ALIBABA CLOUD

阿里云

密钥管理服务

快速入门

文档版本：20201112

阿里云

法律声明

阿里云提醒您在阅读或使用本文档之前仔细阅读、充分理解本法律声明各条款的内容。如果您阅读或使用本文档，您的阅读或使用行为将被视为对本声明全部内容的认可。

1. 您应当通过阿里云网站或阿里云提供的其他授权通道下载、获取本文档，且仅能用于自身的合法合规的业务活动。本文档的内容视为阿里云的保密信息，您应当严格遵守保密义务；未经阿里云事先书面同意，您不得向任何第三方披露本手册内容或提供给任何第三方使用。

2. 未经阿里云事先书面许可，任何单位、公司或个人不得擅自摘抄、翻译、复制本文档内容的部分或全部，不得以任何方式或途径进行传播和宣传。

3. 由于产品版本升级、调整或其他原因，本文档内容有可能变更。阿里云保留在没有任何通知或者提示下对本文档的内容进行修改的权利，并在阿里云授权通道中不时发布更新后的用户文档。您应当实时关注用户文档的版本变更并通过阿里云授权渠道下载、获取最新版的用户文档。

4. 本文档仅作为用户使用阿里云产品及服务的参考性指引，阿里云以产品及服务的“现状”、“有缺陷”和“当前功能”的状态提供本文档。阿里云在现有技术的基础上尽最大努力提供相应的介绍及操作指引，但阿里云在此明确声明对本文档内容的准确性、完整性、适用性、可靠性等不作任何明示或暗示的保证。任何单位、公司或个人因为下载、使用或信赖本文档而发生任何差错或经济损失的，阿里云不承担任何法律责任。在任何情况下，阿里云均不对任何间接性、后果性、惩戒性、偶然性、特殊性或刑罚性的损害，包括用户使用或信赖本文档而遭受的利润损失，承担责任（即使阿里云已被告知该等损失的可能性）。

5. 阿里云网站上所有内容，包括但不限于著作、产品、图片、档案、资讯、资料、网站架构、网站画面的安排、网页设计，均由阿里云和/或其关联公司依法拥有其知识产权，包括但不限于商标权、专利权、著作权、商业秘密等。非经阿里云和/或其关联公司书面同意，任何人不得擅自使用、修改、复制、公开传播、改变、散布、发行或公开发表阿里云网站、产品程序或内容。此外，未经阿里云事先书面同意，任何人不得为了任何营销、广告、促销或其他目的使用、公布或复制阿里云的名称（包括但不限于单独为或以组合形式包含“阿里云”、“Aliyun”、“万网”等阿里云和/或其关联公司品牌，上述品牌的附属标志及图案或任何类似公司名称、商号、商标、产品或服务名称、域名、图案标示、标志、标识或通过特定描述使第三方能够识别阿里云和/或其关联公司）。

6. 如若发现本文档存在任何错误，请与阿里云取得直接联系。

通用约定

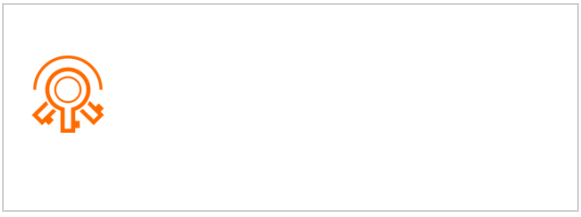
格式	说明	样例
危险	该类警示信息将导致系统重大变更甚至故障，或者导致人身伤害等结果。	**危险** 重置操作将丢失用户配置数据。
警告	该类警示信息可能会导致系统重大变更甚至故障，或者导致人身伤害等结果。	**警告** 重启操作将导致业务中断，恢复业务时间约十分钟。
注意	用于警示信息、补充说明等，是用户必须了解的内容。	**注意** 权重设置为0，该服务器不会再接受新请求。
说明	用于补充说明、最佳实践、窍门等，不是用户必须了解的内容。	**说明** 您也可以通过按Ctrl+A选中全部文件。
>	多级菜单递进。	单击**设置**> **网络**> **设置网络类型**。
粗体	表示按键、菜单、页面名称等UI元素。	在**结果确认**页面，单击**确定**。
Courier字体	命令或代码。	执行 `cd /d C:/window` 命令，进入Windows系统文件夹。
斜体	表示参数、变量。	`bae log list --instanceid` *Instance_ID*
[] 或者 [a|b]	表示可选项，至多选择一个。	`ipconfig [-all|-t]`
{} 或者 {a|b}	表示必选项，至多选择一个。	`switch {active|stand}`

