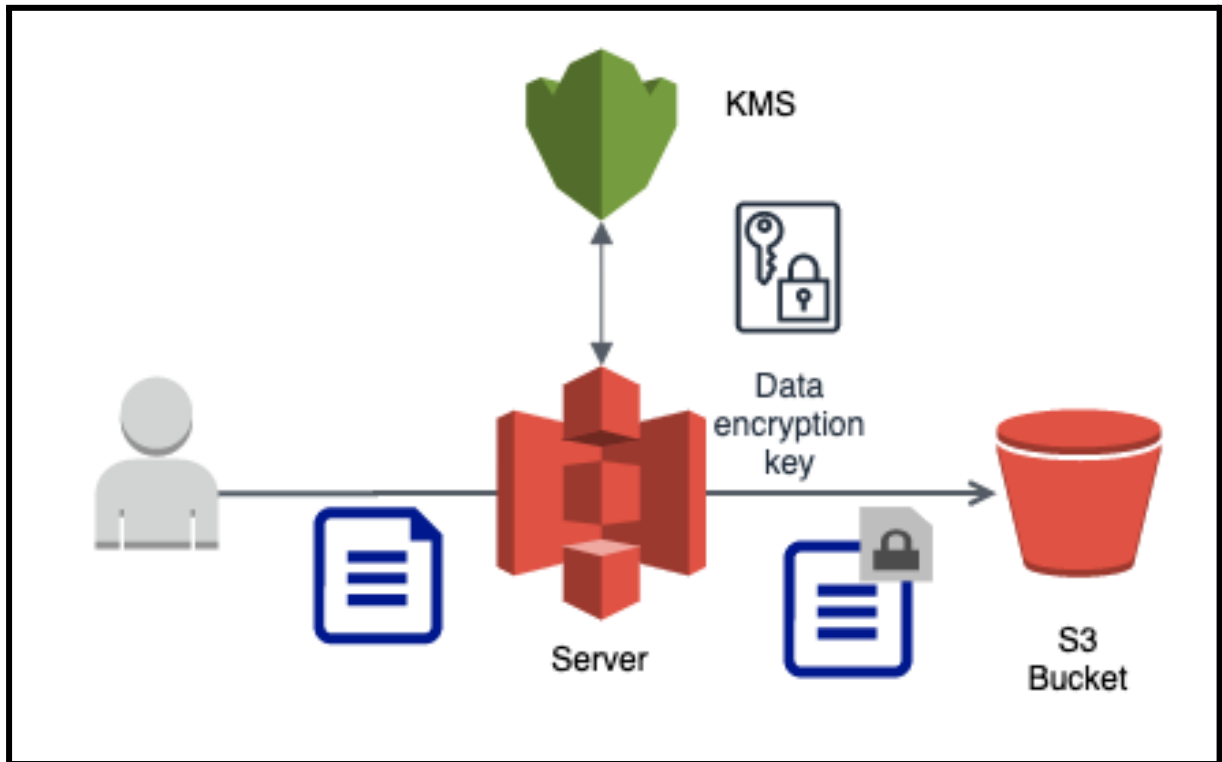


## AIM : Configuring AWS S3 and KMS for Restricted Data Access



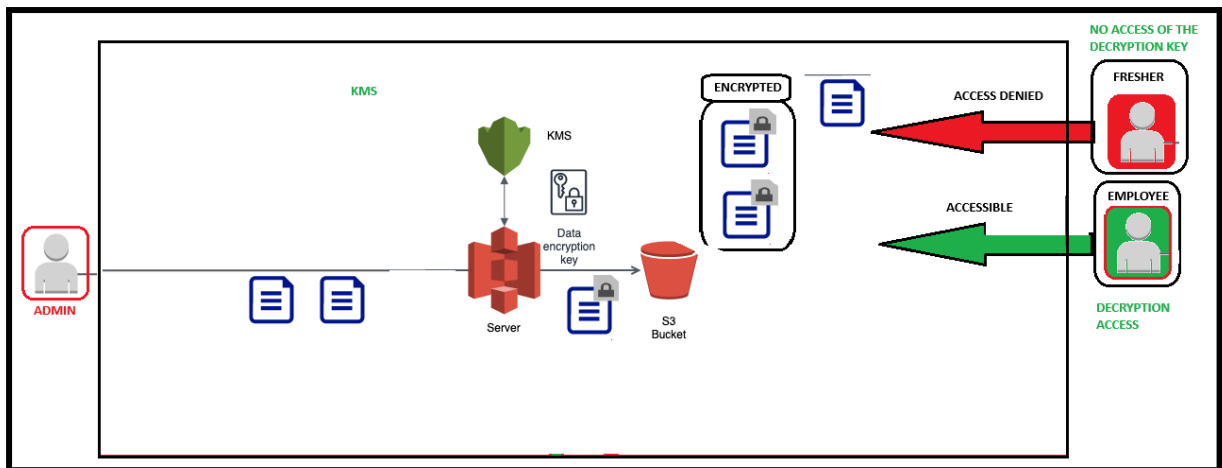
### Introduction

This documentation provides a comprehensive guide to setting up a secure AWS environment using Amazon S3 for storage and AWS Key Management Service (KMS) for encryption. The scenario outlines creating two IAM users with S3 read-only access and configuring selective encryption on S3 objects using a symmetric KMS key.

### Objective

The primary goal is to securely store data in an S3 bucket where:

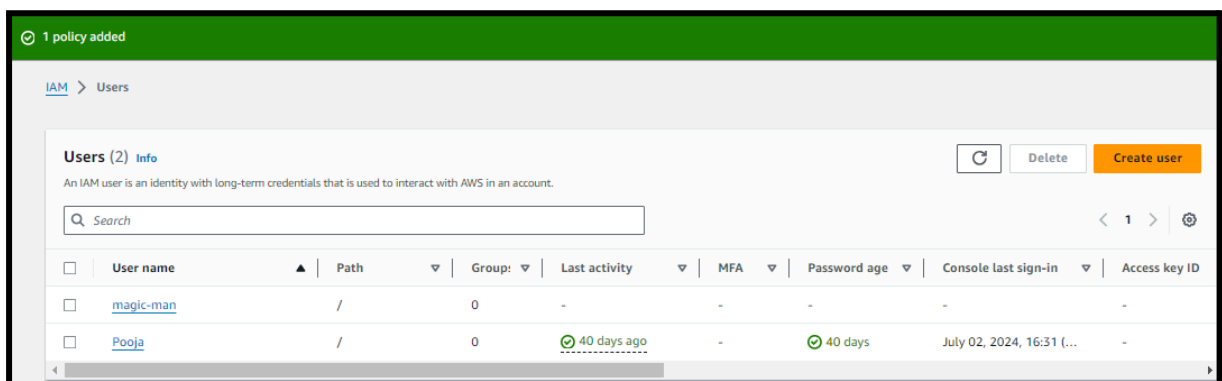
- Two IAM users (USER1 and USER2) have read-only access to all objects in the bucket.
- Some objects are encrypted using a symmetric KMS key, which both Admin and User1 can decrypt.
- The setup ensures compliance with security policies while enabling effective data access and protection.



## Setup Configuration

### Step 1: Creating IAM Users

1. **Login to AWS Management Console** and navigate to the IAM dashboard.
2. **Create two users (Admin and User1):**
  - Click on “Add user”.



- Enter user names and select both Programmatic and AWS Management Console access types.
- Assign **AmazonS3ReadOnlyAccess** to ensure they have read-only access to S3 buckets.

## Set permissions

Add user to an existing group or create a new one. Using groups is a best-practice way to manage user's permissions by job functions. [Learn more](#)

### Permissions options

☐ Add user to group  
Add user to an existing group, or create a new group. We recommend using groups to manage user permissions by job function.

☐ Copy permissions  
Copy all group memberships, attached managed policies, and inline policies from an existing user.

☒ Attach policies directly  
Attach a managed policy directly to a user. As a best practice, we recommend attaching policies to a group instead. Then, add the user to the appropriate group.

### Permissions policies (1/1230)

Choose one or more policies to attach to your new user.

Filter by Type
All types
1 match

<input checked="" type="checkbox"/>	Policy name	Type	Attached entities
<input checked="" type="checkbox"/>	AmazonS3ReadOnlyAccess	AWS managed	0

► Set permissions boundary - optional

Cancel
Previous
Next

## Step 2: Setting Up AWS KMS

1. **Navigate to the KMS section** in the IAM dashboard.
2. **Create a new symmetric key:**
  - Choose “Create a key” and select “Symmetric”.
  - Set an alias and description for the key.
  - Configure permissions to allow only Admin and User1 to use this key for decryption purposes.

## Configure key

### Key type [Help me choose](#)

☒ Symmetric  
A single key used for encrypting and decrypting data or generating and verifying HMAC codes

☐ Asymmetric  
A public and private key pair used for encrypting and decrypting data, signing and verifying messages, or deriving shared secrets

### Key usage [Help me choose](#)

☒ Encrypt and decrypt  
Use the key only to encrypt and decrypt data.

☐ Generate and verify MAC  
Use the key only to generate and verify hash-based message authentication codes (HMAC).

▼ Advanced options

Key material origin

Key material origin is a KMS key property that represents the source of the key material when creating the KMS key. [Help me choose](#)

☒ **KMS - recommended**  
AWS KMS creates and manages the key material for the KMS key.

☐ **External (Import Key material)**  
You create and import the key material for the KMS key.

☐ **AWS CloudHSM key store**  
AWS KMS creates the key material in the AWS CloudHSM cluster of your AWS CloudHSM key store.

☐ **External key store**  
The key material for the KMS key is in an external key manager outside of AWS.

Regionality

Create your KMS key in a single AWS Region (default) or create a KMS key that you can replicate into multiple AWS Regions. [Help me choose](#)

☒ **Single-Region key**  
Never allow this key to be replicated into other Regions

☐ **Multi-Region key**  
Allow this key to be replicated into other Regions

Keep this field empty for now

**Key users (4)**

Select the IAM users and roles that can use the KMS key in cryptographic operations. [Learn more](#)

Q Search Key users

< 1 >

<input type="checkbox"/>	Name	Path	Type
<input type="checkbox"/>	magic-man	/	User
<input type="checkbox"/>	Pooja	/	User
<input type="checkbox"/>	AWSServiceRoleForSupport	/aws-service-role/support.am...	Role
<input type="checkbox"/>	AWSServiceRoleForTrustedAd...	/aws-service-role/trustedadvis...	Role

Success

Your AWS KMS key was created with alias **view-bucket-obj-only** and key ID **6b8a3eaa-5470-46e0-9d4f-9e782cb3040c**.

