AWS KMS
Final Project Presentation

Pooja Desur

RM: Subrata G

Problem Statement

- Research and implement through Springboot a demo on the functionalities of AWS KMS

What is KMS?

- AWS Key Management Service (AWS KMS) used to encrypt and decrypt data useful for AWS services
- Generate data keys usable outside AWS KMS
- An *AWS KMS key* is a logical representation of a cryptographic key that contains metadata -
  - Key ID, spec, usage etc
  - Key material
- Symmetric or Asymmetric
  - Symmetric KMS keys and private Asymmetric KMS keys don't leave AWS KMS unencrypted

Customer Managed Keys (CMK)	AWS managed keys
• KMS keys that you can create • Establish and maintain key policies • enable and disable them • rotate their material • Schedule for deletion	• AWS managed keys are created, managed, and used by an AWS service integrated with AWS KMS • Cannot change properties • cannot rotate them • Cannot schedule to delete

Customer managed keys

- Once created, can be used to encrypt and decrypt data only in AWS KMS – does not leave AWS KMS unencrypted
- Commonly used – 256 bit symmetric encryption key
- Can generate Data keys which can be used outside AWS KMS
- Key material not accessible

Customer managed keys (1)

Filter keys by properties or tags

Aliases	Key ID	Status	Key spec	Key usage
demo-1	3d230b55-c005-4c58-b9a3-70617078d0b2	Enabled	SYMMETRIC_DEFAULT	Encrypt and decrypt

General configuration

Alias	Status	Creation date
demo-1	Enabled	Jul 13, 2022 24:58 GMT+5:30
ARN	**Description**	**Regionality**
arn:aws:kms:us-east-2:431294699696:key/3d230b55-c005-4c58-b9a3-70617078d0b2	-	Single Region

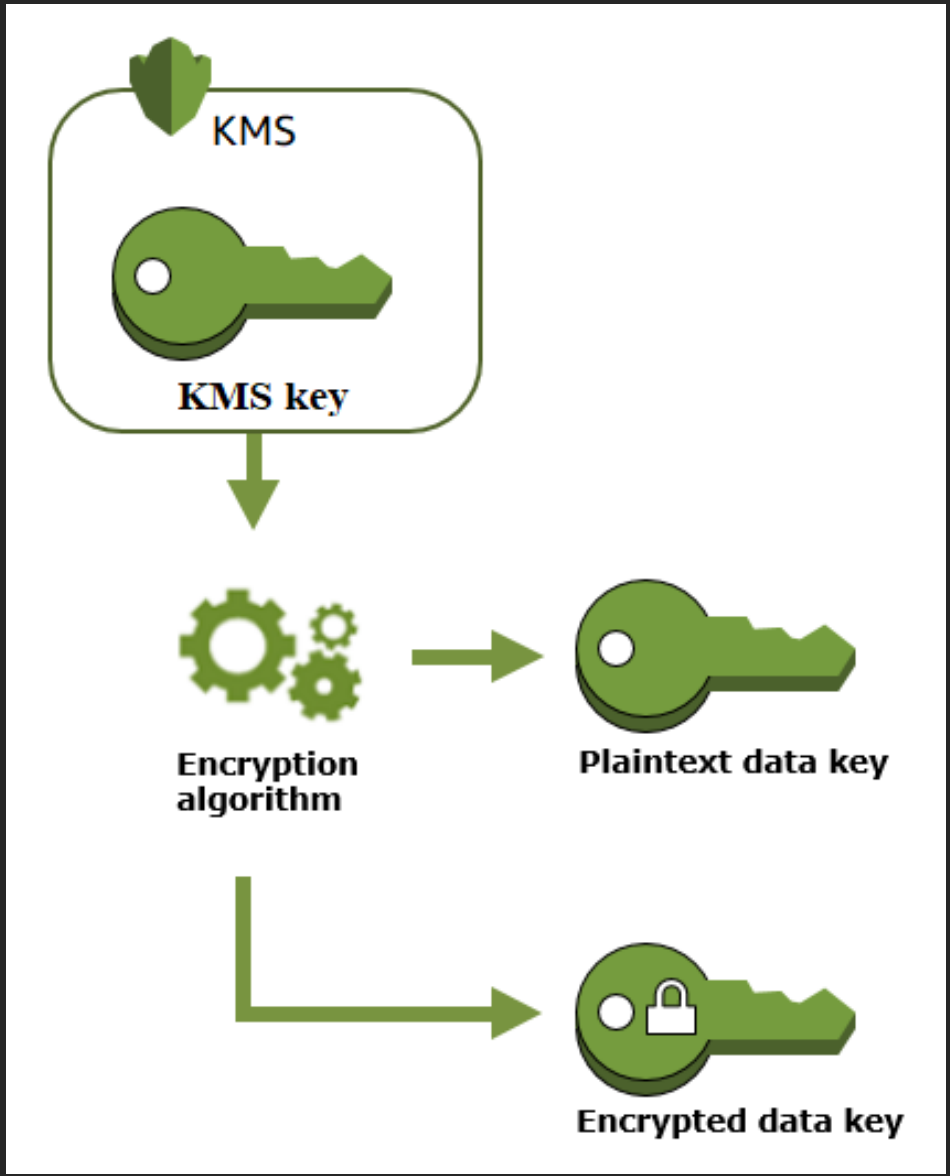
- Create and manage keys using KMS Console