目录

1.入门概述

使用密钥管理服务KMS（Key Management Service），帮助您轻松加密保护敏感的数据资产。本文为您介绍一系列入门操作，方便您快速上手和使用。

开通密钥管理服务

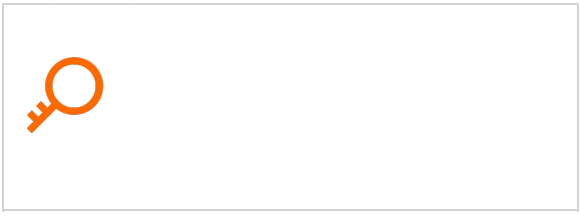


开通密钥管理服务

开通密钥管理服务后才能使用。

- 开通密钥管理服务

管理和使用密钥



创建密钥

在密钥管理服务中创建密钥。

- 创建密钥

使用密钥

使用密钥加密保护云上数据安全。

- 使用KMS加密云服务
- 数据加密代码开发示例

管理和使用凭据

创建凭据

在密钥管理服务中创建凭据。

- 创建凭据

使用凭据

使用凭据存储受保护数据，避免敏感信息泄露。

- 凭据管家代码开发示例

2.开通密钥管理服务

密钥管理服务KMS（Key Management Service）需要开通后才能使用。本文为您介绍KMS的开通方法。

前提条件

请确保您已完成阿里云账号注册和实名认证。

通过密钥管理控制台开通

1. 登录密钥管理服务开通页。
2. 在密钥管理服务开通页，勾选**密钥管理服务协议**。
3. 单击**立即开通**。
   服务开通后会按照实际使用量进行收费，详情请参见计费说明。

通过API开通

您可以调用OpenKmsService接口开通密钥管理服务。

> **说明** 您可以调用DescribeAccountKmsStatus接口查询当前阿里云账号的密钥管理服务状态。

3.管理和使用密钥

3.1. 创建密钥

您可以使用密钥管理服务KMS（Key Management Service）轻松地创建密钥，使用密钥加密自己的数据。

操作步骤

1. 登录密钥管理服务控制台。
2. 在页面左上角的地域下拉列表，选择密钥所在的地域。
3. 在左侧导航栏，单击**用户主密钥**。
4. 单击**创建密钥**。
5. 在弹出的**创建密钥**对话框，根据控制台提示进行配置。

配置项	说明
密钥类型	请选择以下密钥类型。 ○ 对称密钥的类型： ■ Aliyun_AES_256 ■ Aliyun_SM4 ○ 非对称密钥的类型： ■ RSA_2048 ■ EC_P256 ■ EC_P256K ■ EC_SM2 **说明** Aliyun_SM4或EC_SM2的密钥类型，仅在中国内地使用托管密码机的地域支持。
密钥用途	○ Encrypt/Decrypt：数据加密和解密。 ○ Sign/Verify：产生和验证数字签名。
别名	用户主密钥的可选标识，详情请参见别名概述。
保护级别	○ Software：通过软件模块对密钥进行保护。 ○ Hsm：将密钥托管在密码机中，使密钥获得高安全等级的专用硬件的保护。
描述	密钥的说明信息。

配置项	说明
轮转周期	自动轮转的时间周期。取值： ○ 30天。 ○ 90天。 ○ 180天。 ○ 365天。 ○ 不开启：不开启轮转。 ○ 自定义：7~730天。 **说明** 仅密钥类型为Aliyun_AES_256和Aliyun_SM4的对称密钥支持设置轮转周期。

6. 单击**高级选项**，选择**密钥材料来源**。
   - **阿里云KMS**：密钥材料将由KMS生成。
   - **外部**：KMS将不会生成密钥材料，您需要将自己的密钥材料导入KMS，详情请参见导入密钥材料。

     说明 此时需要选中**我了解使用外部密钥材料的方法和意义**复选框。
7. 单击**确认**。密钥创建完成后，您可以查看密钥ID、密钥状态、密钥保护级别等信息。

3.2. 使用KMS加密云服务

密钥管理服务KMS（Key Management Service）已集成云服务器ECS、对象存储OSS、容器服务Kubernetes版ACK、关系型数据库RDS等云服务，您可以使用KMS加密这些云服务，保护云上数据安全。