View key

KMS > Customer managed keys

**Customer managed keys (1)**

Key actions ▼ Create key

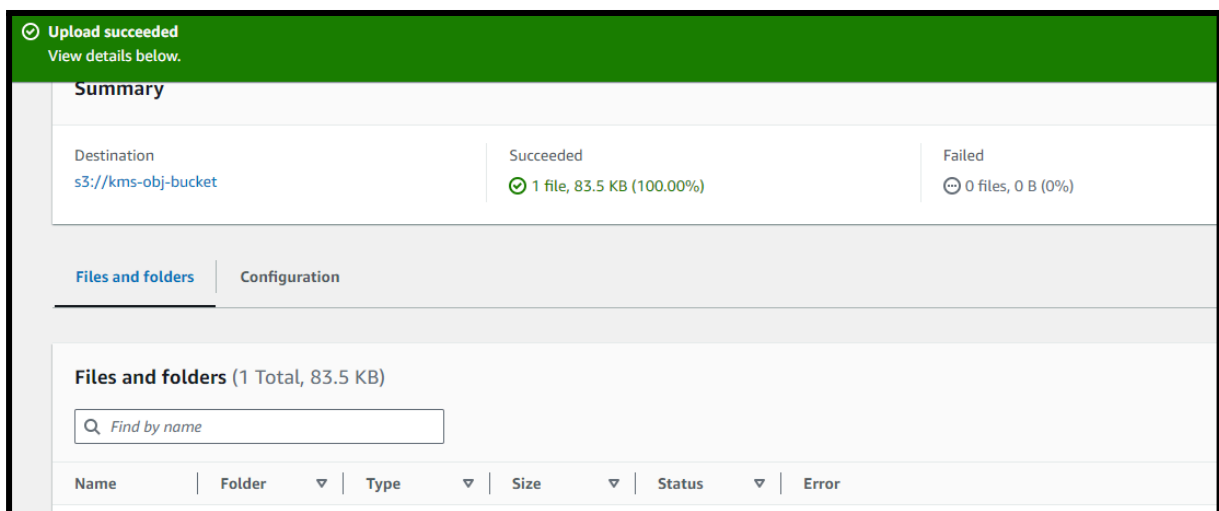
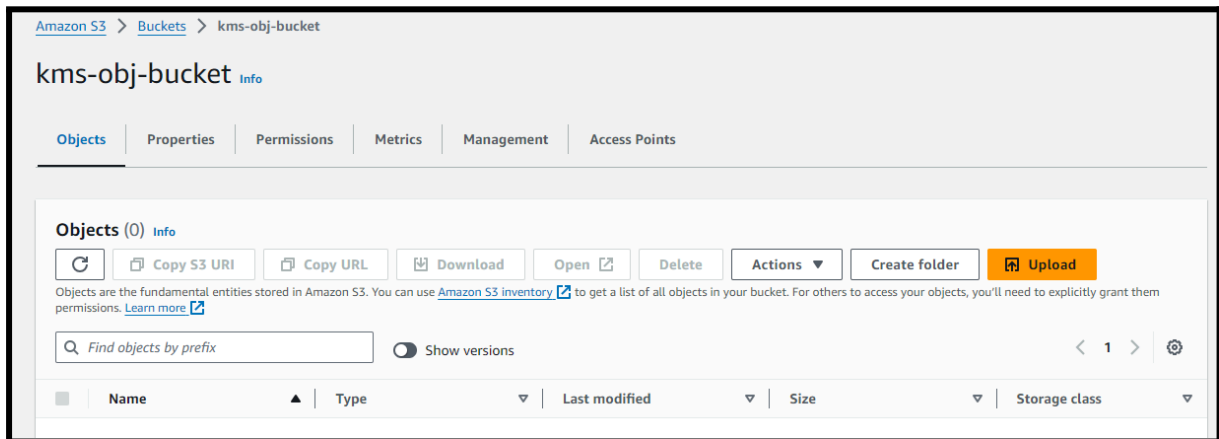
Q Filter keys by properties or tags

< 1 > ⚙

<input type="checkbox"/>	Aliases	Key ID	Status	Key type	Key spec	Key usage
<input type="checkbox"/>	<a href="#">view-bucket-obj-only</a>	<a href="#">6b8a3eaa-547...</a>	Enabled	Symmetric	SYMMETRIC_D...	Encrypt a...

## Step 3: Creating and Configuring an S3 Bucket

1. **Go to the S3 service** in the AWS Management Console.
2. **Create a new bucket:**
  - Click “Create bucket”.
  - Follow the setup to name the bucket and select the appropriate region.
  - Ensure all public access is blocked to enhance security.



## Step 4: Configuring Encryption on S3 Objects

### 1. Manually encrypt specific objects using the KMS key:

- During the upload process, select the encryption option.
- Choose the previously created KMS key for encryption to secure sensitive files.

Amazon S3 > Buckets > kms-obj-bucket > Upload

## Upload [Info](#)

Add the files and folders you want to upload to S3. To upload a file larger than 160GB, use the AWS CLI, AWS SDK or Amazon S3 REST API. [Learn more](#)

Drag and drop files and folders you want to upload here, or choose **Add files** or **Add folder**.

**Files and folders (1 Total, 83.5 KB)** [Remove](#) [Add files](#) [Add folder](#)

All files and folders in this table will be uploaded.

<input type="checkbox"/>	Name	Folder	Type
<input type="checkbox"/>	11zon_resized.jpg	-	image/jpeg

### Server-side encryption

☐ Don't specify an encryption key  
The bucket settings for default encryption are used to encrypt objects when storing them in Amazon S3.

☒ Specify an encryption key  
The specified encryption key is used to encrypt objects before storing them in Amazon S3.

### Encryption settings [Info](#)

☐ Use bucket settings for default encryption

☒ Override bucket settings for default encryption

### Encryption type [Info](#)

☐ Server-side encryption with Amazon S3 managed keys (SSE-S3)

☒ Server-side encryption with AWS Key Management Service keys (SSE-KMS)

☐ Dual-layer server-side encryption with AWS Key Management Service keys (DSSE-KMS)  
Secure your objects with two separate layers of encryption. For details on pricing, see [DSSE-KMS pricing](#) on the **Storage** tab of the [Amazon S3 pricing page](#).

### AWS KMS key [Info](#)

☒ Choose from your AWS KMS keys

☐ Enter AWS KMS key ARN

AWS KMS key

Info

☒ Choose from your AWS KMS keys
 ☐ Enter AWS KMS key ARN

Available AWS KMS keys

arn:aws:kms:us-east-2:637423493890:key/6b8a3... ▼

↺

Create a KMS key ↗

**Bucket Key is enabled for objects uploaded, modified, or copied in this bucket**  
 Uploaded, modified, or copied objects inherit their Bucket Key settings from the bucket default encryption configuration unless they already have Bucket Key configured. [Learn more](#)

Bucket Key

Using an S3 Bucket Key for SSE-KMS reduces encryption costs by lowering calls to AWS KMS. S3 Bucket Keys aren't supported for DSSE-KMS. [Learn more](#)

☐ Disable
 ☒ Enable

## Step 5: Uploading Data

- Upload both encrypted and unencrypted objects:
  - Admin, as the primary user, should upload files to the S3 bucket.
  - Selectively apply encryption to sensitive files during the upload.

Upload succeeded

View details below.

Summary

Destination

s3://kms-obj-bucket

Succeeded

1 file, 83.5 KB (100.00%)

Failed

0 files, 0 B (0%)

Files and folders

Configuration

Files and folders (1 Total, 83.5 KB)

Find by name

< 1 >

Name	Folder	Type	Size	Status	Error
<a href="#">11zon_resiz...</a>	-	image/jpeg	83.5 KB	Succeeded	-

## User Access and Permissions

- Configure IAM roles and policies** to ensure that both user1 and User2 can read all objects.
- Set decryption permissions** using the KMS key for both user1 to access encrypted files securely.

Key deletion

☐ Allow key administrators to delete this key

Key users (1)

Add

Remove

The following IAM users and roles can use this key for cryptographic operations. They can also allow AWS services that are integrated with KMS to use the key on their behalf. [Learn more](#)

Q Search Key users

< 1 >

<input type="checkbox"/>	Name	Path	Type
<input type="checkbox"/>	Pooja	/	User

Other AWS accounts

Add other AWS accounts

aws

Services

0

Pooja @ magic1

≡

Info

🔍

Amazon S3 > ... > 11zon\_resized.jpg

11zon\_resized.jpg

Info

📄 Copy S3 URI

📄 Download

🔗 Open

Object actions

Properties

Permissions

Versions

Object overview

Owner

a26699cd1c24cffae5de1c1a262335e93f4ef57038e2af2bd76969ddb889f907

S3 URI

📄 s3://kms-obj-bucket/11zon\_resize

d.jpg



aws

Services

magic-man

Amazon S3

>

...

