Legal disclaimer

Alibaba Cloud reminds you to carefully read and fully understand the terms and conditions of this legal disclaimer before you read or use this document. If you have read or used this document, it shall be deemed as your total acceptance of this legal disclaimer.

1. You shall download and obtain this document from the Alibaba Cloud website or other Alibaba Cloud-authorized channels, and use this document for your own legal business activities only. The content of this document is considered confidential information of Alibaba Cloud. You shall strictly abide by the confidentiality obligations. No part of this document shall be disclosed or provided to any third party for use without the prior written consent of Alibaba Cloud.

2. No part of this document shall be excerpted, translated, reproduced, transmitted, or disseminated by any organization, company or individual in any form or by any means without the prior written consent of Alibaba Cloud.

3. The content of this document may be changed because of product version upgrade, adjustment, or other reasons. Alibaba Cloud reserves the right to modify the content of this document without notice and an updated version of this document will be released through Alibaba Cloud-authorized channels from time to time. You should pay attention to the version changes of this document as they occur and download and obtain the most up-to-date version of this document from Alibaba Cloud-authorized channels.

4. This document serves only as a reference guide for your use of Alibaba Cloud products and services. Alibaba Cloud provides this document based on the "status quo", "being defective", and "existing functions" of its products and services. Alibaba Cloud makes every effort to provide relevant operational guidance based on existing technologies. However, Alibaba Cloud hereby makes a clear statement that it in no way guarantees the accuracy, integrity, applicability, and reliability of the content of this document, either explicitly or implicitly. Alibaba Cloud shall not take legal responsibility for any errors or lost profits incurred by any organization, company, or individual arising from download, use, or trust in this document. Alibaba Cloud shall not, under any circumstances, take responsibility for any indirect, consequential, punitive, contingent, special, or punitive damages, including lost profits arising from the use or trust in this document (even if Alibaba Cloud has been notified of the possibility of such a loss).

5. By law, all the contents in Alibaba Cloud documents, including but not limited to pictures, architecture design, page layout, and text description, are intellectual property of Alibaba Cloud and/or its affiliates. This intellectual property includes, but is not limited to, trademark rights, patent rights, copyrights, and trade secrets. No part of this document shall be used, modified, reproduced, publicly transmitted, changed, disseminated, distributed, or published without the prior written consent of Alibaba Cloud and/or its affiliates. The names owned by Alibaba Cloud shall not be used, published, or reproduced for marketing, advertising, promotion, or other purposes without the prior written consent of Alibaba Cloud. The names owned by Alibaba Cloud include, but are not limited to, "Alibaba Cloud", "Aliyun", "HiChina", and other brands of Alibaba Cloud and/or its affiliates, which appear separately or in combination, as well as the auxiliary signs and patterns of the preceding brands, or anything similar to the company names, trade names, trademarks, product or service names, domain names, patterns, logos, marks, signs, or special descriptions that third parties identify as Alibaba Cloud and/or its affiliates.

6. Please directly contact Alibaba Cloud for any errors of this document.
# Document conventions

<table>
<thead>
<tr>
<th>Style</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Danger</td>
<td>A danger notice indicates a situation that will cause major system changes, faults, physical injuries, and other adverse results.</td>
<td>!!! Danger: Resetting will result in the loss of user configuration data.</td>
</tr>
<tr>
<td>Warning</td>
<td>A warning notice indicates a situation that may cause major system changes, faults, physical injuries, and other adverse results.</td>
<td>!!! Warning: Restarting will cause business interruption. About 10 minutes are required to restart an instance.</td>
</tr>
<tr>
<td>Notice</td>
<td>A caution notice indicates warning information, supplementary instructions, and other content that the user must understand.</td>
<td>!!! Notice: If the weight is set to 0, the server no longer receives new requests.</td>
</tr>
<tr>
<td>Note</td>
<td>A note indicates supplemental instructions, best practices, tips, and other content.</td>
<td>!!! Note: You can use Ctrl + A to select all files.</td>
</tr>
<tr>
<td>&gt;</td>
<td>Closing angle brackets are used to indicate a multi-level menu cascade.</td>
<td>Click Settings &gt; Network &gt; Set network type.</td>
</tr>
<tr>
<td>Bold</td>
<td>Bold formatting is used for buttons, menus, page names, and other UI elements.</td>
<td>Click OK.</td>
</tr>
<tr>
<td>Courier font</td>
<td>Courier font is used for commands</td>
<td>Run the <code>cd /d C:/window</code> command to enter the Windows system folder.</td>
</tr>
<tr>
<td>Italic</td>
<td>Italic formatting is used for parameters and variables.</td>
<td><code>bae log list --instanceid Instance_ID</code></td>
</tr>
<tr>
<td>[] or [a</td>
<td>b]</td>
<td>This format is used for an optional value, where only one item can be selected.</td>
</tr>
<tr>
<td>() or (a</td>
<td>b)</td>
<td>This format is used for a required value, where only one item can be selected.</td>
</tr>
</tbody>
</table>
# Table of Contents

