数据加密服务

API 参考

文档版本 38

发布日期 2024-12-19

华为技术有限公司

商标声明


和其他华为商标均为华为技术有限公司的商标。
本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受华为公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为公司对本文档内容不做任何明示或暗示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

安全声明

漏洞处理流程

华为公司对产品漏洞管理的规定以“漏洞处理流程”为准，该流程的详细内容请参见如下网址：
https://www.huawei.com/cn/psirt/vul-response-process
如企业客户须获取漏洞信息，请参见如下网址：
https://securitybulletin.huawei.com/enterprise/cn/security-advisory

目 录

1 使用前必读

概述

欢迎使用数据加密服务（Data Encryption Workshop，DEW）。数据加密服务是一个综合的云上数据加密服务。它可以提供专属加密、密钥管理、凭据管理、密钥对管理等服务，安全可靠地为用户解决了数据安全、密钥安全、密钥管理复杂等问题。其密钥由硬件安全模块（Hardware Security Module，HSM）保护，并与多个华为云服务集成。您也可以借此服务开发自己的加密应用。

在调用数据加密服务API之前，请确保已经充分了解数据加密服务相关概念，详细信息请参见**产品介绍**。

须知

对接KMS时，必须有重试，包括不限于504、502、500、429等错误码，且推荐重试3~5次。针对502和504错误码，推荐超时时间5~8秒。不建议配置过长超时时间，否则会造成客户侧无法响应。

调用说明

数据加密服务提供了REST（Representational State Transfer）风格API，支持您通过HTTPS请求调用，调用方法请参见**如何调用API**。

终端节点

终端节点（Endpoint）即调用API的**请求地址**，不同服务不同区域的终端节点不同，您可以从**地区和终端节点**中查询服务的终端节点。

约束与限制

您能创建的密钥的数量与配额有关系，具体请参见**服务配额**。

密钥管理服务对单用户每秒可以请求的API操作数量设置为20，即密钥管理服务的TPS性能指标为20TPS。

基本概念

- 账号

  用户注册时的账号，账号对其所拥有的资源及云服务具有完全的访问权限，可以重置用户密码、分配用户权限等。由于账号是付费主体，为了确保账号安全，建议您不要直接使用账号进行日常管理工作，而是创建用户来进行日常管理工作。

- 用户

  由账号在IAM中创建的用户，是云服务的使用人员，具有身份凭证（密码和访问密钥）。

  在**我的凭证**下，您可以查看账号ID和用户ID。通常在调用API的鉴权过程中，您需要用到账号、用户和密码等信息。

- 区域（Region）

  从地理位置和网络时延维度划分，同一个Region内共享弹性计算、块存储、对象存储、VPC网络、弹性公网IP、镜像等公共服务。Region分为通用Region和专属Region，通用Region指面向公共租户提供通用云服务的Region；专属Region指只承载同一类业务或只面向特定租户提供业务服务的专用Region。

  详情请参见**区域和可用区**。

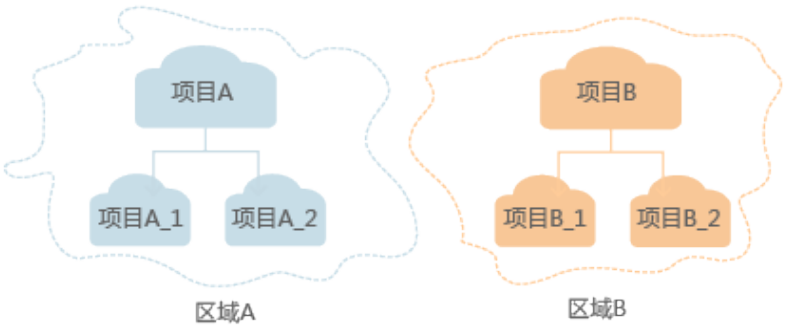
- 可用区（AZ，Availability Zone）

  一个AZ是一个或多个物理数据中心的集合，有独立的风火水电，AZ内逻辑上再将计算、网络、存储等资源划分成多个集群。一个Region中的多个AZ间通过高速光纤相连，以满足用户跨AZ构建高可用性系统的需求。

