阿里云
密钥管理服务

SDK参考

文档版本：20200616

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. 阿里云文档中所有内容，包括但不限于图片、架构设计、页面布局、文字描述，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
	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。	 **禁止：** 重置操作将丢失用户配置数据。
	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。
	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。
	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。	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 SDK概览

密钥管理服务支持Java、Python、PHP、.NET开发。

下表列举了各语言SDK的下载地址和开发指南，更多SDK的信息，请访问阿里云开发平台。

Alibaba Cloud SDK	密钥管理服务SDK	说明文档
Alibaba Cloud SDK for Java	Alibaba Cloud KMS SDK for Java	快速开始
Alibaba Cloud SDK for Python	Alibaba Cloud KMS SDK for Python	快速开始
Alibaba Cloud SDK for PHP	Alibaba Cloud KMS SDK for PHP	快速开始
Alibaba Cloud SDK for .NET	Alibaba Cloud KMS SDK for .NET	快速开始

2 SDK示例

密钥管理服务KMS（Key Management Service）支持Java、Python、Go等语言的SDK。本文为您介绍Java SDK示例。

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>2.10.1</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#unique_9/unique_9_Connect_42_section_x3t_t3p_kfb。

> **说明：**
> KMS支持通过公网或者VPC访问，在使用SDK时，需要指定不同的参数。
>
> - 如果仅指定地域的标识符，SDK会默认使用指定地域的KMS公网接入地址。
> - 如果需要访问KMS的VPC接入地址，需要手动为SDK指定接入地址参数。

示例：KmsClient

创建一个调用封装类。

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

import java.util.*;
import java.util.List;

import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpClientConfig;
import com.aliyuncs.http.MethodType;
import com.aliyuncs.http.ProtocolType;

//Current KMS SDK version:2016-01-20
import com.aliyuncs.kms.model.v20160120.CreateKeyRequest;
import com.aliyuncs.kms.model.v20160120.CreateKeyResponse;
import com.aliyuncs.kms.model.v20160120.DecryptRequest;
import com.aliyuncs.kms.model.v20160120.DecryptResponse;
import com.aliyuncs.kms.model.v20160120.DescribeKeyRequest;
import com.aliyuncs.kms.model.v20160120.DescribeKeyResponse;
import com.aliyuncs.kms.model.v20160120.EncryptRequest;
```

```
import com.aliyuncs.kms.model.v20160120.EncryptResponse;
import com.aliyuncs.kms.model.v20160120.GenerateDataKeyRequest;
import com.aliyuncs.kms.model.v20160120.GenerateDataKeyResponse;
import com.aliyuncs.kms.model.v20160120.ListKeysRequest;
import com.aliyuncs.kms.model.v20160120.ListKeysResponse;
import com.aliyuncs.kms.model.v20160120.ListKeysResponse.Key;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;

public class KmsClient
{
   private DefaultAcsClient kmsClient;

   /**
    * 创建KmsClient，只需要指定regionId，SDK自动配置相应地域的公网Endpoint
    */
   public static KmsClient getClientForPublicEndpoint(String regionId, String accessKeyId,
String accessKeySecret) {
       /**
        * Construct an Aliyun Client:
        * Set RegionId, AccessKeyId and AccessKeySecret
        */
       IClientProfile profile = DefaultProfile.getProfile(regionId, accessKeyId, accessKeyS
ecret);
       DefaultAcsClient client = new DefaultAcsClient(profile);
       return new KmsClient(client);
   }

   /**
    * 创建KmsClient，使用自定义Endpoint，通常用于访问VPC Endpoint
    */
   public static KmsClient getClientForVpcEndpoint(String regionId, String accessKeyId,
String accessKeySecret, String endpoint) {
       //添加自定义Endpoint
       DefaultProfile.addEndpoint(regionId, "kms", endpoint);

       IClientProfile profile = DefaultProfile.getProfile(regionId, accessKeyId, accessKeyS
ecret);
       HttpClientConfig clientConfig = HttpClientConfig.getDefault();
       profile.setHttpClientConfig(clientConfig);
       DefaultAcsClient client = new DefaultAcsClient(profile);
       return new KmsClient(client);
   }

   private KmsClient(DefaultAcsClient acsClient) {
       this.kmsClient = acsClient;
   }

   public CreateKeyResponse CreateKey(String keyDesc, String keyUsage) throws
ClientException {
       final CreateKeyRequest ckReq = new CreateKeyRequest();

       ckReq.setProtocol(ProtocolType.HTTPS);
       ckReq.setAcceptFormat(FormatType.JSON);
       ckReq.setMethod(MethodType.POST);
       ckReq.setDescription(keyDesc);
       ckReq.setKeyUsage(keyUsage);

       final CreateKeyResponse response = kmsClient.getAcsResponse(ckReq);
       return response;
   }

   public DescribeKeyResponse DescribeKey(String keyId) throws ClientException {
       final DescribeKeyRequest decKeyReq = new DescribeKeyRequest();
```