—

- ***KeyId*** – names for KMS keys that are unique to a region and account
- ***KeyARN*** – completely unique identifier (Amazon Resource Name), includes AWS account, region, key ID
- ***Key Material*** – string of bits used in cryptographic algorithm
- ***Encryption context*** – used in symmetric keys, optional key-value pair giving additional contextual information, same context required when encrypting/decrypting
- ***Key policy*** – who can use/manage KMS keys

Data keys

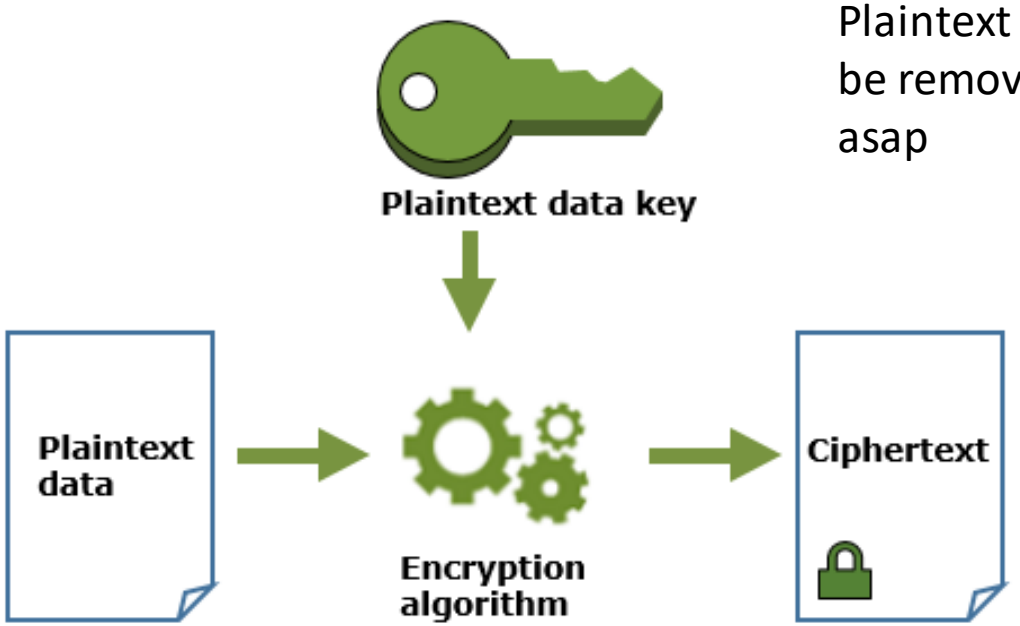
- CMKs can be used to generate data keys
- Data keys can only be used outside AWS KMS
- When generating a data key, encrypts a copy of the data key as well that can be stored
- Can only be decrypted by AWS KMS



