AWS Key Management Service – KMS

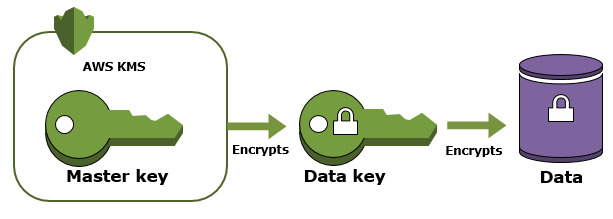
# AWS Key Management Service – KMS

* AWS KMS is a managed encryption service that allows creation and control of encryption keys to enable encryption of data easily
* KMS provides a highly available key storage, management, and auditing solution to encrypt the data across AWS services & within applications
* Customer master keys (CMK) are protected by hardware security modules (HSMs) that are validated by the FIPS 140-2 Cryptographic Module Validation Program
* KMS is seamlessly integrated with several AWS services to make encrypting data in those service easy
* KMS is also integrated with AWS CloudTrail to provide encryption key usage logs to help meet auditing, regulatory and compliance needs.
* KMS Keys are only stored and used in the region in which they are created. They cannot be transferred to another region
* KMS enforces usage and management policies, to control which IAM user, role from the account or other accounts who can manage and use keys
* KMS supports
  + Encrypt, decrypt, and re-encrypt data
  + Generate data encryption keys that can be exported from the service in plaintext or encrypted under a master key that doesn’t leave the service
  + Generate random numbers suitable for cryptographic applications

## Envelope encryption

* AWS cloud services integrated with AWS KMS use a method called envelope encryption to protect the data.
* Envelope encryption is an optimized method for encrypting data that uses two different keys (Master key and Data key)
* With Envelop encryption
  + A data key is generated and used by the AWS service to encrypt each piece of data or resource.
  + Data key is encrypted under a master key defined in AWS KMS.
  + Encrypted data key is then stored by the AWS service.
  + For data decryption by the AWS service, the encrypted data key is passed to AWS KMS and decrypted under the master key that was originally encrypted under so the service can then decrypt the data.
* **KMS does support sending data less than 4 KB to be encrypted, envelope encryption can offer significant performance benefits**
* When the data is encrypted directly with KMS it must be transferred over the network.
* **Envelope encryption reduces the network load for the application or AWS cloud service as Only the request and fulfillment of the data key through KMS must go over the network**

## KMS Service Concepts



* **Customer Master Keys (CMKs)**
  + **AWS KMS customer master keys (CMKs) are 256-bit Advanced Encryption Standard (AES) symmetric keys that are not exportable.**
  + CMKs are created in AWS KMS and never leave AWS KMS unencrypted.
  + **CMK to encrypt and decrypt up to 4 KB (4096 bytes) of data**
  + CMKs to generate, encrypt, and decrypt the data keys that are used outside of AWS KMS to encrypt the data [**Envelope Encryption**]
* **Key Material**
  + CMK is a logical representation of a master key in AWS KMS.
  + CMK contains the key material used to encrypt and decrypt data.
  + CMK also includes master key’s identifiers and other metadata including its creation date, description, and [key state](https://docs.aws.amazon.com/kms/latest/developerguide/key-state.html),
  + By default AWS KMS generates the key material for a newly created CMK. But a CMK can be created without key material and then import your own key material into that CMK.
* **Data Keys**
  + Data keys are encryption keys that you can use to encrypt data, including large amounts of data and other data encryption keys.
  + AWS KMS does not store, manage, or track your data keys.
  + Data keys must be used by services outside of AWS KMS.
* **Encryption & Decryption Process**
  + Use KMS to get encrypted and plaintext data key using CMK
  + Use the plaintext data key to encrypt the data and store the encrypted data key with the data.
  + Use KMS decrypt to get the plaintext data key and decrypt the data
  + Remove the plaintext data key from memory, once operation completed.
* **Key Policies**
  + help determine who can use and manage that CMK
  + add, remove, or change permissions at any time for a customer managed CMK,
  + **cannot edit the key policy for an AWS managed CMK**
* **Grants**
  + provides permissions, an alternative to the key policy.
  + helps to give long-term access that allows AWS principals to use the customer managed CMKs
  + permissions specified in the grant might not take effect immediately due to eventual consistency
* **Grant Tokens**
  + help mitigate the potential delay with grant
  + use the grant token received in the response to CreateGrant API request to make the permissions in the grant take effect immediately

## KMS Working

* KMS centrally manages and securely stores the keys
* Keys can be generated or imported from the key management infrastructure
* Keys can be used from within the applications and supported AWS services to protect the data, but the key never leaves KMS AWS.
* Data is submitted to AWS KMS to be encrypted, or decrypted, under keys that you control.
* Usage policies on these keys can be set that determine which users can use them to encrypt and decrypt data.