1. Overview ......................................................... 05
2. Activate KMS ..................................................... 06
3. Manage and use keys ........................................... 07
   3.1. Create a CMK ............................................. 07
   3.2. Encrypt resources of cloud services .................. 08
   3.3. Sample code for data encryption ...................... 11
4. Manage and use secrets ....................................... 13
   4.1. Create a secret .......................................... 13
   4.2. Sample code for Secrets Manager ..................... 13
1. Overview

You can use Key Management Service (KMS) to encrypt and protect sensitive data assets. This topic describes a series of operations that help you get started with KMS.

Activate KMS

Manage and use keys

Create a CMK
Create a CMK in the KMS console.

Use a key
Use a key to encrypt resources of cloud services to protect data security.

Manage and use secrets

Create a secret
Create a secret in the KMS console.

Use a secret
Use a secret to store protected data and prevent the leak of sensitive information.
2. Activate KMS

Before you can use Key Management Service (KMS), you must activate it. This topic describes how to activate KMS.

Prerequisites
An Alibaba Cloud account is created and passes real-name verification. For more information, see Account registration and Real-name verification.

Procedure
1. Go to the KMS activation page.
2. Select I agree with Key Management Service Agreement of Service.
3. Click Enable Now.
   After the service is activated, you are charged based on your actual resource usage. For more information, see Billing.
3. Manage and use keys

3.1. Create a CMK

This topic describes how to use Key Management Service (KMS) to create customer master keys (CMKs). CMKs are used to encrypt data.

Procedure

1. Log on to the KMS console.
2. In the top navigation bar, select the region where you want to create a CMK.
3. In the left-side navigation pane, click Keys.
4. Click Create Key.
5. In the Create Key dialog box, configure parameters as prompted.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key Spec</td>
<td>Valid values:</td>
</tr>
<tr>
<td></td>
<td>◦ Symmetric keys:</td>
</tr>
<tr>
<td></td>
<td>▪ Aliyun_AES_256</td>
</tr>
<tr>
<td></td>
<td>▪ Aliyun_SM4</td>
</tr>
<tr>
<td></td>
<td>◦ Asymmetric keys:</td>
</tr>
<tr>
<td></td>
<td>▪ RSA_2048</td>
</tr>
<tr>
<td></td>
<td>▪ EC_P256</td>
</tr>
<tr>
<td></td>
<td>▪ EC_P256K</td>
</tr>
<tr>
<td></td>
<td>▪ EC_SM2</td>
</tr>
<tr>
<td>Purpose</td>
<td>◦ Encrypt/Decrypt: The purpose of the CMK is to encrypt or decrypt data.</td>
</tr>
<tr>
<td></td>
<td>◦ Sign/Verify: The purpose of the CMK is to generate or verify a digital signature.</td>
</tr>
<tr>
<td>Alias Name</td>
<td>The optional identifier of the CMK. For more information, see Use aliases.</td>
</tr>
<tr>
<td>Protection Level</td>
<td>◦ Software: Use a software module to protect the CMK.</td>
</tr>
<tr>
<td></td>
<td>◦ Hsm: Host the CMK in a hardware security module (HSM). Managed HSM uses the HSM as dedicated hardware to safeguard the CMK.</td>
</tr>
<tr>
<td>Description</td>
<td>The description of the CMK.</td>
</tr>
</tbody>
</table>

Note: Aliyun_SM4 and EC_SM2 types are used only in mainland China regions where Managed HSM is available.
## 6. Click Advanced and specify Key Material Source.

- **Alibaba Cloud KMS**: Use KMS to generate key material.
- **External**: Import key material from an external source. For more information about how to import key material, see Import key material.

> Note If you select External, you must also select I understand the implications of using the external key materials key.

## 7. Click OK. After the CMK is created, you can view its detailed information, such as the CMK ID, status, and protection level on the Keys page.