```
        decKeyReq.setProtocol(ProtocolType.HTTPS);
        decKeyReq.setAcceptFormat(FormatType.JSON);
        decKeyReq.setMethod(MethodType.POST);
        decKeyReq.setKeyId(keyId);

        final DescribeKeyResponse decKeyRes = kmsClient.getAcsResponse(decKeyReq);
        return decKeyRes;
    }

    public ListKeysResponse ListKey(int pageNumber, int pageSize) throws ClientException
{
        final ListKeysRequest listKeysReq = new ListKeysRequest();

        listKeysReq.setProtocol(ProtocolType.HTTPS);
        listKeysReq.setAcceptFormat(FormatType.JSON);
        listKeysReq.setMethod(MethodType.POST);
        listKeysReq.setPageNumber(pageNumber);
        listKeysReq.setPageSize(pageSize);

        final ListKeysResponse listKeysRes = kmsClient.getAcsResponse(listKeysReq);
        return listKeysRes;
    }

    public GenerateDataKeyResponse GenerateDataKey(String keyId, String keyDesc, int
numOfBytes) throws ClientException {
        final  GenerateDataKeyRequest genDKReq = new GenerateDataKeyRequest();

        genDKReq.setProtocol(ProtocolType.HTTPS);
        genDKReq.setAcceptFormat(FormatType.JSON);
        genDKReq.setMethod(MethodType.POST);

        /**
         * Set parameter according to KMS openAPI document:
         * 1.KeyId
         * 2.KeyDescription
         * 3.NumberOfBytes
         */
        genDKReq.setKeySpec(keyDesc);
        genDKReq.setKeyId(keyId);
        genDKReq.setNumberOfBytes(numOfBytes);

        final GenerateDataKeyResponse genDKRes = kmsClient.getAcsResponse(genDKReq);
        return genDKRes;
    }

    public EncryptResponse Encrypt(String keyId, String plainText) throws ClientException {
        final EncryptRequest encReq = new EncryptRequest();

        encReq.setProtocol(ProtocolType.HTTPS);
        encReq.setAcceptFormat(FormatType.JSON);
        encReq.setMethod(MethodType.POST);
        encReq.setKeyId(keyId);
        encReq.setPlaintext(plainText);
        final EncryptResponse encResponse = kmsClient.getAcsResponse(encReq);
        return encResponse;
    }


    public DecryptResponse Decrypt(String cipherBlob) throws ClientException {
        final DecryptRequest decReq = new DecryptRequest();

        decReq.setProtocol(ProtocolType.HTTPS);
        decReq.setAcceptFormat(FormatType.JSON);
```