>

11zon\_resized.jpg

11zon\_resized.jpg

Info

Copy S3 URI

Download

Open

Object actions

Properties

Permissions

Versions

Object overview

Owner

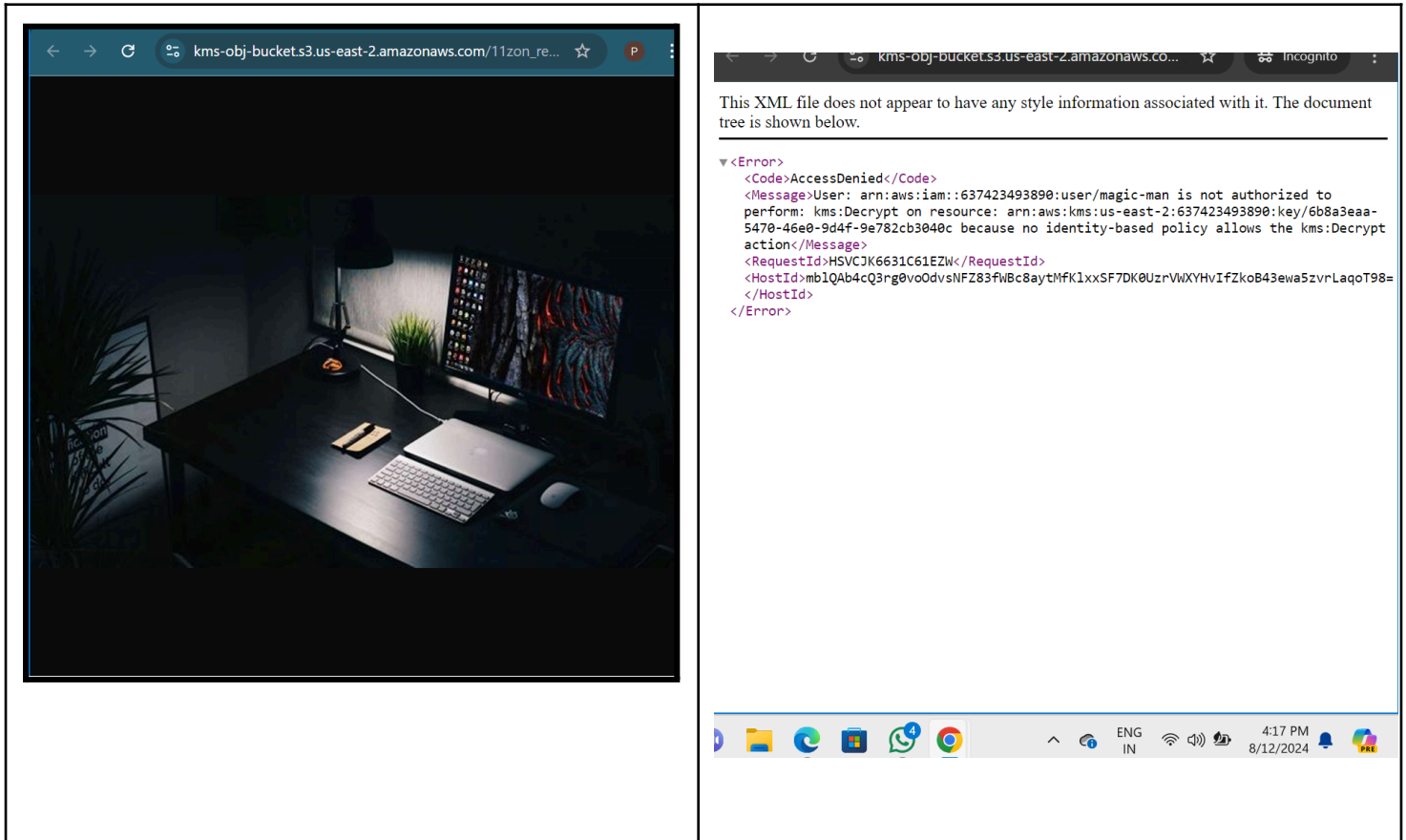
a26699cd1c24cfae5de1c1a262335e93f4ef57038e2af2bd76969ddb889f907

AWS Region

S3 URI

s3://kms-obj-bucket/11zon\_resized.jpg

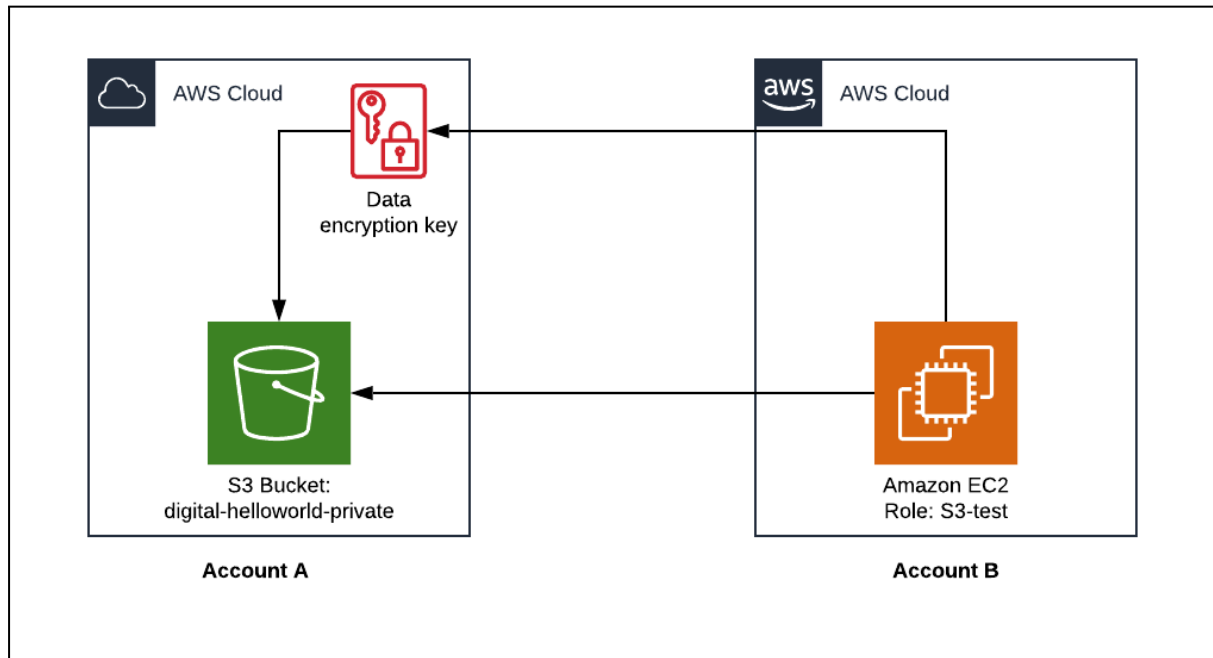
Amazon Resource Name (ARN)



## Conclusion

organisations can effectively manage data security in AWS using S3 and KMS. The read-only access coupled with selective encryption provides a robust security framework while maintaining straightforward access for authorised users.

## AIM : Secure Cross-Account Data Sharing in AWS Using S3, IAM, and KMS



Amazon EC2 instance in one AWS account (Account B) needs access to a privately encrypted S3 bucket in another account (Account A). This is a common use case in environments where different teams or divisions manage resources separately but need to share data securely.

### Objective

To securely share encrypted data stored in an S3 bucket in Account A with an EC2 instance in Account B, using AWS Identity and Access Management (IAM) roles and AWS Key Management Service (KMS) for encryption key access.

### Architecture Overview

- **Account A** contains an S3 bucket (**digital-helloworld-private**) that uses a data encryption key managed by AWS KMS for encrypting data at rest.
- **Account B** has an EC2 instance with an IAM role (**S3-test**) which needs to access the encrypted data in Account A's S3 bucket.

### Steps to Implement

## Step 1: Configure IAM Roles and Policies

- **Account A:**
  1. Create a KMS key for S3 bucket encryption. Attach a key policy that allows the **S3-test** role in Account B to use this key for decryption.
  2. Modify the bucket policy of **digital-helloworld-private** to permit the **S3-test** role from Account B to perform **s3:GetObject** on the bucket.
- **Account B:**
  1. Create an IAM role (**S3-test**) with permissions to access S3. Attach policies that grant access to the S3 bucket in Account A and use the KMS key for decryption.
  2. Attach this IAM role to the EC2 instance.

## Step 2: Enable Cross-Account Access

- Use AWS Resource Access Manager (RAM) if applicable, or ensure that trust relationships are configured correctly in IAM roles and policies to allow the EC2 instance in Account B to access resources in Account A.

## Step 3: Data Encryption and Access

- Encrypt data in the S3 bucket using the specified KMS key in Account A before it is stored.
- Access the data from the EC2 instance in Account B by utilizing the assigned IAM role, which should include permissions to use the KMS key for decryption.

## Step 4: Monitoring and Logging

- Implement AWS CloudTrail and AWS CloudWatch in both accounts to monitor access and operations on the S3 bucket and the EC2 instance, ensuring all accesses are logged and auditable.

## Conclusion

This setup ensures that sensitive data stored in an S3 bucket can be securely shared between different AWS accounts while maintaining stringent security measures through encryption and controlled access. This approach not only

maximises data security but also adheres to principles of least privilege by restricting access to what is necessary.

### **Real-Life Corporate Use Case**

In a corporate environment, such as a multinational corporation with multiple independent departments, the need to share sensitive financial reports or customer data securely between departments is common.

For instance, a finance department (Account A) could store encrypted budget reports in an S3 bucket, which need to be accessed by a data analytics team using an EC2 instance (Account B) for real-time financial analysis and forecasting.

This architecture facilitates secure data sharing while complying with corporate governance and data privacy laws.

# Securing Data with AWS KMS and OpenSSL on EC2

date : 09-8-24

## AIM : KMS ( KEY MANAGEMENT SERVICE )

Create IAM user and role :

### Name, review, and create

#### Role details

Role name

Enter a meaningful name to identify this role.

Maximum 64 characters. Use alphanumeric and '+,=, @, -, \_' characters.

Description

Add a short explanation for this role.

Allows EC2 instances to call AWS services on your behalf.

Maximum 1000 characters. Use letters (A-Z and a-z), numbers (0-9), tabs, new lines, or any of the following characters: \_+=, @-/\[\]!#\$%^\*()';:"'`

#### Permissions policy summary

Policy name	Type	Attached as
<a href="#">AmazonS3FullAccess</a>	AWS managed	Permissions policy

### Step 3: Add tags

Add tags - *optional* [Info](#)

Tags are key-value pairs that you can add to AWS resources to help identify, organize, or search for resources.

No tags associated with the resource.

Add new tag

You can add up to 50 more tags.

Cancel

Previous

Create role

## Configure key

### Key type [Help me choose](#)

☒ Symmetric

A single key used for encrypting and decrypting data or generating and verifying HMAC codes

☐ Asymmetric

A public and private key pair used for encrypting and decrypting data, signing and verifying messages, or deriving shared secrets

### Key usage [Help me choose](#)

☒ Encrypt and decrypt

Use the key only to encrypt and decrypt data.

☐ Generate and verify MAC

Use the key only to generate and verify hash-based message authentication codes (HMAC).

### ▼ Advanced options

#### Key material origin

Key material origin is a KMS key property that represents the source of the key material when creating the KMS key. [Help me choose](#)

☐ KMS - *recommended*

AWS KMS creates and manages the key material for the KMS key.

☒ External (Import Key material)

You create and import the key material for the KMS key.

☐ AWS CloudHSM key store

AWS KMS creates the key material in the AWS CloudHSM cluster of your AWS CloudHSM key store.

☐ External key store

The key material for the KMS key is in an external key manager outside of AWS.

You can import key material from your key management infrastructure into AWS KMS and use it like any other AWS KMS key.