## KMS Access Control

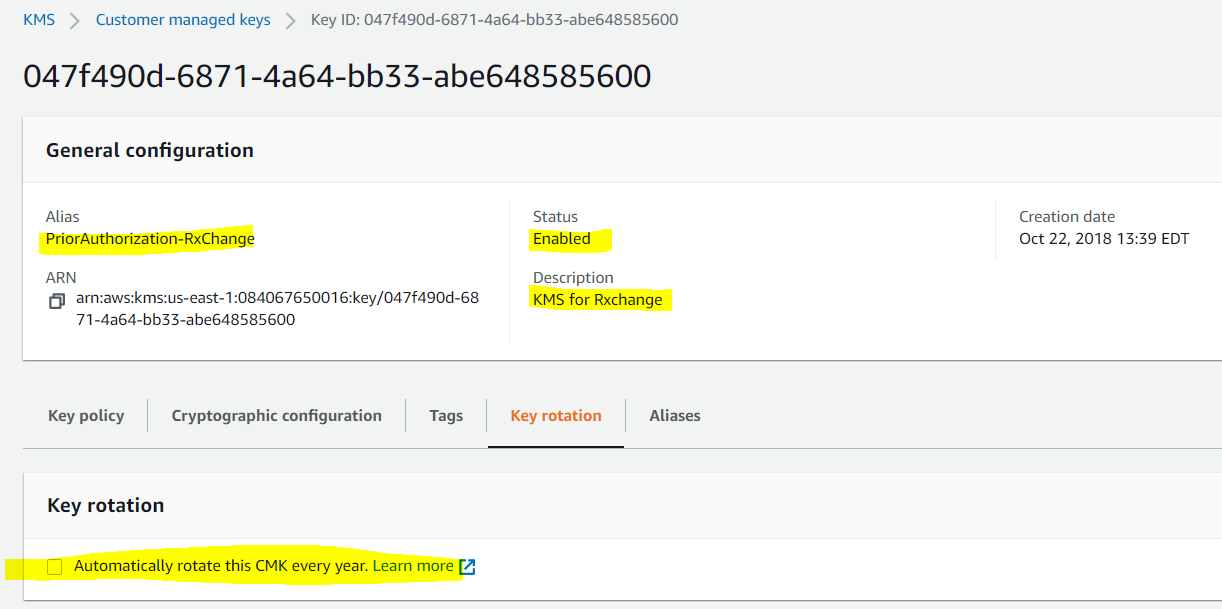
* Primary way to manage access to AWS KMS CMKs is with policies.
* AWS KMS required you attach resource-based policies to the customer master keys (CMKs), called ***key policies***
* All KMS CMKs have a key policy
* KMS CMKs access can be controlled using
  + Use the key policy – use the key policy to control access to a CMK.
  + Use IAM policies with the key policy – use IAM policies in combination with the key policy to control access to a CMK. Controlling access this way enables you to manage all of the permissions for your IAM identities in IAM.
  + Use grants in combination with the key policy – use grants in combination with the key policy to allow access to a CMK. Controlling access this way enables you to allow access to the CMK in the key policy, and to allow users to delegate their access to others.
* To allow access to a KMS CMK, key policy**MUST** be used, either alone or in combination with IAM policies or grants. IAM policies by themselves are not sufficient to allow access to a CMK, though they can be used in combination with a CMK’s key policy.

## Rotating Customer Master Keys

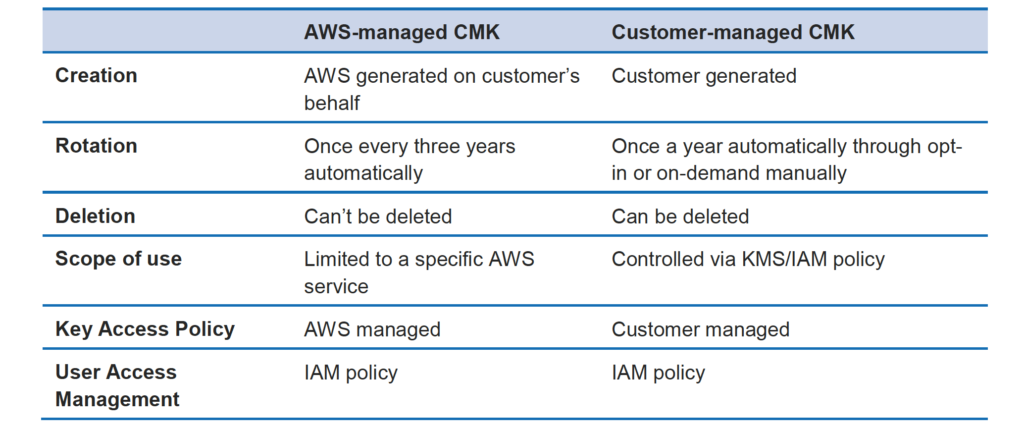
* **AWS KMS managed CMK**
  + **automatically rotated every 3 years**