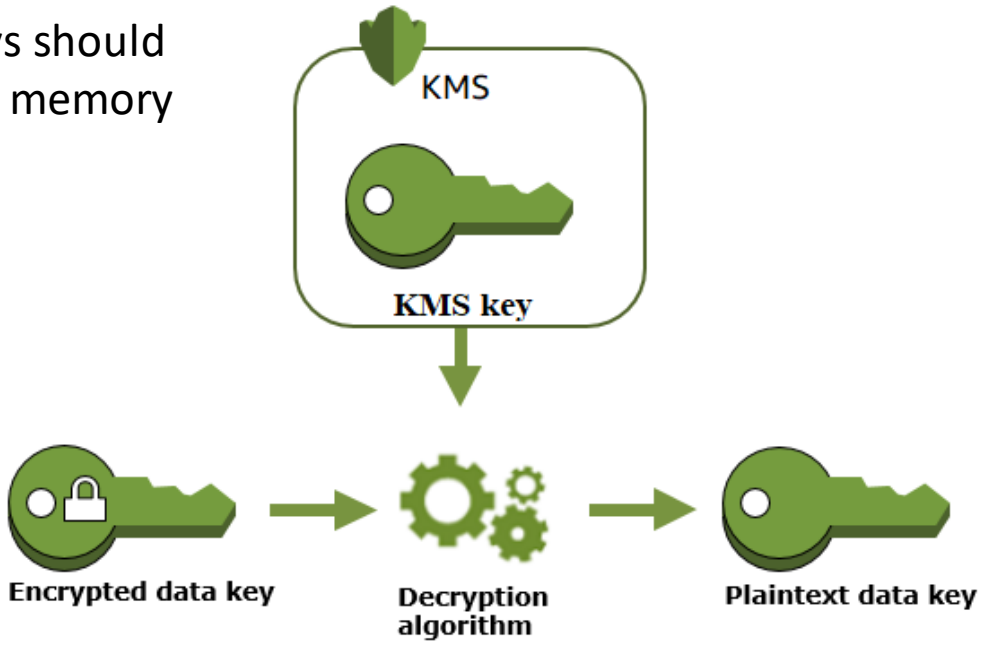
Creating Data keys

- Plaintext data key can be deleted from memory after encryption and encrypted key can be stored along with the data

Encrypting and Decrypting with Data keys



Plaintext data keys should be removed from memory asap



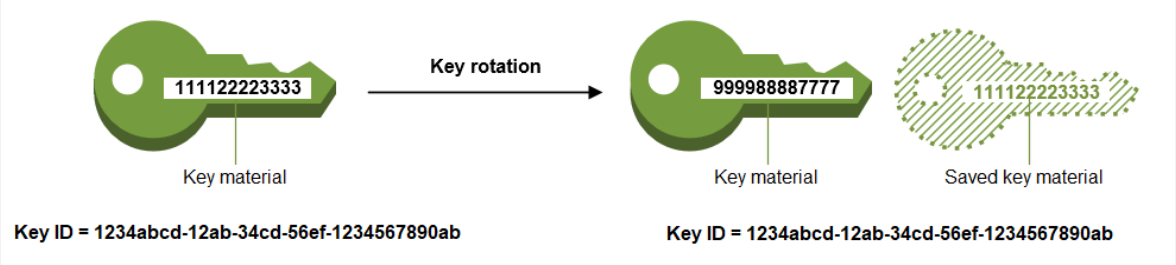
AWS KMS cannot use data key to encrypt data – only can be used outside (AWS Encryption SDK)

Encrypted data key decrypted by AWS KMS using CMK

Using Data key

- Encryption outside KMS
  1. get data key from CMK
  2. use plaintext data key to encrypt data outside KMS. Erase plaintext datakey from memory
  3. store encrypted data key with encrypted data
- Decrypt outside KMS
  1. Decrypt encrypted data key through AWS KMS, getting plaintext data key
  2. Use plaintext data key to decrypt data outside KMS, erase plaintext data key from memory

Rotation



- Rather than creating new keys, can rotate existing CMKs
- KMS keeps track of all keys in perpetuity, does not delete until you delete
- When enabled, AWS KMS rotates the key automatically once a year
- Can be enabled on console
- Rotate manually – new KMS key has different cryptographic material than existing key (replacing keys)
- When decrypting data with rotated key, decrypts with key version used to encrypt it
- No code changes are required

Implementation

- Springboot with Maven
- JDK 18
- AWS Encryption SDK
- AWS KMS Java package

AWS Encryption SDK

- client-side encryption library
- Uses AWS KMS keys and data keys to encrypt/decrypt data
- Only supports symmetric encryption KMS keys

Demo

1. encrypting and decrypting with CMKs using local data
2. generating data keys using CMKs (encrypted and decrypted)

Problems Faced

- Guides for api calls were hard to follow, not many examples provided
- Dependency management on Maven and IntelliJ
- Encrypting using Cipher with generated Data keys had many bugs -
    - Algorithm did not match
    - Provider did not recognize algorithm (AES)

What I Learnt

Software

- Java
- Springboot and Spring 5
- RESTful API calls

Concepts

- AWS KMS
- AWS console (s3)
- Encryption/decryption methods

Further Work

- Data key pairs – asymmetric key pairs
- Encrypt and Decrypt data outside KMS using generated data keys from CMK
- Can be set up with S3 for server side encryption