☐ I understand the [security and durability implications](#) of using an imported key.

#### Regionality

Create your KMS key in a single AWS Region (default) or create a KMS key that you can replicate into multiple AWS Regions. [Help me choose](#)

☒ Single-Region key

Never allow this key to be replicated into other Regions

☐ Multi-Region key

Allow this key to be replicated into other Regions

Cancel

Next

## Create key

### Add labels

#### Alias

You can change the alias at any time. [Learn more](#)

Alias

## Define key administrative permissions

### Key administrators (1/4)

Choose the IAM users and roles who can administer this key through the KMS API. You may need to add additional permissions for the users or roles to administer this key from this console. [Learn more](#)

< 1 >

<input type="checkbox"/>	Name	Path	Type
<input type="checkbox"/>	AWSServiceRoleForSupport	/aws-service-role/support.am...	Role
<input type="checkbox"/>	AWSServiceRoleForTrustedAd...	/aws-service-role/trustedadvis...	Role
<input type="checkbox"/>	role-with-kms	/	Role
<input checked="" type="checkbox"/>	Pooja	/	User

### Key deletion

☐ Allow key administrators to delete this key.

Cancel

Previous

Next



## Define key usage permissions

### Key users (1/4)

Select the IAM users and roles that can use the KMS key in cryptographic operations. [Learn more](#)

< 1 >

<input type="checkbox"/>	Name	Path	Type
<input type="checkbox"/>	Pooja	/	User
<input type="checkbox"/>	AWSServiceRoleForSupport	/aws-service-role/support.am...	Role
<input type="checkbox"/>	AWSServiceRoleForTrustedAd...	/aws-service-role/trustedadvis...	Role
<input checked="" type="checkbox"/>	role-with-kms	/	Role

### Other AWS accounts

Specify the AWS accounts that can use this key. Administrators of the accounts you specify are responsible for managing the permissions that allow their IAM users and roles to use this key. [Learn more](#)

[KMS](#) > [Customer managed keys](#) > [Key ID: f11d9d35-8448-4fb4-810f-5ce9db69a5e5](#) > Import key material

Step 1

**Download wrapping public key and import token**

Step 2

Upload your wrapped key material

## Download wrapping public key and import token

To import key material, first select the wrapping key spec and wrapping algorithm you will use to encrypt the key material, then download the wrapping public key and import token for this AWS KMS key. [Learn more](#)

### Configuration

#### Select wrapping key spec

The key spec of the wrapping public key determines the length of the keys in the key pair that protects your key material during its transport to AWS KMS.

☒ RSA\_4096 - recommended

☐ RSA\_3072

☐ RSA\_2048

#### Select wrapping algorithm

Choose the encryption algorithm that you'll use to protect ("wrap") your key material in transit to AWS KMS.

RSAES\_OAEP\_SHA\_256

### Download

Wrapping public key

WrappingPublicKey.bin

Import token

ImportToken.bin

Read me instructions

README.txt

## Download key

Select wrapping key spec

The key spec of the wrapping public key determines the length of the keys in the key pair that protects your key material during its transport to AWS KMS.

☒ RSA\_4096 - recommended

☐ RSA\_3072

☐ RSA\_2048

Select wrapping algorithm

Choose the encryption algorithm that you'll use to protect ("wrap") your key material in transit to AWS KMS.

RSAES\_OAEP\_SHA\_256

▼

Download

Wrapping public key	Import token	Read me instructions
WrappingPublicKey.bin	ImportToken.bin	README.txt

ⓘ

This wrapping public key and import token will expire in 24 hours.

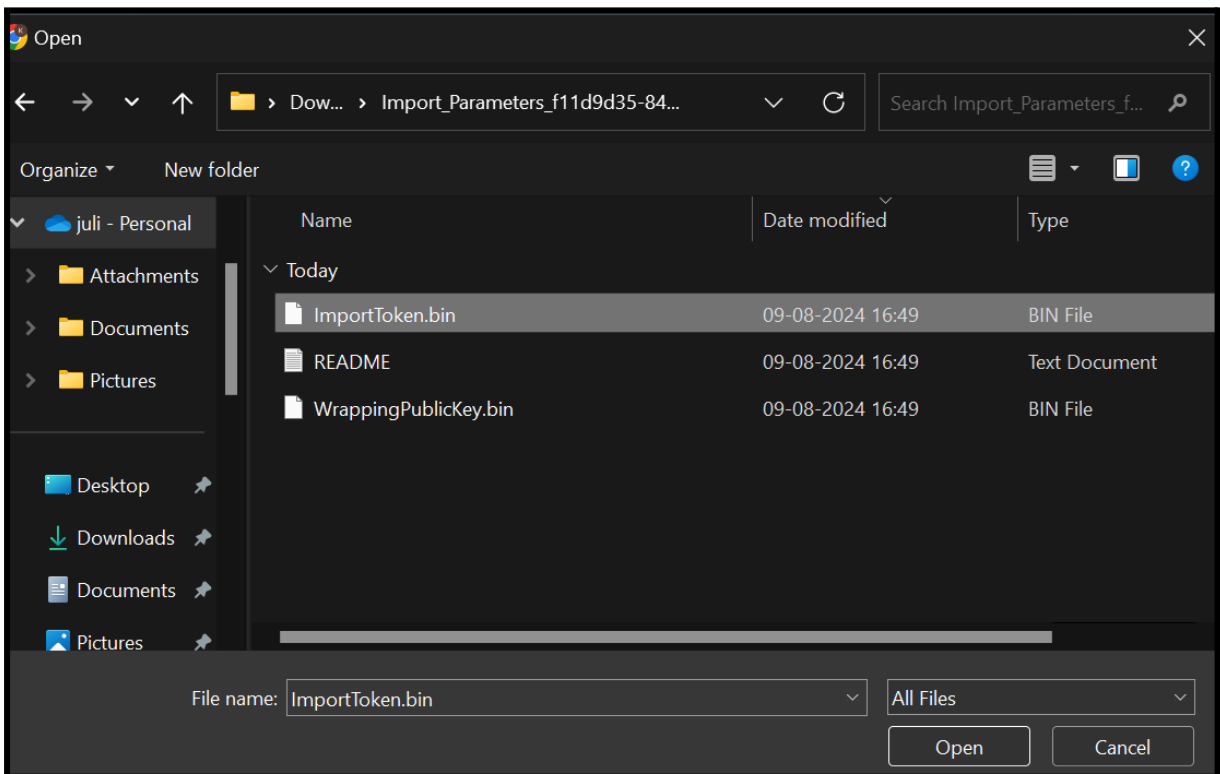
📄

Download wrapping public key and import token

Cancel

Next

## Import token



## Launch an EC2

Instances (1) [Info](#)

Find Instance by attribute or tag (case-sensitive)

All states ▼

<input type="checkbox"/>	Name <a href="#">✎</a> ▼	Instance ID	Instance state ▼	Instance type ▼	Status check
<input type="checkbox"/>	ec2withkms	i-010ac647a6d84fe75	Pending	t2.micro	–

## Create S3 bucket

Amazon S3 > Buckets

Account snapshot - updated every 24 hours [All AWS Regions](#)

[View Storage Lens dashboard](#)

General purpose buckets

Directory buckets

General purpose buckets (1) [Info](#) [All AWS Regions](#)

Find buckets by name

< 1 >

Name ▲	AWS Region ▼	IAM Access Analyzer	Creation date ▼
bucketwithkms	Asia Pacific (Mumbai) ap-south-1	<a href="#">View analyzer for ap-south-1</a>	August 9, 2024, 17:01:17 (UTC+05:30)

Copy ARN

Empty

Delete

Create bucket

## Connect EC2 instance

[EC2](#) > [Instances](#) > [i-0961246ae21992897](#) > Modify IAM role

Modify IAM role [Info](#)

Attach an IAM role to your instance.

Instance ID

i-0961246ae21992897 (ec2-kms)

IAM role

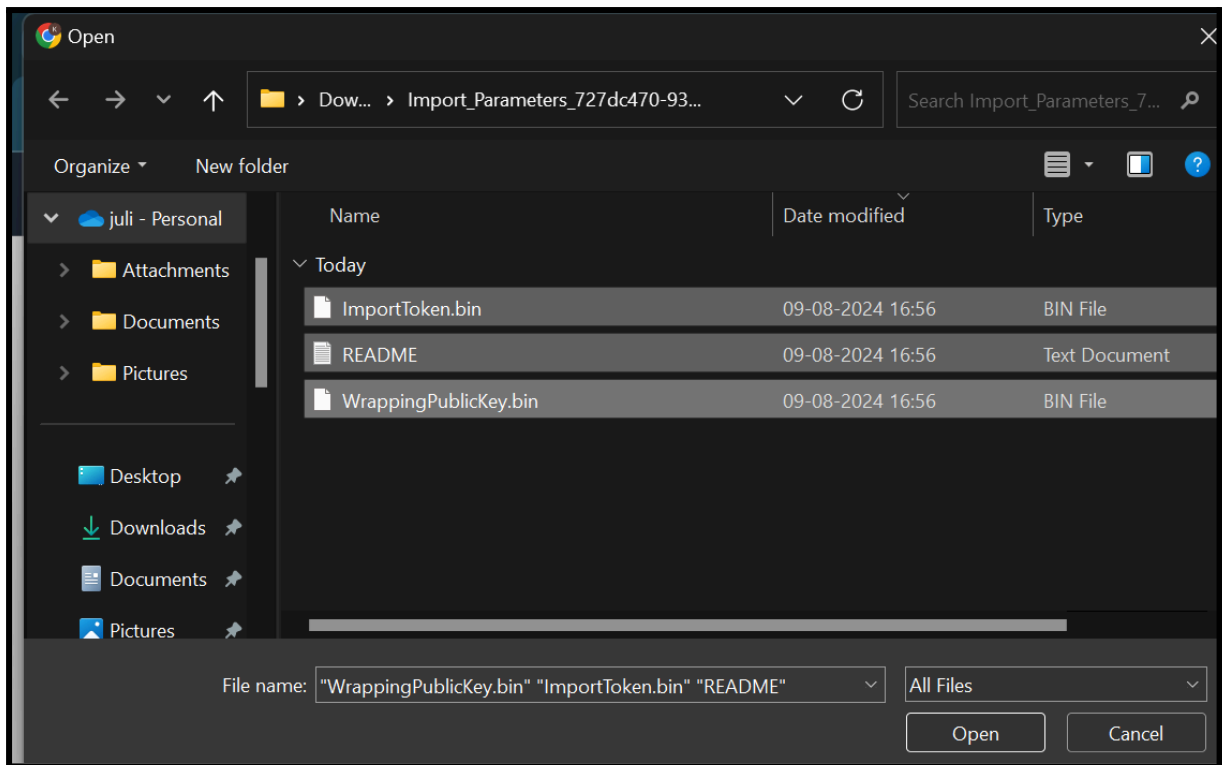
Select an IAM role to attach to your instance or create a new role if you haven't created any. The role you select replaces any roles that are currently attached to your instance.