### 3.2. Encrypt resources of cloud services

Key Management Service (KMS) is integrated with cloud services such as Elastic Compute Service (ECS), Object Storage Service (OSS), Container Service for Kubernetes, and ApsaraDB for RDS. You can use KMS to encrypt the resources of these cloud services to ensure data security on the cloud.

**Encrypt ECS resources**

You can use KMS to encrypt ECS resources, such as system disks, data disks, and relevant images and snapshots.

The following example describes how to encrypt data disks when you create an ECS instance. For information about how to encrypt other ECS resources, see Use KMS to protect ECS instance workloads with one click.

1. Log on to the ECS console.
2. In the left-side navigation pane, choose Instances & Images > Instances.
3. In the upper-right corner of the Instances page, click Create Instance.
4. In the Storage section of the Basic Configurations step, perform the following steps to encrypt a data disk:
   i. Click Add Disk.
ii. Configure specifications for the disk.

iii. Select Disk Encryption and select a key from the drop-down list. You can select Default Service CMK to use the default service CMK or select a CMK you created in KMS for encryption.

5. Specify other parameters as prompted. For more information, see Create an instance by using the provided wizard.

Encrypt OSS resources
After files are uploaded to an OSS bucket, KMS automatically encrypts the files.

- Enable encryption during bucket creation
  i. Log on to the OSS console.
  ii. In the upper-right corner of the Overview page, click Create Bucket.
  iii. In the Create Bucket pane, set Encryption Method to KMS.
  iv. Specify Encryption Algorithm.
  v. Specify CMK.

  - alias/acs/oss: OSS uses the service CMK in KMS to generate different keys to encrypt different objects. The objects are automatically decrypted when they are downloaded.
  - CMK ID: OSS uses a specific CMK to generate different keys to encrypt different objects. The objects are automatically decrypted when they are downloaded by the users who have decryption permissions. Before you specify a CMK ID, you must create a common key or an external key in the same region as the bucket in the KMS console. For more information, see Create a CMK.

- Encrypt data in an existing bucket
  i. Log on to the OSS console.
  ii. In the left-side navigation pane, click Buckets.
iii. Click the name of your bucket.

iv. In the left-side navigation pane, choose Basic Settings > Server-side Encryption.

v. Click Configure.

- Set Encryption Method to KMS.
- Specify Encryption Algorithm.
- Specify CMK.
  - alias/acs/oss: OSS uses the service CMK in KMS to generate different keys to encrypt different objects. The objects are automatically decrypted when they are downloaded.
  - CMK ID: OSS uses a specific CMK to generate different keys to encrypt different objects. The objects are automatically decrypted when they are downloaded by the users who have decryption permissions. Before you specify a CMK ID, you must create a common key or an external key in the same region as the bucket in the KMS console. For more information, see Create a CMK.
- Click Save.

Notice: The configurations of the default encryption method for a bucket do not affect the encryption configurations of the existing files in the bucket.

Encrypt Container Service for Kubernetes resources

In clusters of Container Service for Kubernetes Pro, you can use a CMK that you created in KMS to encrypt Kubernetes secrets.

1. Log on to the ACK console.

2. In the left-side navigation pane, choose Clusters > Clusters.

3. In the upper-right corner of the Clusters page, click Create Kubernetes Cluster. In the Select Cluster Template dialog box, find Professional Managed Cluster (Preview) and click Create.

4. On the ACK managed edition tab, find Secret Encryption, select Select Key, and then select a specific CMK ID from the drop-down list.

5. Specify other parameters as prompted. For more information, see Create a professional managed cluster.

Encrypt ApsaraDB for RDS resources

ApsaraDB for RDS supports disk encryption and Transparent Data Encryption (TDE). Disk encryption for ApsaraDB RDS for MySQL is used as an example to describe how to encrypt data.

1. Go to the instance creation page of ApsaraDB for RDS.

2. In the Storage Type field, select Standard SSD or Enhanced SSD (Recommended) and then select Disk Encryption.

3. Select a CMK ID from the Key drop-down list.
4. Specify other parameters as prompted. For more information, see Create an ApsaraDB RDS for MySQL instance.

Encrypt resources of other cloud services
For information about how to encrypt resources of other cloud services, see Alibaba Cloud services that support integration with KMS.

3.3. Sample code for data encryption

After you create a CMK of the AES or SM4 type, you can use code of KMS SDK for Java to encrypt data. This topic provides sample code of KMS SDK for Java to describe how to encrypt data.