加密云服务器ECS

您可以使用KMS加密ECS的资源，例如：ECS系统盘、数据盘，以及和它们相关的镜像、快照。

如下以创建ECS实例时加密数据盘为例，为您介绍加密ECS的方法。其他操作方法，请参见使用KMS一键保护ECS工作负载。

1. 登录ECS管理控制台。
2. 在左侧导航栏，单击**实例与镜像 > 实例**。
3. 在**实例列表**页面的右上角，单击**创建实例**。
4. 在**基础配置**页面的**存储**区域，按以下步骤加密数据盘。
   i. 单击**增加一块数据盘**。
   ii. 设置数据盘规格参数。
   iii. 选中**加密**，然后在下拉列表中选择一个密钥。您可以选择KMS服务主密钥（Default Service CMK）或在KMS中提前创建的用户主密钥进行加密。

   
5. 根据控制台提示完成其他参数设置，详情请参见使用向导创建实例。

加密对象存储OSS

当文件上传到对象存储（OSS）的存储空间（Bucket） 后，使用KMS自动对其进行落盘加密存储。

- **在创建Bucket时进行加密**
  i. 登录OSS管理控制台。
  ii. 在**概览**页右侧单击**创建Bucket**。
  iii. 在**创建Bucket**页面的**服务器端加密**区域，选择**KMS**。
  iv. 设置**加密算法**。
  v. 选择**加密密钥**。
    - alias/acs/oss：使用KMS服务主密钥生成不同的密钥来加密不同的Object，并且在Object被下载时自动解密。
    - CMK ID：使用指定的CMK生成不同的密钥来加密不同的Object，具有解密权限的用户下载Object时会自动解密。选择指定的CMK ID前，需在密钥管理服务控制台创建一个与Bucket相同地域的普通密钥或外部密钥，详情请参见创建密钥。

    > **说明** 其他参数设置详情，请参见创建存储空间。

- **对已有Bucket进行加密**
  i. 登录OSS管理控制台。
  ii. 在左侧导航栏，单击**Bucket列表**。
  iii. 单击目标Bucket名称。
  iv. 在左侧导航栏，选择**基础设置 > 服务器端加密**。
  v. 单击**设置**。
    - 设置**服务端加密方式**为**KMS**。
    - 设置**加密算法**。
    - 选择**加密密钥**。
      - alias/acs/oss：使用KMS服务主密钥生成不同的密钥来加密不同的Object，并且在Object被下载时自动解密。
      - CMK ID：使用指定的CMK生成不同的密钥来加密不同的Object，具有解密权限的用户下载Object时会自动解密。选择指定的CMK ID前，需在密钥管理服务控制台创建一个与Bucket相同地域的普通密钥或外部密钥，详情请参见创建密钥。
    - 单击**保存**。

    > **注意** 开启或修改Bucket默认加密方式不会对Bucket内已有文件添加或修改加密方式。

加密容器服务Kubernetes版ACK

在ACK Pro托管集群中，您可以使用在KMS中创建的密钥加密Kubernetes Secret密钥。

1. 登录容器服务管理控制台。
2. 在控制台左侧导航栏中，选择**集群 > 集群**。
3. 单击页面右上角的**创建集群**，在弹出的**选择集群模板**对话框，选择**Pro版集群**，并单击**创建**。
4. 在**ACK托管版**页签找到**Secret落盘加密**，选中**选择KMS密钥**，在下拉列表中选择KMS密钥ID。

5. 根据控制台提示完成其他参数设置，详情请参见创建Kubernetes Pro版集群。

加密关系型数据库RDS

RDS数据加密支持云盘加密和透明数据加密TDE。如下以MySQL引擎的云盘加密为例，为您介绍加密RDS的方法。

1. 登录RDS实例创建页面。
2. 在**存储类型**区域，选择**SSD云盘**或**ESSD云盘**并选中右侧**云盘加密**。
3. 在**密钥**区域下拉列表中选择KMS密钥ID。

4. 根据控制台提示完成其他参数设置，详情请参见创建RDS MySQL实例。

加密更多云服务

更多云服务加密介绍，请参见支持服务端集成加密的云服务。

3.3. 数据加密代码开发示例

创建密钥类型为AES或SM4的用户主密钥后，您可以使用KMS简单易用的SDK代码进行数据加密保护。本文以Java SDK为例为您介绍如何进行数据加密。

准备工作

1. 获取Java SDK的依赖声明，需要获取的版本请参见SDK概览。示例如下：

```
<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. 根据您使用的KMS地域，确认正确的KMS服务接入地址。详情请参见服务地址。

> **说明** 本文示例通过指定地域标识符，快速访问KMS的公网接入地址。访问KMS的VPC地址操作方法，请参见Java SDK示例。

加密数据

使用以下Java SDK代码进行数据加密保护。