role-with-kms ▼

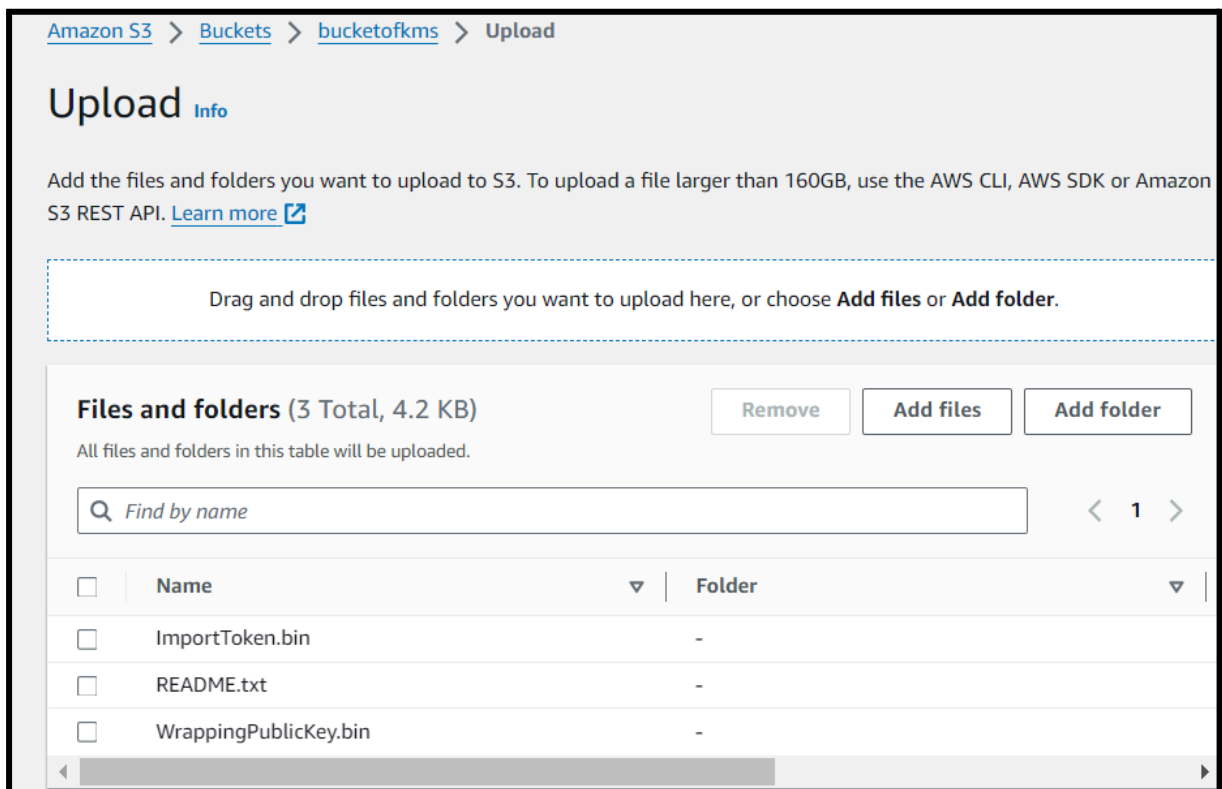
[Create new IAM role](#) [🔗](#)

Cancel

Update IAM role



## Upload in s3



Files and folders | Configuration

Files and folders (3 Total, 4.2 KB)

Find by name

Name	Folder	Type	Size	Status	Error
<a href="#">ImportToken...</a>	-	application/...	3.4 KB	✓ Succeeded	-
<a href="#">README.txt</a>	-	text/plain	278.0 B	✓ Succeeded	-
<a href="#">WrappingBu</a>	-	application/	550.0 B	✓ Succeeded	-

```
Complete!
[root@ip-172-31-0-12 ec2-user]# yum upgrade -y
Last metadata expiration check: 0:01:06 ago on Fri Aug 9 11:49:29 2024.
Dependencies resolved.
Nothing to do.
Complete!
[root@ip-172-31-0-12 ec2-user]# openssl version
bash: openssl: command not found
[root@ip-172-31-0-12 ec2-user]# openssl version
OpenSSL 3.0.8 7 Feb 2023 (Library: OpenSSL 3.0.8 7 Feb 2023)
[root@ip-172-31-0-12 ec2-user]# pwd
/home/ec2-user
[root@ip-172-31-0-12 ec2-user]# aws s3 cp s3://bucketwithkms /home/ec2-user --recursive
fatal error: Unable to locate credentials
[root@ip-172-31-0-12 ec2-user]# aws s3 cp s3://bucketofkms /home/ec2-user --recursive
fatal error: Unable to locate credentials
[root@ip-172-31-0-12 ec2-user]# aws --version
aws-cli/2.15.30 Python/3.9.16 Linux/6.1.102-108.177.amzn2023.x86_64 source/x86_64.amzn.2023 prompt/off
[root@ip-172-31-0-12 ec2-user]# aws configure
AWS Access Key ID [None]:
AWS Secret Access Key [None]:
Default region name [None]:
Default output format [None]:
[root@ip-172-31-0-12 ec2-user]#
[root@ip-172-31-0-12 ec2-user]# aws s3 cp s3://bucketofkms /home/ec2-user --recursive
[root@ip-172-31-0-12 ec2-user]# aws s3 cp s3://bucketofkms /home/ec2-user --recursive
[root@ip-172-31-0-12 ec2-user]# aws s3 cp s3://bucketofkms /home/ec2-user --recursive
download: s3://bucketofkms/ImportToken.bin to ./ImportToken.bin
download: s3://bucketofkms/WrappingPublicKey.bin to ./WrappingPublicKey.bin
download: s3://bucketofkms/README.txt to ./README.txt
[root@ip-172-31-0-12 ec2-user]#
```

```
openssl version

aws s3 cp s3://bucketname /home/ec2-user --recursive

openssl rand -out plaintextkeymaterial.bin 32

openssl pkeyutl -in plaintextkeymaterial.bin -out
EncryptedKeyMaterial.bin -inkey <wrapping key> -keyform der -pubin
-encrypt -pkeyopt rsa_padding_mode:oaep -pkeyopt
rsa_oaep_md:sha256
```

# aws s3 cp /home/ec2-user s3://bucketname --recursive

```
[root@ip-172-31-0-12 ec2-user]#
[root@ip-172-31-0-12 ec2-user]# aws s3 cp s3://bucketofkms /home/ec2-user --recursive
[root@ip-172-31-0-12 ec2-user]# aws s3 cp s3://bucketofkms /home/ec2-user --recursive
[root@ip-172-31-0-12 ec2-user]# aws s3 cp s3://bucketofkms /home/ec2-user --recursive
download: s3://bucketofkms/ImportToken.bin to ./ImportToken.bin
download: s3://bucketofkms/WrappingPublicKey.bin to ./WrappingPublicKey.bin
download: s3://bucketofkms/README.txt to ./README.txt
[root@ip-172-31-0-12 ec2-user]#
openssl rand -out plaintextkeymaterial.bin 32
[root@ip-172-31-0-12 ec2-user]# openssl rand -out plaintextkeymaterial.bin 32
[root@ip-172-31-0-12 ec2-user]# openssl pkeyutl -in plaintextkeymaterial.bin -out EncryptedKeyMaterial.bin -inkey WrappingPublicKey.bin -keyform der -pubin -encrypt -pkeyopt rsa_padding_mode:oaep -pkeyopt rsa_oaep_md:sha256
[root@ip-172-31-0-12 ec2-user]#
[root@ip-172-31-0-12 ec2-user]# openssl pkeyutl -in plaintextkeymaterial.bin -out EncryptedKeyMaterial.bin -inkey WrappingPublicKey.bin -keyform der -pubin -encrypt -pkeyopt rsa_padding_mode:oaep -pkeyopt rsa_oaep_md:sha256
[root@ip-172-31-0-12 ec2-user]# aws s3 cp /home/ec2-user s3://bucketofkms --recursive
upload: ./bash_logout to s3://bucketofkms/.bash_logout
upload: ./ImportToken.bin to s3://bucketofkms/ImportToken.bin
upload: ./WrappingPublicKey.bin to s3://bucketofkms/WrappingPublicKey.bin
upload: ./EncryptedKeyMaterial.bin to s3://bucketofkms/EncryptedKeyMaterial.bin
upload: ./plaintextkeymaterial.bin to s3://bucketofkms/plaintextkeymaterial.bin
upload: ./README.txt to s3://bucketofkms/README.txt
upload: ./bash_profile to s3://bucketofkms/.bash_profile
upload: ./ssh/authorized_keys to s3://bucketofkms/.ssh/authorized_keys
upload: ./bashrc to s3://bucketofkms/.bashrc
[root@ip-172-31-0-12 ec2-user]#
```

## >aws s3 cp /home/ec2-user/EncryptedKeyMaterial.bin s3://bucketofkms

```
[root@ip-172-31-0-12 ec2-user]# aws s3 cp /home/ec2-user/EncryptedKeyMaterial.bin s3://bucketofkms
upload: ./EncryptedKeyMaterial.bin to s3://bucketofkms/EncryptedKeyMaterial.bin
[root@ip-172-31-0-12 ec2-user]#
```

## Download encrypted file

[Amazon S3](#) > [Buckets](#) > [bucketofkms](#) > EncryptedKeyMaterial.bin

EncryptedKeyMaterial.bin

Copy S3 URI

Download

Open

Object actions

Properties

Permissions

Versions

Object overview

Owner

a26699cd1c24cfae5de1c1a262335e93f4ef57038e2af2bd76969ddb889f907

AWS Region

US East (Ohio) us-east-2

Last modified

August 9, 2024, 17:43:43 (UTC+05:30)

