AWS KMS

AWS KMS

- AWS Key Management Service (KMS) makes it easy for you to create and manage cryptographic keys and control their use across a wide range of AWS services and in your applications.
- AWS KMS is a secure and resilient service that uses hardware security modules that have been validated under FIPS 140-2, or are in the process of being validated, to protect your keys.
- AWS KMS is integrated with AWS CloudTrail to provide you with logs of all key usage to help meet your regulatory and compliance needs.

AWS KMS Symmetric - Asymmetric Keys

- Symmetric encryption is a type of encryption where only one key (a secret key) is used to both encrypt and decrypt electronic information.
- Asymmetric cryptography, also known as public-key cryptography, is a process that uses a pair of related keys -- one public key and one private key -- to encrypt and decrypt a message and protect it from unauthorized access or use.

AWS KMS Features

- AWS KMS provides you with centralized control over the lifecycle and permissions of your keys. You can create new keys whenever you wish, and you can control who can manage keys versus who can use them.
- As an alternative to using keys generated by AWS KMS, you can import keys from your own key management infrastructure, or use keys stored in your AWS CloudHSM cluster.
- You can choose automatic rotation of master keys generated in AWS KMS once per year without the need to re-encrypt previously encrypted data.
- The service automatically keeps older versions of the master key available to decrypt previously encrypted data.

AWS KMS Features

Alexa for Business*	Amazon Elasticsearch	Amazon Personalize	AWS CodeArtifact
Amazon AppFlow	Amazon EMR	Amazon Redshift	AWS CodeBuild
Amazon Athena	Amazon Forecast	Amazon Relational Database Service (RDS)	AWS CodeCommit*
Amazon Aurora	Amazon FSx for Windows File Server	Amazon S3	AWS CodeDeploy
Amazon CloudWatch Logs	Amazon Glacier	Amazon SageMaker	AWS CodePipeline
Amazon Comprehend	Amazon GuardDuty	Amazon Simple Email Service (SES)	AWS Database Migration Service
Amazon Connect	Amazon Kendra	Amazon Simple Notification Service (SNS)	AWS Glue
Amazon DocumentDB	Amazon Kinesis Data Streams	Amazon Simple Queue Service (SQS)	AWS Lambda
Amazon DynamoDB Accelerator (DAX)*	Amazon Kinesis Firehose	Amazon Transcribe	AWS Secrets Manager
Amazon DynamoDB	Amazon Kinesis Video Streams	Amazon Translate	AWS Snowball
Amazon EBS	Amazon Lex	Amazon WorkMail	AWS Snowball Edge
Amazon EC2 Image Builder	Amazon Lightsail*	Amazon WorkSpaces	AWS Snowcone
Amazon EFS	Amazon Macie	AWS Backup	AWS Snowmobile
Amazon Elastic Kubernetes Service (EKS)	Amazon Managed Streaming for Kafka (MSK)	AWS Certificate Manager*	AWS Storage Gateway
Amazon Elastic Transcoder	Amazon MQ	AWS Cloud9*	AWS Systems Manager
Amazon ElastiCache	Amazon Neptune	AWS CloudTrail	AWS X-Ray

AWS KMS is seamlessly integrated with most AWS services. These integrations use envelope encryption, where a data encryption key used by the AWS service to encrypt your data is protected under a customer master key (CMK) stored in AWS KMS. There are two types of CMKs: (i) an AWS managed CMK that is created automatically when you first create an encrypted resource in an AWS service. You can track the usage of an AWS managed CMK, but the lifecycle and permissions of the key are managed on your behalf. (ii) a customer managed CMK that only you can create.

AWS KMS Features

- If you have AWS CloudTrail enabled for your AWS account, each request you make to AWS KMS is recorded in a log file that is delivered to the Amazon S3 bucket that you specified when you enabled AWS CloudTrail. The information recorded includes details of the user, time, date, API action and, when relevant, the key used.
- The CMKs you create or ones that are created on your behalf by other AWS services cannot be exported from the service. Therefore AWS KMS takes responsibility for their durability. To help ensure that your keys and your data is highly available, it stores multiple copies of encrypted versions of your keys in systems that are designed for 99.999999999% durability.

