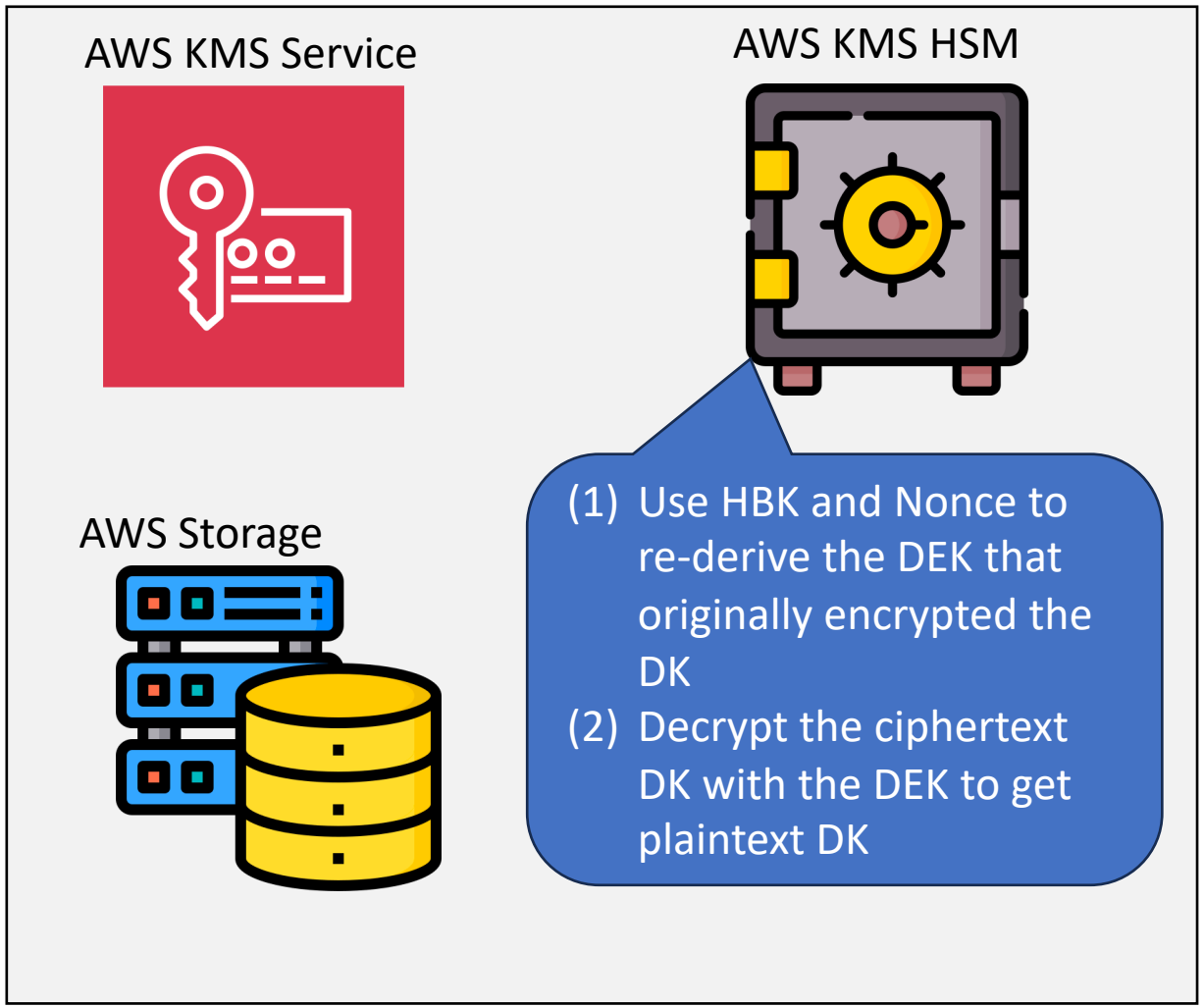
Here is how is decryption ACTUALLY Works 2