Size

512.0 B

Type

bin

S3 URI

s3://bucketofkms/EncryptedKeyMaterial.bin

Amazon Resource Name (ARN)

arn:aws:s3:::bucketofkms/EncryptedKeyMaterial.bin

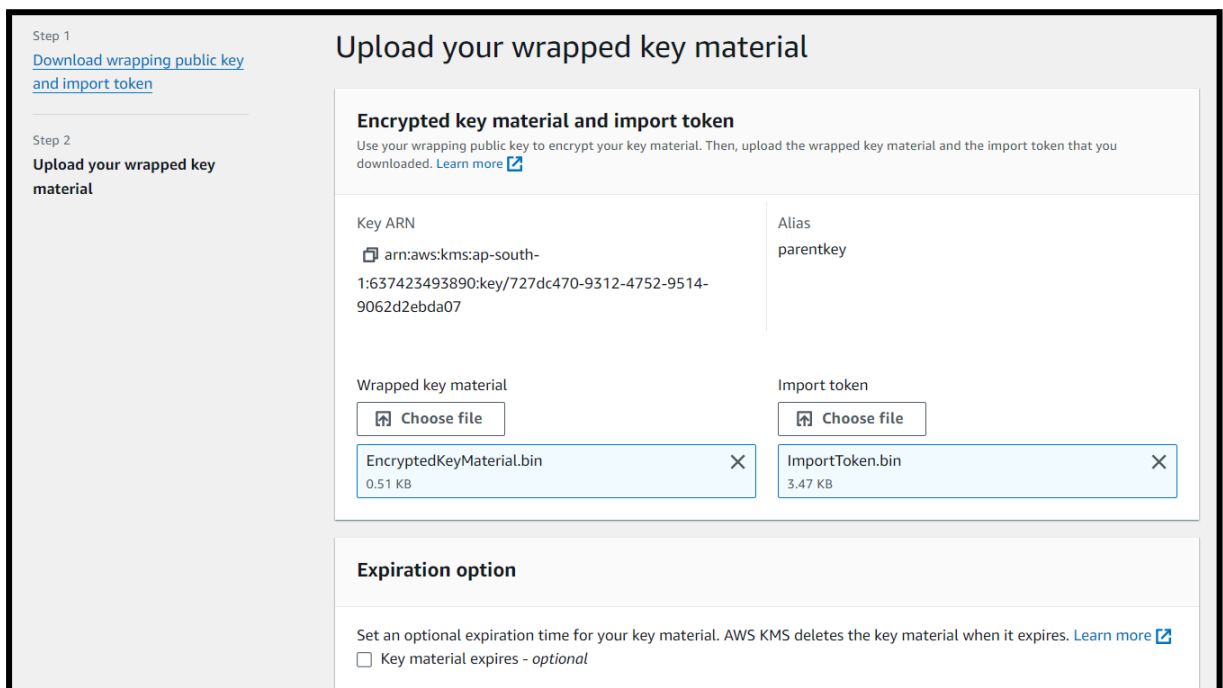
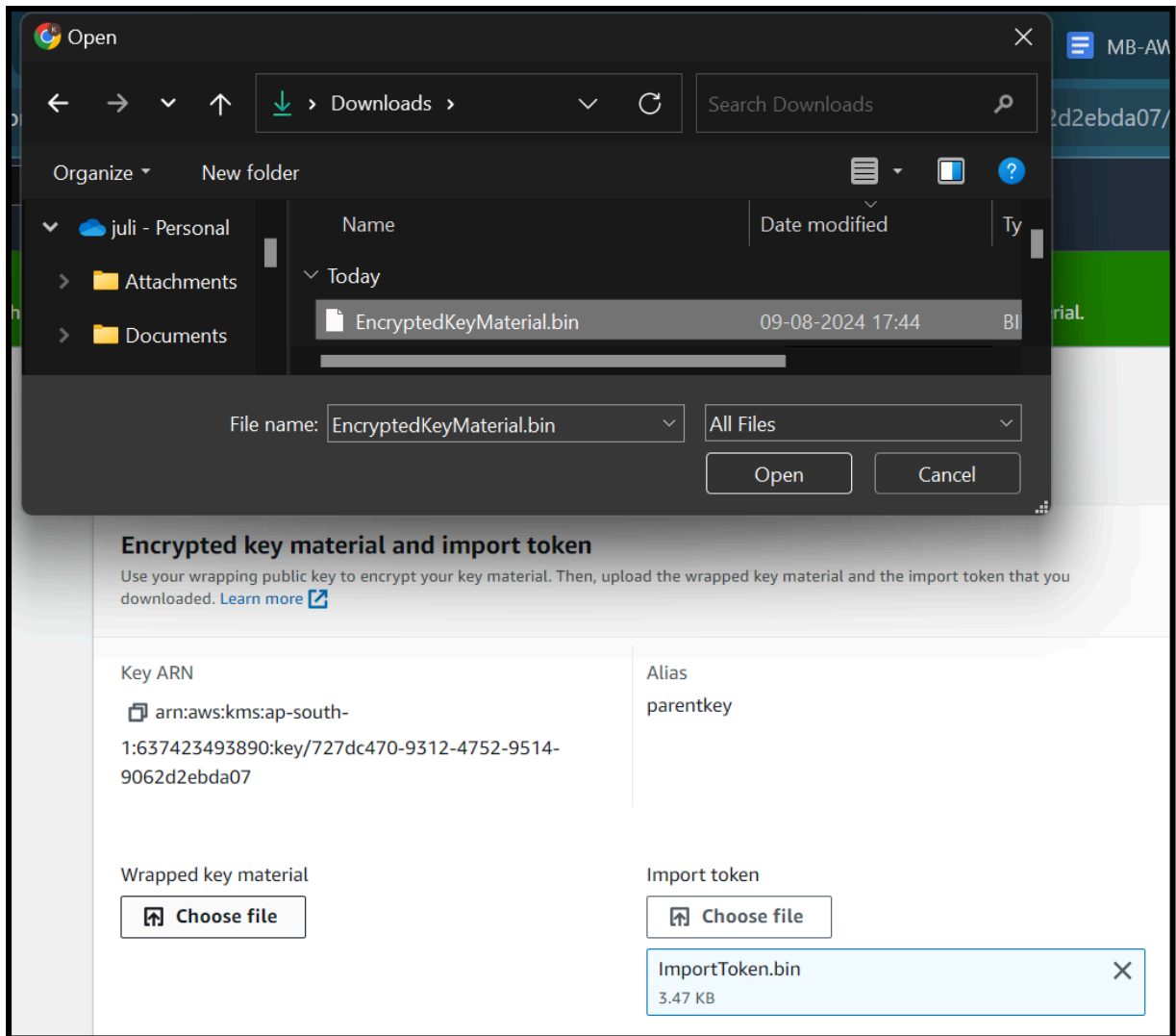
Entity tag (Etag)

57310bc16db18d61774efd8fce32e6e6

Object URL

<https://bucketofkms.s3.us-east-2.amazonaws.com/EncryptedKeyMaterial.bin>

## Upload



# Key is enabled now

[KMS](#) > Customer managed keys

Customer managed keys (2)

Key actions ▼Create key

Q

Filter keys by properties or tags

< 1 > ⚙

<input type="checkbox"/>	Aliases <span>▼</span>	Key ID <span>▼</span>	Status	Key type <span>▼</span>	Key spec <span>ⓘ</span>	Key usage
<input type="checkbox"/>	<a href="#">parentkey</a>	<a href="#">727dc470-9312-475...</a>	Enabled	Symmetric	SYMMETRIC_DEFAULT	Encrypt and decrypt



Date : 12-8-2024

## AIM : KEY MANAGEMENT SERVICE WITH EC2

### Step 1 : launch an EC2 machine

Instances (1) <a href="#">Info</a>							
<input type="text" value="Find Instance by attribute or tag (case-sensitive)"/>				All states ▾		< 1 > ⚙	
<input type="checkbox"/>	Name <a href="#">✎</a> ▾	Instance ID	Instance state ▾	Instance type ▾	Status check	Alarm status	A
<input type="checkbox"/>	my-ec2	i-04d5dc1bf7872a742	Running <a href="#">🔍</a> <a href="#">🔍</a>	t2.micro	🕒 Initializing	<a href="#">View alarms +</a>	a

### Step 2 : create KMS

## Configure key

**Key type** [Help me choose](#) [🔗](#)

☒ **Symmetric**  
A single key used for encrypting and decrypting data or generating and verifying HMAC codes

☐ **Asymmetric**  
A public and private key pair used for encrypting and decrypting data, signing and verifying messages, or deriving shared secrets

**Key usage** [Help me choose](#) [🔗](#)

☒ **Encrypt and decrypt**  
Use the key only to encrypt and decrypt data.

☐ **Generate and verify MAC**  
Use the key only to generate and verify hash-based message authentication codes (HMAC).

▼ Advanced options

Key material origin

Key material origin is a KMS key property that represents the source of the key material when creating the KMS key. [Help me choose](#)

☐ KMS - *recommended*  
AWS KMS creates and manages the key material for the KMS key.

☒ External (Import Key material)  
You create and import the key material for the KMS key.

☐ AWS CloudHSM key store  
AWS KMS creates the key material in the AWS CloudHSM cluster of your AWS CloudHSM key store.

☐ External key store  
The key material for the KMS key is in an external key manager outside of AWS.

You can import key material from your key management infrastructure into AWS KMS and use it like any other AWS KMS key.

☐ I understand the [security and durability implications](#) of using an imported key.

Regionality

Create your KMS key in a single AWS Region (default) or create a KMS key that you can replicate into multiple AWS Regions. [Help me choose](#)

☒ Single-Region key  
Never allow this key to be replicated into other Regions

[KMS](#) > Customer managed keys

Customer managed keys (1)

Key actions ▼

Create key

Filter keys by properties or tags

< 1 > ⚙

<input type="checkbox"/>	Aliases ▼	Key ID ▼	Status	Key type ▼	Key spec ⓘ	Key usage
<input type="checkbox"/>	<a href="#">kms-key</a>	<a href="#">33936a2e-7...</a>	Pending imp...	Symmetric	SYMMETRIC_...	Encrypt and ...

Download

Wrapping public key  
WrappingPublicKey.bin

Import token  
ImportToken.bin

Read me instructions  
README.txt

ⓘ This wrapping public key and import token will expire in 24 hours.