* **CMKs with generated key material**
  + **supports automatic key rotation**
  + **keys are rotated every year**



* CMKs with imported key material or keys generated in an CloudHSM cluster using the KMS custom key store feature
  + do not support automatic key rotation
  + provides flexibility to manual rotate keys as required
* **for keys rotated automatically by KMS, data does not need to be re-encrypted. KMS keeps previous versions of keys to use for decryption of data encrypted under an old version of a key. All new encryption requests against a key in AWS KMS are encrypted under the newest version of the key.**
* **For manually rotated keys, data has to be re-encrypted depending on the application’s configuration**



## KMS Features

* Create keys with a unique alias and description
* Import your own keys
* Control which IAM users and roles can manage keys
* Control which IAM users and roles can use keys to encrypt & decrypt data
* Choose to have AWS KMS automatically rotate keys on an annual basis
* Temporarily disable keys so they cannot be used by anyone
* Re-enable disabled keys
* Delete keys that you no longer use
* Audit use of keys by inspecting logs in AWS CloudTrail

## AWS Certification Exam Practice Questions

1. You are designing a personal document-archiving solution for your global enterprise with thousands of employee. Each employee has potentially gigabytes of data to be backed up in this archiving solution. The solution will be exposed to he employees as an application, where they can just drag and drop their files to the archiving system. Employees can retrieve their archives through a web interface. The corporate network has high bandwidth AWS DirectConnect connectivity to AWS. You have regulatory requirements that all data needs to be encrypted before being uploaded to the cloud. How do you implement this in a highly available and cost efficient way?
   1. Manage encryption keys on-premise in an encrypted relational database. Set up an on-premises server with sufficient storage to temporarily store files and then upload them to Amazon S3, providing a client-side master key. (Storing temporary increases cost and not a high availability option)
   2. Manage encryption keys in a Hardware Security Module (HSM) appliance on-premise server with sufficient storage to temporarily store, encrypt, and upload files directly into amazon Glacier. (Not cost effective)
   3. **Manage encryption keys in amazon Key Management Service (KMS), upload to amazon simple storage service (s3) with client-side encryption using a KMS customer master key ID and configure Amazon S3 lifecycle policies to store each object using the amazon glacier storage tier.** (With CSE-KMS the encryption happens at client side before the object is upload to S3 and KMS is cost effective as well)
   4. Manage encryption keys in an AWS CloudHSM appliance. Encrypt files prior to uploading on the employee desktop and then upload directly into amazon glacier (Not cost effective)
2. An AWS customer is deploying an application that is composed of an Auto Scaling group of EC2 Instances. The customers security policy requires that every outbound connection from these instances to any other service within the customers Virtual Private Cloud must be authenticated using a unique x 509 certificate that contains the specific instance-id. In addition an x 509 certificates must be designed by the customer’s Key management service in order to be trusted for authentication.  
   Which of the following configurations will support these requirements?
   1. Configure an IAM Role that grants access to an Amazon S3 object containing a signed certificate and configure the Auto Scaling group to launch instances with this role. Have the instances bootstrap get the certificate from Amazon S3 upon first boot.
   2. Embed a certificate into the Amazon Machine Image that is used by the Auto Scaling group Have the launched instances generate a certificate signature request with the instance’s assigned instance-id to the Key management service for signature.
   3. **Configure the Auto Scaling group to send an SNS notification of the launch of a new instance to the trusted key management service. Have the Key management service generate a signed certificate and send it directly to the newly launched instance.**
   4. Configure the launched instances to generate a new certificate upon first boot. Have the Key management service poll the AutoScaling group for associated instances and send new instances a certificate signature that contains the specific instance-id.
3. A company has a customer master key (CMK) with imported key materials. Company policy requires that all encryption keys must be rotated every year. What can be done to implement the above policy?
   1. Enable automatic key rotation annually for the CMK.
   2. Use AWS Command Line interface to create an AWS Lambda function to rotate the existing CMK annually.
   3. Import new key material to the existing CMK and manually rotate the CMK.
   4. **Create a new CMK, import new key material to it, and point the key alias to the new CMK.**
4. An organization policy states that all encryption keys must be automatically rotated every 12 months. Which AWS Key Management Service (KMS) key type should be used to meet this requirement?
   1. AWS managed Customer Master Key (CMK)
   2. **Customer managed CMK with AWS generated key material**
   3. Customer managed CMK with imported key material
   4. AWS managed data key