Amazon S3

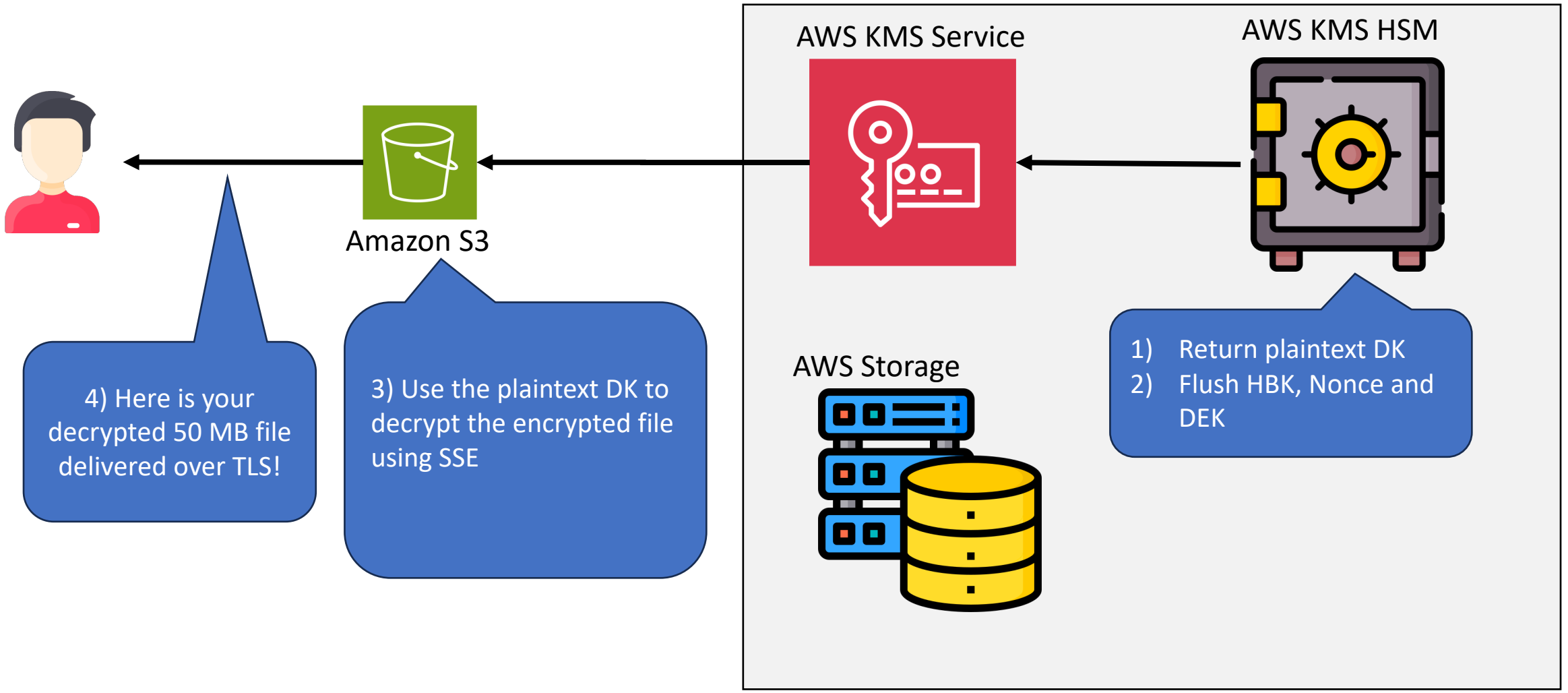


Here is how is decryption ACTUALLY Works 3

Amazon S3



Here is how is decryption ACTUALLY Works 4



What was not covered here

There is a lot of depth but simply not enough time to cover in one session

- Details on symmetric encryption, asymmetric encryption, hashing and HMAC
- AWS Policies – KMS Key policy and how it is used with IAM policies
- Grants – Delegated permission to use a KMS key
- Encryption Context – additional contextual info about the data added to the Ciphertext DK
- The key (Domain Key) that encrypts the HBK to make it an EKT
- Deactivated HBKs (Where your HBK goes when you rotate it)
- Everything above the Domain Key (yes, there's even more!)
- Multi-Region Keys (MRK)
- Key Rotation (That actual thing that is rotated is the HBK)
- Sources of key material other than the AWS KMS HSM
    - AWS CloudHSM Key Store
    - External (Import key material from your on-prem HSM)
    - External Key Store (double-encrypt the DK in your on-prem datacenter!)

AWS KMS HSM

https://csrc.nist.gov/CSRC/media/projects/cryptographic-module-validation-program/documents/security-policies/140sp4523.pdf

Figure 1 – Cryptographic Module Boundary (Front)

Figure 2 - Cryptographic Module Boundary (Back)