📄 Download wrapping public key and import token


Cancel

Next

## Encrypted key material and import token

Use your wrapping public key to encrypt your key material. Then, upload the wrapped key material and the import token that you downloaded. [Learn more](#)


Key ARN

 arn:aws:kms:ap-southeast-1:637423493890:key/33936a2e-790c-4c3b-9098-efc66d70d9fa


Alias

kms-key

Wrapped key material

 Choose file

Import token

 Choose file

ImportToken.bin  
3.47 KB


## Create s3 bucket


kms-bucket-ec2 [Info](#)


[Objects](#) | [Properties](#) | [Permissions](#) | [Metrics](#) | [Management](#) | [Access Points](#)

Objects (0) [Info](#)



 Copy S3 URI

 Copy URL

 Download

 Open

Delete

Actions ▼

Create folder

 Upload

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

 Find objects by prefix

☐ Show versions


< 1 > 

<input type="checkbox"/>	Name	Type	Last modified	Size	Storage class
--------------------------	------	------	---------------	------	---------------

No objects

You don't have any objects in this bucket.

## Upload files

 **Upload succeeded**  
View details below.

Files and folders

Configuration

Files and folders (3 Total, 4.2 KB)

Name	Folder	Type	Size	Status	Error
<a href="#">ImportToken.bin</a>	-	application/...	3.4 KB	Succeeded	-
<a href="#">README.txt</a>	-	text/plain	278.0 B	Succeeded	-
<a href="#">WrappingPublicKey.bin</a>	-	application/...	550.0 B	Succeeded	-

openssl version

aws s3 cp s3://bucketname /home/ec2-user --recursive


openssl rand -out plaintextkeymaterial.bin 32

openssl pkeyutl -in plaintextkeymaterial.bin -out EncryptedKeyMaterial.bin -inkey <wrapping key> -keyform der -pubin -encrypt -pkeyopt rsa\_padding\_mode:oaep -pkeyopt rsa\_oaep\_md:sha256

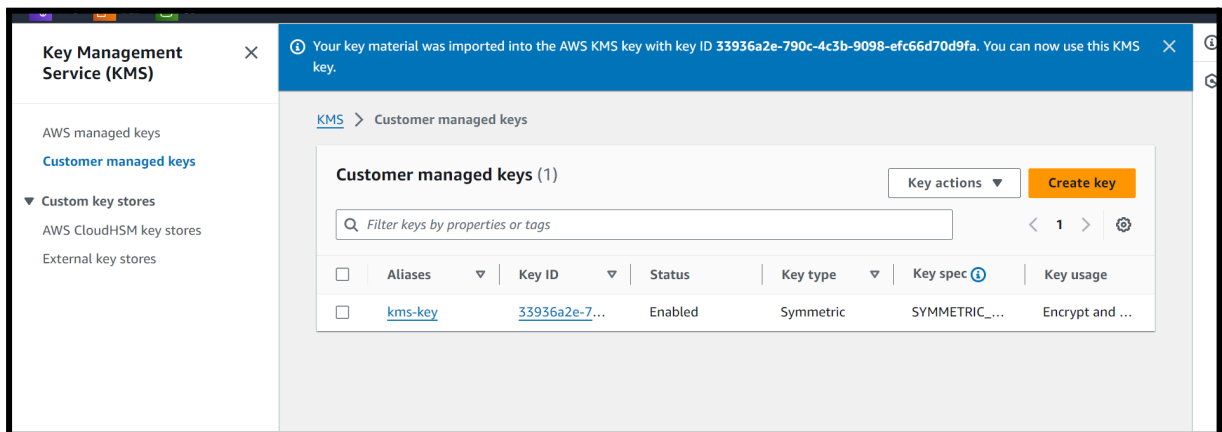
aws s3 cp /home/ec2-user s3://bucketname --recursive

```

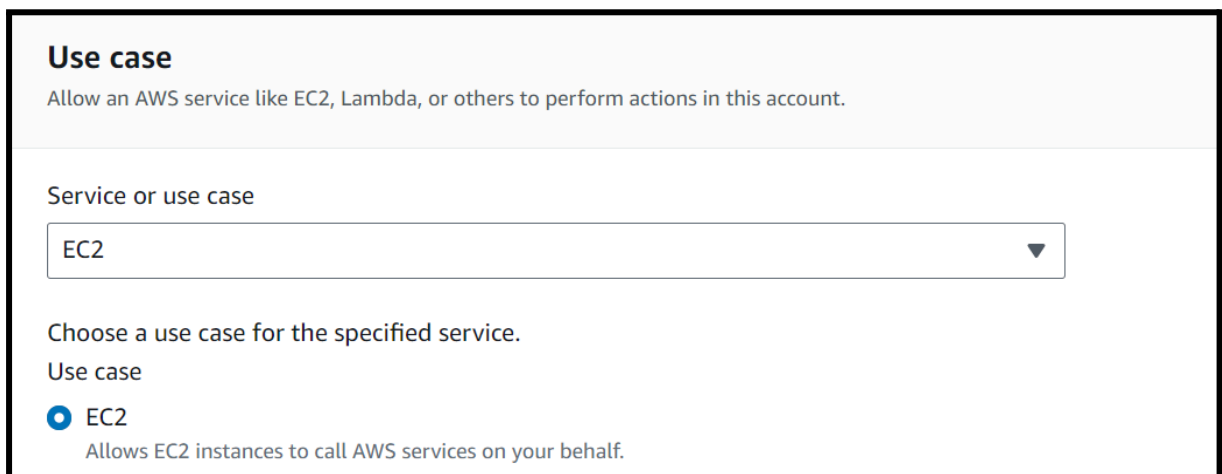
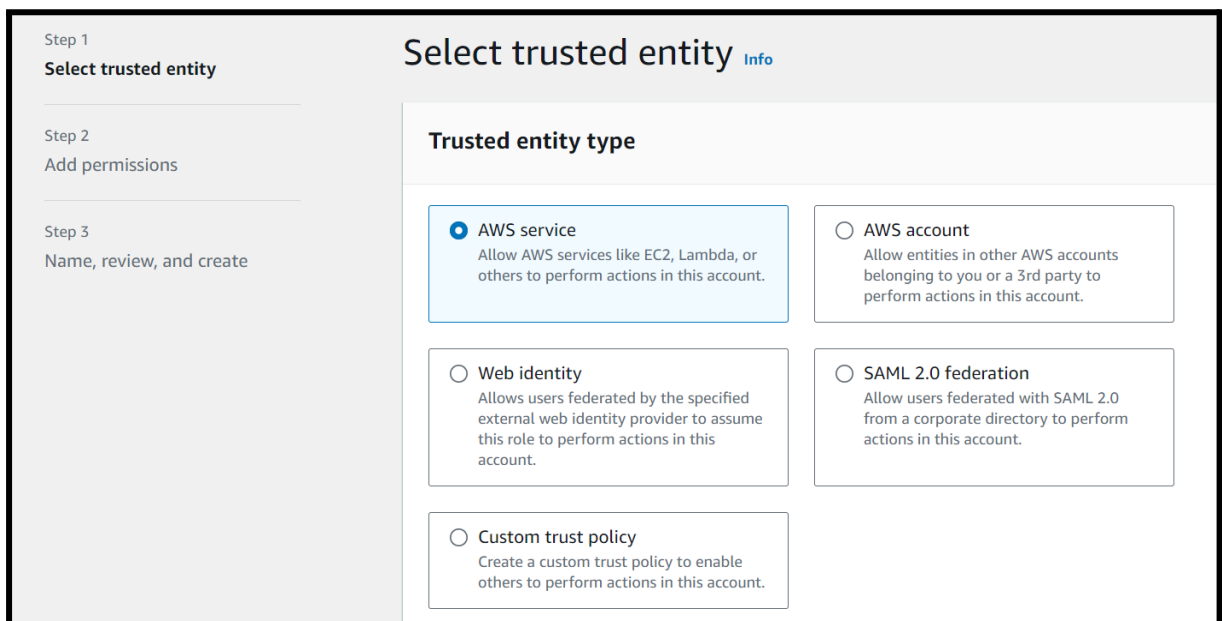
OpenSSL 3.0.8 7 Feb 2023 (Library: OpenSSL 3.0.8 7 Feb 2023)
[root@ip-172-31-44-52 ec2-user]# aws s3 cp s3://kms-bucket-ec2 /home/ec2-user --recursive
download: s3://kms-bucket-ec2/ImportToken.bin to ./ImportToken.bin
download: s3://kms-bucket-ec2/WrappingPublicKey.bin to ./WrappingPublicKey.bin
[root@ip-172-31-44-52 ec2-user]# openssl rand -out plaintextkeymaterial.bin 32
[root@ip-172-31-44-52 ec2-user]# openssl pkeyutl -in plaintextkeymaterial.bin -out EncryptedKeyMaterial.bin -inkey <wrapping key> -keyform der -pubin -encrypt -pkeyopt rsa_padding_mode:oaep -pkeyopt rsa_oaep_md:sha256
bash: wrapping: No such file or directory
[root@ip-172-31-44-52 ec2-user]# openssl pkeyutl -in plaintextkeymaterial.bin -out EncryptedKeyMaterial.bin -inkey WrappingPublicKey.bin -keyform der -pubin -encrypt -pkeyopt rsa_padding_mode:oaep -pkeyopt rsa_oaep_md:sha256
[root@ip-172-31-44-52 ec2-user]# aws s3 cp /home/ec2-user s3://kms-bucket-ec2 --recursive
upload: ./ImportToken.bin to s3://kms-bucket-ec2/ImportToken.bin
upload: ./file.txt to s3://kms-bucket-ec2/file.txt
upload: ./bash_logout to s3://kms-bucket-ec2/.bash_logout
upload: ./plaintextkeymaterial.bin to s3://kms-bucket-ec2/plaintextkeymaterial.bin
upload: ./WrappingPublicKey.bin to s3://kms-bucket-ec2/WrappingPublicKey.bin
upload: ./ssh/authorized_keys to s3://kms-bucket-ec2/.ssh/authorized_keys
upload: ./bashrc to s3://kms-bucket-ec2/.bashrc
upload: ./EncryptedKeyMaterial.bin to s3://kms-bucket-ec2/EncryptedKeyMaterial.bin
upload: ./bash_profile to s3://kms-bucket-ec2/.bash_profile
[root@ip-172-31-44-52 ec2-user]#
    
```

i-04d5dc1bf7872a742 (my-ec2)
 

PublicIPs: 47.129.204.84 PrivateIPs: 172.31.44.52



### Step 3 : create role



Role details

Role name

Enter a meaningful name to identify this role.

kms-cse-ec2-role

Maximum 64 characters. Use alphanumeric and '+=, @-\_' characters.