AWS KMS Features

- AWS KMS provides the option for you to create your own key store using HSMs that you control. Each custom key store is backed by an AWS CloudHSM cluster. When you create a CMK in a custom key store, the service generates and stores key material for the CMK in an AWS CloudHSM cluster that you own and manage. When you use a CMK in a custom key store, the cryptographic operations under that key are performed in your AWS CloudHSM cluster.

AWS KMS Compliance

- AWS Service Organization Controls (SOC 1, SOC 2, and SOC 3) Reports. You can download a copy of these reports from AWS Artifact.
- PCI DSS Level 1. For more details on PCI DSS compliant services in AWS, you can read the PCI DSS FAQs.
- FIPS 140-2. The AWS KMS cryptographic module is validated, or in the process of being validated, at FIPS 140-2 Level 2 overall with Level 3 for several other categories, including physical security. For more details, you can view the FIPS 140-2 certificate for AWS KMS HSM along with the associated Security Policy.
- FedRAMP. You can get more details on AWS FedRAMP compliance at FedRAMP Compliance.
- HIPAA. For more details, you can visit the HIPAA Compliance page.

AWS KMS – Key Policy

```
{
  "Sid": "Allow access for Key Administrators",
  "Effect": "Allow",
  "Principal": {"AWS": [
    "arn:aws:iam::111122223333:user/KMSAdminUser",
    "arn:aws:iam::111122223333:role/KMSAdminRole"
  ]},
  "Action": [
    "kms:Create*",
    "kms:Describe*",
    "kms:Enable*",
    "kms:List*",
    "kms:Put*",
    "kms:Update*",
    "kms:Revoke*",
    "kms:Disable*",
    "kms:Get*",
    "kms:Delete*",
    "kms:TagResource",
    "kms:UntagResource",
    "kms:ScheduleKeyDeletion",
    "kms:CancelKeyDeletion"
  ],
  "Resource": "*"
}
```

The key policy effectively gives access to the user and the role mentioned under the principal block full access to administer the key.

Always good to remember is that if there is an explicit deny mentioned in the key policy and an allow is mentioned in the IAM policy, the user or role will still be denied as an explicit deny always trumps an allow.

AWS KMS – Key Policy

```
{
  "Sid": "Allow use of the key",
  "Effect": "Allow",
  "Principal": {"AWS": [
    "arn:aws:iam::111122223333:user/CMKUser",
    "arn:aws:iam::111122223333:role/CMKRole",
    "arn:aws:iam::444455556666:root"
  ]},
  "Action": [
    "kms:Encrypt",
    "kms:Decrypt",
    "kms:ReEncrypt*",
    "kms:GenerateDataKey*",
    "kms:DescribeKey"
  ],
  "Resource": "*"
},
{
  "Sid": "Allow attachment of persistent resources",
  "Effect": "Allow",
  "Principal": {"AWS": [
    "arn:aws:iam::111122223333:user/CMKUser",
    "arn:aws:iam::111122223333:role/CMKRole",
    "arn:aws:iam::444455556666:root"
  ]},
  "Action": [
    "kms:CreateGrant",
    "kms:ListGrants",
    "kms:RevokeGrant"
  ],
  "Resource": "*",
  "Condition": {"Bool": {"kms:GrantIsForAWSResource": true}}
}
```

The key policy here allows the user, role as well as the root user of the AWS account permission to use the key as this policy is for Key Users.

Grants are temporary permissions. These are commands executed from the AWS CLI to provide the user, role or the root user temporary permissions to encrypt or decrypt the objects and so on. You can only use grants to Allow and not Deny.

The minimal permissions that are required are Encrypt, Decrypt, ReEncrypt and GenerateDataKey.