Preparations

1. Obtain the dependency declaration of KMS SDK for Java. For information about the required SDK version, see SDK overview. Example:

   ```xml
   <dependency>
     <groupId>com.aliyun</groupId>
     <artifactId>aliyun-java-sdk-core</artifactId>
     <version>4.5.2</version>
   </dependency>
   
   <dependency>
     <groupId>com.aliyun</groupId>
     <artifactId>aliyun-java-sdk-kms</artifactId>
     <version>2.11.1</version>
   </dependency>
   ```

2. Obtain the endpoints to access KMS based on the region where you use KMS. For more information, see Endpoints.

   

   Note  In the example, you can specify the region ID to access the public endpoint of KMS. For more information about how to access the VPC endpoint of KMS, see SDK code samples.

Encrypt data

Use the following code of KMS SDK for Java to encrypt data:
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.profile.DefaultProfile;
import com.google.gson.Gson;
import java.util.*;
import com.aliyuncs.kms.model.v20160120.*;

public class Encrypt {

    public static void main(String[] args) {
        /*
         * 1. Specify the region where your CMK resides.
         * 2. Specify the AccessKey ID and AccessKey secret that are required to access KMS.
         */
        DefaultProfile profile = DefaultProfile.getProfile("cn-hangzhou", "<accessKeyId>", "<accessSecret>");
        IAcsClient client = new DefaultAcsClient(profile);

        EncryptRequest request = new EncryptRequest();

        // Specify the CMK alias or CMK ID that is used to encrypt "Hello world".
        request.setKeyId("alias/Apollo/SalaryEncryptionKey");
        request.setPlaintext("Hello world");

        try {
            EncryptResponse response = client.getAcsResponse(request);
            System.out.println(new Gson().toJson(response));
        } catch (ServerException e) {
            e.printStackTrace();
        } catch (ClientException e) {
            System.out.println("ErrCode:" + e.getErrCode());
            System.out.println("ErrMsg:" + e.getErrMsg());
            System.out.println("RequestId:" + e.getRequestId());
        }
    }
}
4. Manage and use secrets

4.1. Create a secret
This topic describes how to use Key Management Service (KMS) to create a secret. KMS allows you to manage secrets in a centralized manner.

Procedure
1. Log on to the KMS console.
2. In the top navigation bar, select the region where you want to create a secret.
3. In the left-side navigation pane, click Secrets.
4. Click Create Secret.
5. In the Create Secret dialog box, configure parameters as prompted.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Secret Name</td>
<td>The name of the secret.</td>
</tr>
<tr>
<td>Secret Value</td>
<td>The value of the secret. Secrets Manager encrypts the secret value and stores it in the initial version.</td>
</tr>
<tr>
<td>Secret InitVersion</td>
<td>The number of the initial version. Version numbers are unique in each secret object.</td>
</tr>
<tr>
<td>Encryption Master Key</td>
<td>The customer master key (CMK) that is used for encryption. You can select a system-managed key or a custom key as the encryption CMK. The system-managed key is automatically distributed by Secrets Manager. The custom key is the CMK that you created in KMS.</td>
</tr>
<tr>
<td>Secret Description</td>
<td>The description of the secret.</td>
</tr>
</tbody>
</table>

6. Click OK. After the secret is created, you can view detailed information such as the secret name, encryption key, and creation time.

4.2. Sample code for Secrets Manager
This topic provides sample code of KMS SDK for Java to describe how to use a secret created in Secrets Manager.

Preparations
1. Obtain the dependency declaration of KMS SDK for Java. For information about the required SDK version, see SDK overview. Example:
2. Obtain the endpoints to access KMS based on the region where you use KMS. For more information, see [Endpoints](#).

Note: In the example, you can specify the region ID to access the public endpoint of KMS. For more information about how to access the VPC endpoint of KMS, see [SDK code samples](#).