Description

Add a short explanation for this role.

Allows EC2 instances to call AWS services on your behalf.

Maximum 1000 characters. Use letters (A-Z and a-z), numbers (0-9), tabs, new lines, or any of the following characters: \_+=, @-/[{}!#\$%^&\*()';:"`

Roles (3) Info

Refresh

Delete

Create role

An IAM role is an identity you can create that has specific permissions with credentials that are valid for short durations. Roles can be assumed by entities that you trust.

Search

< 1 >

Settings

<input type="checkbox"/>	Role name	▲	Trusted entities
<input type="checkbox"/>	<a href="#">kms-cse-ec2-role</a>		AWS Service: ec2

Go to > ec2 > modify security

Instances (1/1) Info

Refresh

Connect

Instance state ▼

Actions ▲

Launch instances ▼

Find Instance by attribute or tag (case-sensitive)

All states

<input checked="" type="checkbox"/>	Name ↗	Instance ID	Instance state	Instance type
<input checked="" type="checkbox"/>	my-ec2	i-04d5dc1bf7872a742	Running	t2.micro

Connect

View details

Manage instance state

Instance settings ▶

Networking ▶

Security ▶

Image and templates ▶

Monitor and troubleshoot ▶


Change security groups	Security ▶
Get Windows password	Image and templates ▶
Modify IAM role	Monitor and troubleshoot ▶

[EC2](#) > [Instances](#) > [i-04d5dc1bf7872a742](#) > [Modify IAM role](#)

# Modify IAM role [Info](#)

Attach an IAM role to your instance.

Instance ID


 [i-04d5dc1bf7872a742](#) (my-ec2)

IAM role

Select an IAM role to attach to your instance or create a new role if you haven't created any. The role you select replaces any roles that are currently attached to your instance.

kms-cse-ec2-role

▼




[Create new IAM role](#) 

Cancel

Update IAM role

KMS > key users > add IAM role

Key Management Service (KMS) 

AWS managed keys

Customer managed keys


▼ Custom key stores


AWS CloudHSM key stores

External key stores

Key users (0) 

Add Remove

The following IAM users and roles can use this key for cryptographic operations. They can also allow AWS services that are integrated with KMS to use the key on their behalf. [Learn more](#) 

 Search Key users

< 1 >

	Name	Path	Type
Empty Resources			
No resources to display			

Add key users

The following IAM users and roles can use this key to encrypt and decrypt data from within applications and when using AWS services integrated with KMS.

Key users (5)

Q kms

X

1 matches

< 1 >

<input type="checkbox"/>	Name	Path	Type
<input type="checkbox"/>	kms-cse-ec2-role	/	Role

Cancel

Add

## Connect EC2 machine

1. Create a file with data which you want to encrypt

echo "Hello,Pooja! Encrypt Me." > file.txt

```
[ec2-user@ip-172-31-44-52 ~]$ sudo su
[root@ip-172-31-44-52 ec2-user]# yum update -y
Last metadata expiration check: 0:14:19 ago on Mon Aug 12 10:59:03 2024.
Dependencies resolved.
Nothing to do.
Complete!
[root@ip-172-31-44-52 ec2-user]# yum upgrade -y
Last metadata expiration check: 0:14:27 ago on Mon Aug 12 10:59:03 2024.
Dependencies resolved.
Nothing to do.
Complete!
[root@ip-172-31-44-52 ec2-user]# echo "Hello,Pooja! Encrypt Me." > file.txt
[root@ip-172-31-44-52 ec2-user]#
```

i-04d5dc1bf7872a742 (my-ec2)

PublicIPs: 47.129.204.84 PrivateIPs: 172.31.44.52

## Generate data key using CMK



```
> aws kms generate-data-key --key-id alias/kms-key --key-spec AES_256
--region ap-southeast-1
```

```
[root@ip-172-31-44-52 ec2-user]# echo "Hello, Pooja! Please encrypt me ." > file.txt
[root@ip-172-31-44-52 ec2-user]# aws kms generate-data-key --key-id alias/kms-key --key-spec AES_256 --region ap-southeast-1
{
  "CiphertextBlob": "AQIDAHgH/QSZkrrkP+tAydUWGaUNhoxg44VQxAiTEW7NYikXUwEpRTK5vCJ8QTdHe4nA2fGzAAAAfjB8BgkqhkiG9w0BBwagbzBtAgEAMGgGCSqGSIb3DQEHATAeBglgghkgBZQMEAS4wEQQMCQ8O6+ZSy0sd9ftgAgEQgDv10qIINxOUw74TX9rOubKKrImbilNKOQ074rUEN8E6n7S2B5EIC+0PZivDtYpKZ0OsJ+KJMK5h7sm1EQ=="
,
  "Plaintext": "v/SjBZL9hWrfAscs3oQBuDo6S5F8G/JsFOzlkpgGzFk=",
  "KeyId": "arn:aws:kms:ap-southeast-1:637423493890:key/33936a2e-790c-4c3b-9098-efc66d70d9fa"
}
[root@ip-172-31-44-52 ec2-user]#
```

Create a data key file which contain decoded Plaintext

```
> echo v/SjBZL9hWrfAscs3oQBuDo6S5F8G/JsFOzlkpgGzFk= | base64
--decode > datakey
```

```
[root@ip-172-31-44-52 ec2-user]# echo AQIDAHgH/QSZkrrkP+tAydUWGaUNhoxg44VQxAiTEW7NYikXUwEpRTK5vCJ8QTdHe4nA2fGzAAAAfjB8BgkqhkiG9w0BBwagbzBtAgEAMGgGCSqGSIb3DQEHATAeBglgghkgBZQMEAS4wEQQMCQ8O6+ZSy0sd9ftgAgEQgDv10qIINxOUw74TX9rOubKKrImbilNKOQ074rUEN8E6n7S2B5EIC+0PZivDtYpKZ0OsJ+KJMK5h7sm1EQ= | base64 --decode > encrypteddatakey
[root@ip-172-31-44-52 ec2-user]#
```

```
[root@ip-172-31-44-52 ec2-user]# echo v/SjBZL9hWrfAscs3oQBuDo6S5F8G/JsFOzlkpgGzFk= | base64 --decode > datakey
[root@ip-172-31-44-52 ec2-user]#
```

7. Create a encrypted data key file which contains decoded

```
> echo 7. Create a encrypted data key file which contains decoded
echo <ciphertext-bob> | base64 --decode > encrypteddatakey
| base64 --decode > encrypteddatakey
```

Encrypt file using data key

```
openssl enc -in ./file.txt -out ./encrypted-file.txt -e -aes256 -k fileb://./datakey
```

```
ls -la
total 32
-rw-r--r--. 1 root root 512 Aug 12 12:07 EncryptedKeyMaterial.bin
-rw-r--r--. 1 root root 3466 Aug 12 11:58 ImportToken.bin
-rw-r--r--. 1 root root 550 Aug 12 11:58 WrappingPublicKey.bin
-rw-r--r--. 1 root root 32 Aug 12 12:18 datakey
-rw-r--r--. 1 root root 64 Aug 12 12:22 encrypted-file.txt
-rw-r--r--. 1 root root 184 Aug 12 12:21 encrypteddatakey
-rw-r--r--. 1 root root 34 Aug 12 12:15 file.txt
-rw-r--r--. 1 root root 32 Aug 12 12:06 plaintextkeymaterial.bin
[root@ip-172-31-44-52 ec2-user]# cat file.,,txt
cat: file.,,txt: No such file or directory
[root@ip-172-31-44-52 ec2-user]# cat file.txt
Hello, Pooja! Please encrypt me .
[root@ip-172-31-44-52 ec2-user]#
```

Remove datakey and datafile

```
rm datakey
rm file.txt
```

```
[root@ip-172-31-44-52 ec2-user]# rm datakey
rm: remove regular file 'datakey'? y
[root@ip-172-31-44-52 ec2-user]# rm file.txt
rm: remove regular file 'file.txt'? y
[root@ip-172-31-44-52 ec2-user]#
```

## .Decrypt "encrypted data key"

aws kms decrypt --ciphertext-blob fileb:///encrypteddatakey --region ap-south-1

```
[root@ip-172-31-44-52 ec2-user]# aws kms decrypt --ciphertext-blob fileb:///encrypteddatakey --region ap-southeast-1
{
  "KeyId": "arn:aws:kms:ap-southeast-1:637423493890:key/33936a2e-790c-4c3b-9098-efc66d70d9fa",
  "Plaintext": "v/SjBZL9hWrfAscs3oQBudo6S5F8G/JsFOzlkpgGzFk=",
  "EncryptionAlgorithm": "SYMMETRIC_DEFAULT"
}
```

## Create a Data Key file which contains plaintext

echo <plaintext> | base64 --decode > datakey

## Decrypt "encrypted data file" using the data key

openssl enc -in ./encrypted-file.txt -out ./file.txt -d -aes256 -k fileb:///datakey

```
[root@ip-172-31-44-52 ec2-user]# echo v/SjBZL9hWrfAscs3oQBudo6S5F8G/JsFOzlkpgGzFk= | base64 --decode > datakey
[root@ip-172-31-44-52 ec2-user]# openssl enc -in ./encrypted-file.txt -out ./file.txt -d -aes256 -k fileb:///datakey
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[root@ip-172-31-44-52 ec2-user]# ll
total 32
-rw-r--r--. 1 root root 512 Aug 12 12:07 EncryptedKeyMaterial.bin
-rw-r--r--. 1 root root 3466 Aug 12 11:58 ImportToken.bin
-rw-r--r--. 1 root root 550 Aug 12 11:58 WrappingPublicKey.bin
-rw-r--r--. 1 root root 32 Aug 12 12:29 datakey
-rw-r--r--. 1 root root 64 Aug 12 12:22 encrypted-file.txt
-rw-r--r--. 1 root root 184 Aug 12 12:21 encrypteddatakey
-rw-r--r--. 1 root root 34 Aug 12 12:29 file.txt
-rw-r--r--. 1 root root 32 Aug 12 12:06 plaintextkeymaterial.bin
[root@ip-172-31-44-52 ec2-user]#
```

```
[root@ip-172-31-44-52 ec2-user]# cat file.txt
Hello, Pooja! Please encrypt me .
[root@ip-172-31-44-52 ec2-user]#
```