

**Qualcomm Car-to-Cloud Platform**

**KMS IMPLEMENTATION DOCUMENT**

**Version No.2.0**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Prepared By / Last Updated By** | **Reviewed By** | **Approved By** |
| **Name** | Sneha John | Sudheer | Sudheer |
| **Role** | PA | Sr Manager | Sr Manager |
| **Signature** |  |  |  |
| **Date** | 18/05/2021 | 11/06/2021 | 11/06/2021 |

May 18, 2021

Cognizant

Contents

[1. KMS IMPLEMENTATION 2](#_Toc74563123)

[1.1. The Dependencies 2](#_Toc74563124)

[1.2. The Encrypt Method 3](#_Toc74563125)

[1.3. The Decrypt Method 3](#_Toc74563126)

[1.4. The Reconnect Method 4](#_Toc74563127)

# KMS IMPLEMENTATION

Consider a sample KMS application – that is meant to encrypt and decrypt the message payload using a key via cloud services. For this application AWS Key Management Service (KMS) is used.

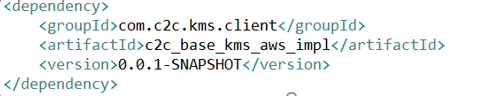
The project can be divided into two sections –

* **c2c\_base\_key\_management\_service\_intf** which contains the interfaces containing signatures of functions related to encrypt, decrypt and reconnect.
* **c2c\_base\_kms\_aws\_impl** which contains the implementation of the interface **c2c\_base\_key\_management\_service\_intf**. The implementations will be responsible for encryption and decryption using AWS KMS and client reconnection.

To use the implementation project, we are going to be having a client caller project. This caller project will consume the implementation methods**.**

## The Dependencies

Create the jars of both the interface and the implementation project. Add these jars to the POM file of the caller project. The required dependencies that need to be added are:



## The Encrypt Method

We create a bean of the **IC2CKeyManagement**. This is done by calling the build function of the Builder Class for **C2CKeyManagementConnectionConfig** which is the model configuration class from **c2c\_base\_key\_management\_service\_intf** and instantiate it by passing the credentials and KMS endpoint. The build function returns an object of **C2CKeyManagementConnectionConfig.**

This object is passed as a parameter to the **C2CKeyManagementImpl** tocreate the object and return it as the bean of the **IC2CKeyManagement.** Using this bean, call the encrypt method to encrypt the message /payload using a key. The method parameters include the payload to be encrypted, the key id of the KMS key, the key version id, key ring id and encryption type, which is a Boolean that determines whether the key is symmetric or asymmetric. Key version id and ring id are null for AWS implementation.



***Exceptions***

**KeyManagementApplicationException** is thrown when a generic AWSKMS exception occurs or when there is an error during KMS client creation or during validation of inputs.

**Limitations**

The maximum size of payload using a symmetric key is 4 KB and while using an asymmetric key, it is 446 bytes.

## The Decrypt Method

We create a bean of the **IC2CKeyManagement**. This is done by calling the build function of the Builder Class for **C2CKeyManagementConnectionConfig** which is the model configuration class from **c2c\_base\_key\_management\_service\_intf** and instantiate it by passing the credentials and KMS endpoint. The build function returns an object of **C2CKeyManagementConnectionConfig.** This object is passed as a parameter to the **C2CKeyManagementImpl** tocreate the object and return it as the bean of the **IC2CKeyManagement.** Using this bean, call the decrypt method to decrypt the message /payload using a key. The method parameters include the payload to be decrypted, the key id of the KMS key, key version id, key ring id and encryption type which is a Boolean that determines whether the key is symmetric or not. Key version id and key ring id are null for AWS implementation.

***Exceptions:***

**KeyManagementApplicationException** is thrown when a generic AWSKMS exception occurs or when there is an error during KMS client creation or during validation of inputs.

**Limitations**

The maximum size of payload using a symmetric key is 4 KB and while using an asymmetric key, it is 446 bytes.

## The Reconnect Method

We create a bean of the **IC2CKeyManagement**. This is done by calling the build function of the Builder Class for **C2CKeyManagementConnectionConfig** which is the model configuration class from **c2c\_base\_key\_management\_service\_intf** and instantiate it by passing the credentials and KMS endpoint from the external properties. The build function returns an object of **C2CKeyManagementConnectionConfig.** This object is passed as a parameter to the **C2CKeyManagementImpl** tocreate the object and return it as the bean of the **IC2CKeyManagement.** Using this bean, call the reconnect method to re-establish the connection for any failure connection issues with the AWS KMS.