- 项目

  区域默认对应一个项目，这个项目由系统预置，用来隔离物理区域间的资源（计算资源、存储资源和网络资源），以默认项目为单位进行授权，用户可以访问您账号中该区域的所有资源。如果您希望进行更加精细的权限控制，可以在区域默认的项目中创建子项目，并在子项目中创建资源，然后以子项目为单位进行授权，使得用户仅能访问特定子项目中资源，使得资源的权限控制更加精确。

  图 1-1 项目隔离模型

- 企业项目

  企业项目是项目的升级版，针对企业不同项目间资源的分组和管理，是逻辑隔离。企业项目中可以包含多个区域的资源，且项目中的资源可以迁入迁出。

  关于企业项目ID的获取及企业项目特性的详细信息，请参见**企业管理应用场景**。

API 版本选择建议

SSH密钥对提供了V3、V2.1和V2两个版本，建议您使用V3版本，能够更好的满足您的需求。

2 如何调用 API

2.1 构造请求

本节介绍如何构造REST API的请求，并以调用IAM服务的**管理员创建IAM用户**说明如何调用API，该API获取用户的Token，Token可以用于调用其他API时鉴权。

您还可以通过这个视频教程了解如何构造请求调用API：**https://bbs.huaweicloud.com/videos/102987**。

请求 URI

请求URI由如下部分组成。

{URI-scheme} :// {Endpoint} / {resource-path} ? {query-string}

尽管请求URI包含在请求消息头中，但大多数语言或框架都要求您从请求消息中单独传递它，所以在此单独强调。

- **URI-scheme**：

  表示用于传输请求的协议，当前所有API均采用**HTTPS**协议。

- **Endpoint**：

  指定承载REST服务端点的服务器域名或IP，不同服务不同区域的Endpoint不同，您可以从**地区和终端节点**获取。

  例如IAM服务在“华北-北京一”区域的Endpoint为“iam.cn-north-1.myhuaweicloud.com”。

- **resource-path**：

  资源路径，也即API访问路径。从具体API的URI模块获取，例如“获取用户Token”API的resource-path为“/v3/auth/tokens”。

- **query-string**：

  查询参数，是可选部分，并不是每个API都有查询参数。查询参数前面需要带一个“？”，形式为“参数名=参数取值”，例如“limit=10”，表示查询不超过10条数据。

例如您需要创建IAM用户，由于IAM为全局服务，则使用任一区域的Endpoint（比如“华北-北京四”区域的Endpoint：“iam.cn-north-4.myhuaweicloud.com”），并

在**管理员创建IAM用户**的URI部分找到resource-path（/v3.0/OS-USER/users），拼接起来如下所示。

```
https://iam.cn-north-4.myhuaweicloud.com/v3.0/OS-USER/users
```

图 2-1 URI 示意图

说明

为查看方便，在每个具体API的URI部分，只给出resource-path部分，并将请求方法写在一起。这是因为URI-scheme都是HTTPS，同一个服务的Endpoint在同一个区域也相同，所以简洁起见将这两部分省略。

请求方法

HTTP请求方法（也称为操作或动词），它告诉服务正在请求什么类型的操作。

- **GET**：请求服务器返回指定资源。
- **PUT**：请求服务器更新指定资源。
- **POST**：请求服务器新增资源或执行特殊操作。
- **DELETE**：请求服务器删除指定资源，如删除对象等。
- **HEAD**：请求服务器资源头部。
- **PATCH**：请求服务器更新资源的部分内容。当资源不存在的时候，PATCH可能会去创建一个新的资源。