### Use secrets

You can create a secret to store protected data. For more information, see [Overview](#).

```java
package com.aliyun.kms.secretmanager.samples;

import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.kms.model.v20160120.*;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import com.aliyuncs.http.HttpClientConfig;

public class FastUsage {
    
    /*
     * Before you access Secrets Manager, assign a permission policy to your access account in the RAM console. For example, assign the AliyunKMSFullAccess policy, which grants management permissions on KMS, to the account. You can also assign system policies or custom policies that grant required permissions on API operations.
     * */
    public static DefaultAcsClient getKmsClient() {
        /*
         * 1. Specify the region where Secrets Manager resides.
         * 2. Specify the AccessKey ID and AccessKey secret that are required to access KMS.
         * */
```
IClientProfile profile = DefaultProfile.getProfile("cn-hangzhou","$your_access_key","$your_access_secret");

HttpClientConfig clientConfig = HttpClientConfig.getDefault();
profile.setHttpClientConfig(clientConfig);
return new DefaultAcsClient(profile);

public static void CreateSecretSample(String secret_name,String secret_data,String version_id) throws ClientException {
    DefaultAcsClient acsClient = getkmsClient();

    CreateSecretRequest req = new CreateSecretRequest();
    req.setSecretName(secret_name);
    req.setSecretData(secret_data);
    req.setVersionId(version_id);
    req.setSecretDataType("text");
    req.setDescription("my app passwd");
    req.setEncryptionKeyId("");           //You can use a symmetric CMK or leave this parameter empty.
    req.setTags("");

    CreateSecretResponse rsp = acsClient.getAcsResponse(req);
    System.out.printf("CreateSecret arn: %s; secret_name: %s; versionid: %s; requestid: %s \n",rsp.getArn(),rsp.getSecretName(),rsp.getVersionId(),rsp.getRequestId());
}

public static void GetSecretValueSample(String secret_name,String version_stage) throws ClientException {
    DefaultAcsClient acsClient = getkmsClient();

    GetSecretValueRequest req = new GetSecretValueRequest();
    req.setSecretName(secret_name);
    req.setVersionStage(version_stage);

    GetSecretValueResponse  rsp = acsClient.getAcsResponse(req);
    System.out.printf("GetSecretValue  data: %s \n",rsp.getSecretData());
}
public static void PutSecretValueSample(String secret_name,String secret_data,String version_id,String version_stages) throws ClientException {
    DefaultAcsClient acsClient = getkmsClient();

    PutSecretValueRequest req = new PutSecretValueRequest();
    req.setSecretName(secret_name);
    req.setSecretData(secret_data);
    req.setSecretDataType("text");
    req.setVersionId(version_id);
    req.setVersionStages(version_stages); //The secret value of the version that is marked with a specified stage label in the JSON format.

    PutSecretValueResponse rsp = acsClient.getAcsResponse(req);
    System.out.printf("PutSecretValue versionid: %s; now stages: %s \n",rsp.getVersionId(),rsp.getVersionStages());
}

public static void DeleteScretSample() throws ClientException {
    DefaultAcsClient acsClient = getkmsClient();

    DeleteSecretRequest req = new DeleteSecretRequest();
    req.setSecretName("myapp_secret");
    req.setForceDeleteWithoutRecovery("true");

    DeleteSecretResponse rsp = acsClient.getAcsResponse(req);
    System.out.printf("DeleteSecret force delete secret:%s \n",rsp.getSecretName());
}

public static void main(String[] args ){
    try {
       /*
        * Create a secret and specify the initial version and the secret value that you want to encrypt.
        * The initial version is marked with ACSCurrent.
        */
       FastUsage.CreateSecretSample("myapp_secret","mysqpasswdv1","v1");
    }
* Obtain the secret value. If you do not specify a version number or stage label, Secrets Manager returns the secret value of the version marked with ACSCurrent.

```java
FastUsage.GetSecretValueSample("myapp_secret","");
```

/*
 * Store the secret value of a new version in the secret and specify VersionStages for this version. If VersionStages is not specified, the version is marked with ACSCurrent.

```java
FastUsage.PutSecretValueSample("myapp_secret","mysqpasswdv2","v2","
["ACSCurrent","MyUserstage"]");
```

/*
 * Obtain the secret value again. By default, the secret value of the new version is obtained.

```java
FastUsage.GetSecretValueSample("myapp_secret","");
```

FastUsage.PutSecretValueSample("myapp_secret","mysqpasswdv3","v3","");

/*
 * Obtain the secret value. By default, the secret value of the new version is obtained.

```java
FastUsage.GetSecretValueSample("myapp_secret","");
```

/*
 * Obtain the secret value. After VersionId or VersionStages is specified, you can obtain the secret value of an earlier version.

```java
FastUsage.GetSecretValueSample("myapp_secret","MyUserstage");
```

FastUsage.DeleteSecretSample();

```java
} catch (ClientException e) {
    e.printStackTrace();
}
```
For more sample code, see KMS code development sample library.