```
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. 指定用户主密钥所在地域。
     * 2. 指定访问KMS所需要的凭证AccessKey ID和AccessKey Secret。
     */
    DefaultProfile profile = DefaultProfile.getProfile("cn-hangzhou", "<accessKeyId>", "<accessSecret>");
    IAcsClient client = new DefaultAcsClient(profile);

    EncryptRequest request = new EncryptRequest();

    // 指定用于加密“Hello world”的用户主密钥别名或者用户主密钥ID。
    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());
    }
  }
}
```

更多代码示例，请参见KMS代码开发样例库。

4.管理和使用凭据

4.1. 创建凭据

您可以使用密钥管理服务KMS（Key Management Service）创建凭据，轻松实现对敏感凭据的统一管理。

操作步骤

1. 登录密钥管理服务控制台。
2. 在页面左上角的地域下拉列表，选择凭据所在的地域。
3. 在左侧导航栏单击**凭据列表**。
4. 单击**创建凭据**。
5. 在弹出的**创建凭据**对话框，根据控制台提示进行配置。

配置项	说明
名称	凭据名称。
凭据值	凭据管家将其加密后，存入初始版本中。
初始版本号	初始版本的版本号。凭据对象内版本号唯一。
加密主密钥	您可以选择系统托管密钥（凭据管家自动分配的密钥）或者自选密钥（在KMS上创建的用户主密钥）作为加密主密钥。
描述信息	凭据的说明信息。

6. 单击**确认**。凭据创建完成后，您可以查看凭据名称、加密密钥、创建时间等信息。

4.2. 凭据管家代码开发示例

创建凭据后，您可以使用KMS简单易用的SDK代码使用凭据。本文以Java SDK为例为您介绍如何使用凭据。

准备工作

1. 获取Java SDK的依赖声明，需要获取的版本请参见SDK概览。示例如下：

```
<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>2.12.0</version>
</dependency>
```

2. 根据您使用的KMS地域，确认正确的KMS服务接入地址。详情请参见服务地址。

> **说明** 本文示例通过指定地域标识符，快速访问KMS的公网接入地址。访问KMS的VPC地址操作方法，请参见Java SDK示例。

使用凭据

您可以创建凭据，将受保护数据存入凭据。凭据管家详情，请参见凭据管家概述。

```
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 {
    /*
     * 访问凭据管家前，请在RAM控制台为访问账号添加访问策略（例如：设置密钥管理服务的管理权限AliyunKMSFullAccess）。您也可以添加包含所需的API权限的系统策略或自定义策略。
     **/
  public static DefaultAcsClient getkmsClient() {
    /*
     * 1. 指定凭据管家所在地域。
     * 2. 指定访问KMS所需要的凭证AccessKey ID和AccessKey Secret。
     **/
    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("");       //您可以使用对称密钥类型的用户主密钥，或设置为空。设置为空时将使用凭据管家为用户创建的托管密钥。
    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); //凭据指定状态的参数取值为JSON格式。


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

```
tags());
  }


  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 {
      /*
       * 创建凭据，并指定初始版本VersionId和需被加密的凭据值SecretData。初始版本的状态被系统标记为ACSCurrent。
       * */
      FastUsage.CreateSecretSample("myapp_secret","mysqpasswdv1","v1");
      /*
       * 获取凭据。如果不指定版本号或版本状态，则凭据管家默认返回被标记为ACSCurrent版本的凭据值。
       * */
      FastUsage.GetSecretValueSample("myapp_secret","");


      /*
       * 为凭据存入一个新版本的凭据值，并指定此版本所处状态VersionStages。若不指定，系统默认将ACSCurrent移动至新版本。
       * */
      FastUsage.PutSecretValueSample("myapp_secret","mysqpasswdv2","v2","[\"ACSCurrent\", \"MyUserstage\"]");
      /*
       * 再次获取凭据。默认获取最新版本的凭据值。
       * */
      FastUsage.GetSecretValueSample("myapp_secret","");
```

```
        FastUsage.PutSecretValueSample("myapp_secret","mysqpasswdv3","v3","");
        /*
         * 获取凭据。默认获取最新版本的凭据值。
         **/
        FastUsage.GetSecretValueSample("myapp_secret","");
        /*
         * 获取凭据。指定参数VersionId或VersionStage后，您可以获取之前创建版本的凭据值。
         **/
        FastUsage.GetSecretValueSample("myapp_secret","MyUserstage");


        FastUsage.DeleteScretSample();

      } catch (ClientException e) {
        e.printStackTrace();
      }

    }
}
```

更多代码示例，请参见KMS代码开发样例库。