AWS KMS – Policy Conditions

AWS KMS supports Amazon Virtual Private Cloud (Amazon VPC) endpoints that are powered by AWS PrivateLink. You can use the following global condition keys in IAM policies to allow or deny access to a particular VPC or VPC endpoint. You can also use these global condition keys in AWS KMS key policies to restrict access to AWS KMS CMKs to requests from the VPC or VPC endpoint.

- aws:SourceVpc limits access to requests from the specified VPC.
- aws:SourceVpce limits access to requests from the specified VPC endpoint.

AWS KMS – Policy Conditions

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:user/ExampleUser"
  },
  "Action": "kms:CreateKey",
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:KeyOrigin": "AWS_KMS"
    }
  }
}
```

The kms:KeyOrigin condition key controls access to operations based on the value of the Origin property of the CMK that is created by or used in the operation.
Valid values for Origin are AWS_KMS, AWS_CLOUDHSM, and EXTERNAL.

AWS KMS – Policy Conditions

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:user/ExampleUser"
  },
  "Action": [
    "kms:Encrypt",
    "kms:Decrypt",
    "kms:ReEncrypt*",
    "kms:GenerateDataKey*",
    "kms:CreateGrant",
    "kms:ListGrants",
    "kms:DescribeKey"
  ],
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "kms:ViaService": [
        "ec2.us-west-2.amazonaws.com",
        "rds.us-west-2.amazonaws.com"
      ]
    }
  }
}
```

The kms:ViaService condition key limits use of an AWS KMS customer master key (CMK) to requests from specified AWS services. You can specify one or more services in each kms:ViaService condition key.

You can also use a kms:ViaService condition key to deny permission to use a CMK when the request comes from particular services

AWS KMS vs CloudHSM

CloudHSM

- Protect and store your cryptographic keys with industry standard, tamper-resistant HSM appliances. No one but you has access to your keys (including Amazon administrators who manage and maintain the appliance).
- Use your most sensitive and regulated data on Amazon EC2 without giving applications direct access to your data's encryption keys.
- Store and access data reliably from your applications that demand highly available and durable key storage and cryptographic operations.

AWS KMS vs CloudHSM

AWS KMS

- Centralized Key Management
- Integrated with AWS services
- Encryption for all your applications

Note : Both AWS KMS and CloudHSM are FIPS140-2 compliant and both support symmetric and asymmetric key encryption. AWS KMS is multi-tenant but CloudHSM is single-tenant.

AWS KMS – Cross Account Access

To give permission to use a CMK to users and roles in another account, you must use two different types of policies:

- The **key policy** for the CMK must give the external account (or users and roles in the external account) permission to use the CMK. The key policy is in the account that owns the CMK.
- You must attach **IAM policies** to IAM users and roles in the external account. These IAM policies delegate the permissions that are specified in the key policy.

AWS KMS – Cross Account Access

```
{
    "Sid": "Allow an external account to use this CMK",
    "Effect": "Allow",
    "Principal": {
        "AWS": [
            "arn:aws:iam::444455556666:role/ExampleRole",
            "arn:aws:iam::444455556666:user/ExampleUser"
        ]
    },
    "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:DescribeKey"
    ],
    "Resource": "*"
}
```

For example, the following example key policy statement allows ExampleRole and ExampleUser in account 444455556666 to use a CMK in account 111122223333. This key policy statement gives the external account, 444455556666, permission to use the CMK in cryptographic operations for symmetric CMKs.

AWS KMS – Cross Account Access

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Allow Use Of CMK In Account 111122223333",
      "Effect": "Allow",
      "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:DescribeKey"
      ],
      "Resource": "arn:aws:kms:us-west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
    }
  ]
}
```

The following example IAM policy allows the principal to use the CMK in account 111122223333 for cryptographic operations. To give this permission to users and roles in account 444455556666, att ach the policy to the users or roles in account 444455556666.