在**管理员创建IAM用户**的URI部分，您可以看到其请求方法为“POST”，则其请求为：

```
POST https://iam.cn-north-4.myhuaweicloud.com/v3.0/OS-USER/users
```

请求消息头

附加请求头字段，如指定的URI和HTTP方法所要求的字段。例如定义消息体类型的请求头“Content-Type”，请求鉴权信息等。

如下公共消息头需要添加到请求中。

- **Content-Type**：消息体的类型（格式），必选，默认取值为“application/json”，有其他取值时会在具体接口中专门说明。
- **Authorization**：签名认证信息，可选，当使用AK/SK方式认证时，使用SDK对请求进行签名的过程中会自动填充该字段。AK/SK认证的详细说明请参见AK/SK认证。
- **X-Sdk-Date**：请求发送的时间，可选，当使用AK/SK方式认证时，使用SDK对请求进行签名的过程中会自动填充该字段。AK/SK认证的详细说明请参见AK/SK认证。

- **X-Auth-Token**：用户Token，可选，当使用Token方式认证时，必须填充该字段。用户Token也就是调用**获取用户Token**接口的响应值，该接口是唯一不需要认证的接口。
- **X-Project-ID**：子项目ID，可选，在多项目场景中使用。如果云服务资源创建在子项目中，AK/SK认证方式下，操作该资源的接口调用需要在请求消息头中携带X-Project-ID。
- **X-Domain-ID**：账号ID，可选。AK/SK认证方式下，全局服务的接口调用时，需在请求消息头中携带X-Domain-ID。

对于**管理员创建IAM用户**接口，使用AK/SK方式认证时，添加消息头后的请求如下所示。

```
POST https://iam.cn-north-4.myhuaweicloud.com/v3.0/OS-USER/users
Content-Type: application/json
X-Sdk-Date: 20240416T095341Z
Authorization: SDK-HMAC-SHA256 Access=**********, SignedHeaders=content-type;host;x-sdk-date,
Signature=**********
```

说明

API同时支持使用AK/SK认证，AK/SK认证是使用SDK对请求进行签名，签名过程会自动往请求中添加Authorization（签名认证信息）和X-Sdk-Date（请求发送的时间）请求头。

AK/SK认证的详细说明请参见**AK/SK认证**。

请求消息体

请求消息体通常以结构化格式发出，与请求消息头中Content-type对应，传递除请求消息头之外的内容。若请求消息体中参数支持中文，则中文字符必须为UTF-8编码。

每个接口的请求消息体内容不同，也并不是每个接口都需要有请求消息体（或者说消息体为空），GET、DELETE操作类型的接口就不需要消息体，消息体具体内容需要根据具体接口而定。

对于**管理员创建IAM用户**接口，您可以从接口的请求部分看到所需的请求参数及参数说明。将消息体加入后的请求如下所示，加粗的斜体字段需要根据实际值填写。

- **accountid**为IAM用户所属的账号ID。
- **username**为要创建的IAM用户名。
- **email**为IAM用户的邮箱。
- **************为IAM用户的登录密码。

```
POST https://iam.cn-north-4.myhuaweicloud.com/v3.0/OS-USER/users
Content-Type: application/json
 X-Sdk-Date: 20240416T095341Z
 Authorization: SDK-HMAC-SHA256 Access=**********, SignedHeaders=content-type;host;x-sdk-date,
Signature=**********

{
    "user": {
        "domain_id": "accountid",
        "name": "username",
        "password": "**********",
        "email": "email",
        "description": "IAM User Description"
    }
}
```

到这里为止这个请求需要的内容就具备齐全了，您可以使用**curl**、**Postman**或直接编写代码等方式发送请求调用API。

2.2 认证鉴权

调用接口有如下两种认证方式，您可以选择其中一种进行认证鉴权。

- Token认证：通过Token认证调用请求。
- AK/SK认证：通过AK（Access Key ID）/SK（Secret Access Key）加密调用请求。推荐使用AK/SK认证，其安全性比Token认证要高。