```
        decReq.setMethod(MethodType.POST);
        decReq.setCiphertextBlob(cipherBlob);
        final DecryptResponse decResponse = kmsClient.getAcsResponse(decReq);
        return decResponse;
    }
}
```

示例：KmsSample

通过KmsClient调用KMS的接口，实现对密钥的枚举、加解密等操作。注意：

- 本示例假定您在杭州地域至少有一个KMS主密钥。
- KmsClient.getClientForPublicEndpoint方法用于初始化KmsClient访问KMS公网接入地址。
- KmsClient.getClientForVpcEndpoint方法用于初始化KmsClient访问KMS的VPC接入地址。

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

import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
import com.google.gson.Gson;
import java.util.*;
import com.aliyuncs.kms.model.v20160120.*;
import com.aliyuncs.kms.model.v20160120.ListKeysResponse.Key;

public class KmsSample {

 public static void main(String[] args) {
        String accessKeyId = System.getenv("ACCESS_KEY_ID");
        String accessKeySecret = System.getenv("ACCESS_KEY_SECRET");

        KmsClient kmsClient = KmsClient.getClientForPublicEndpoint("cn-hangzhou",
accessKeyId, accessKeySecret);
        //KmsClient kmsClient = KmsClient.getClientForVpcEndpoint("cn-hangzhou-vpc",
accessKeyId, accessKeySecret, "kms-vpc.cn-hangzhou.aliyuncs.com");
        String keyId = null;
        String plainText = "hello world";
        String cipherBlob = null;

    /*List all MasterKeys in your account*/
        try {
            final ListKeysResponse listKeysRes = kmsClient.ListKey(1, 100);

            /**
             * Parse response and do more further
             */
            System.out.println("TotalCount: " + listKeysRes.getTotalCount());
            System.out.println("PageNumber: " + listKeysRes.getPageNumber());
            System.out.println("PageSize: " + listKeysRes.getPageSize());

            List<Key> keys = listKeysRes.getKeys();
            Iterator<Key> iterator = keys.iterator();

            while (iterator.hasNext()) {
                keyId = iterator.next().getKeyId();
                System.out.println("KeyId: " + keyId);
            }

            System.out.println("List All MasterKeys success!\n");
        } catch (ClientException eResponse) {
            System.out.println("Failed.");
```

```
            System.out.println("Error code: " + eResponse.getErrCode());
            System.out.println("Error message: " + eResponse.getErrMsg());
        }


        /*Describe the Key */
        try {
            final DescribeKeyResponse decKeyRes = kmsClient.DescribeKey(keyId);

            /**
             * Parse response and do more further
             */
            System.out.println("DescribeKey Response: ");
            DescribeKeyResponse.KeyMetadata meta = decKeyRes.getKeyMetadata();

            System.out.println("KeyId: " + meta.getKeyId());
            System.out.println("Description: " + meta.getDescription());
            System.out.println("KeyState: " + meta.getKeyState());
            System.out.println("KeyUsage: " + meta.getKeyUsage());

            System.out.println("==========================================");
            System.out.println("Describe the MasterKey success!");
            System.out.println("==========================================\n");
        } catch (ClientException eResponse) {
            System.out.println("Failed.");
            System.out.println("Error code: " + eResponse.getErrCode());
            System.out.println("Error message: " + eResponse.getErrMsg());
        }

        /*Generate DataKey*/
        /**
         * Request and got response
         */
        try {
            final GenerateDataKeyResponse genDKResponse = kmsClient.GenerateDataKey(
keyId, "AES_256", 64);

            /**
             * Parse response and do more further
             */
            System.out.println("CiphertextBlob: " + genDKResponse.getCiphertextBlob());
            System.out.println("KeyId: " + genDKResponse.getKeyId());
            System.out.println("Plaintext: " + genDKResponse.getPlaintext());

            System.out.println("==========================================");
            System.out.println("Generate DataKey success!");
            System.out.println("==========================================\n");
        } catch (ClientException eResponse) {
            System.out.println("Failed.");
            System.out.println("Error code: " + eResponse.getErrCode());
            System.out.println("Error message: " + eResponse.getErrMsg());
        }

        /**
         * Encrypt the plain text and got a cipher one
         */
        try {
            EncryptResponse encResponse = kmsClient.Encrypt(keyId, plainText);

            cipherBlob = encResponse.getCiphertextBlob();
            System.out.println("CiphertextBlob: " + cipherBlob);
            System.out.println("KeyId: " + encResponse.getKeyId());

            System.out.println("==========================================");
```

```
            System.out.println("Encrypt the plain text success!");
            System.out.println("========================================\n");
        } catch (ClientException eResponse) {
            System.out.println("Failed.");
            System.out.println("Error code: " + eResponse.getErrCode());
            System.out.println("Error message: " + eResponse.getErrMsg());
        }

        /**
         * Decrypt the cipher text and verify result with original plain text.
         */
        try {
            DecryptResponse decResponse = kmsClient.Decrypt(cipherBlob);

            System.out.println("Plaintext: " + decResponse.getPlaintext());
            String verifyPlainText = decResponse.getPlaintext();
            int isMatch = verifyPlainText.compareTo(plainText);
            System.out.println("KeyId: " + decResponse.getKeyId());
            System.out.println("========================================");
            System.out.printf("Decrypt the cipher text success, result " + (isMatch == 0 ? "match
" : "mismatch" + "\n"));
            System.out.println("========================================\n");
        } catch (ClientException eResponse) {
            System.out.println("Failed.");
            System.out.println("Error code: " + eResponse.getErrCode());
            System.out.println("Error message: " + eResponse.getErrMsg());
        }
    }

}
```

更多示例

KMS通过开源社区，提供了Java、Python、Go等语言SDK示例，请参见KMS SDK示例。