Token 认证

说明

- Token的有效期为24小时，需要使用一个Token鉴权时，可以先缓存起来，避免频繁调用。
- 使用Token前请确保Token离过期有足够的时间，防止调用API的过程中Token过期导致调用API失败。

Token在计算机系统中代表令牌（临时）的意思，拥有Token就代表拥有某种权限。Token认证就是在调用API的时候将Token加到请求消息头，从而通过身份认证，获得操作API的权限。

Token可通过调用**获取用户Token**接口获取，调用本服务API需要project级别的Token，即调用获取用户Token接口时，请求body中auth.scope的取值需要选择project，如下所示。

```
{
    "auth": {
        "identity": {
            "methods": [
                "password"
            ],
            "password": {
                "user": {
                    "name": "username",
                    "password": "********",
                    "domain": {
                        "name": "domainname"
                    }
                }
            }
        },
        "scope": {
            "project": {
                "name": "xxxxxxxx"
            }
        }
    }
}
```

获取Token后，再调用其他接口时，您需要在请求消息头中添加“X-Auth-Token”，其值即为Token。例如Token值为“ABCDEFJ....”，则调用接口时将“X-Auth-Token: ABCDEFJ....”加到请求消息头即可，如下所示。

```
POST https://iam.cn-north-4.myhuaweicloud.com/v3.0/OS-USER/users
Content-Type: application/json
X-Auth-Token: ABCDEFJ....
```

您还可以通过这个视频教程了解如何使用Token认证：**https://bbs.huaweicloud.com/videos/101333**。

AK/SK 认证

说明

- AK/SK签名认证方式仅支持消息体大小12MB以内，12MB以上的请求请使用Token认证。
- AK/SK既可以使用永久访问密钥中的AK/SK，也可以使用临时访问密钥中的AK/SK，但使用临时访问密钥的AK/SK时需要额外携带“X-Security-Token”字段，字段值为临时访问密钥的security_token。

AK/SK认证就是使用AK/SK对请求进行签名，在请求时将签名信息添加到消息头，从而通过身份认证。

- AK(Access Key ID)：访问密钥ID。与私有访问密钥关联的唯一标识符；访问密钥ID和私有访问密钥一起使用，对请求进行加密签名。
- SK(Secret Access Key)：与访问密钥ID结合使用的密钥，对请求进行加密签名，可标识发送方，并防止请求被修改。

使用AK/SK认证时，您可以基于签名算法使用AK/SK对请求进行签名，也可以使用专门的签名SDK对请求进行签名。详细的签名方法和SDK使用方法请参见**API签名指南**。

须知

签名SDK只提供签名功能，与服务提供的SDK不同，使用时请注意。

您也可以通过这个视频教程了解AK/SK认证的使用：**https://bbs.huaweicloud.com/videos/100697** 。

2.3 返回结果

请求发送以后，您会收到响应，包含状态码、响应消息头和消息体。

状态码

状态码是一组从1xx到5xx的数字代码，状态码表示了请求响应的状态，完整的状态码列表请参见**状态码**。

对于**管理员创建IAM用户**接口，如果调用后返回状态码为“201”，则表示请求成功。

响应消息头

对应请求消息头，响应同样也有消息头，如“Content-type”。

对于**管理员创建IAM用户**接口，返回如**图2-2**所示的消息头。

图 2-2 获取用户 Token 响应消息头


```
"X-Frame-Options": "SAMEORIGIN",
"X-IAM-ETag-id": "2562365939-d8f6f12921974cb097338ac11fceac8a",
"Transfer-Encoding": "chunked",
"Strict-Transport-Security": "max-age=31536000; includeSubdomains;",
"Server": "api-gateway",
"X-Request-Id": "af2953f2bcc67a42325a69a19e6c32a2",
"X-Content-Type-Options": "nosniff",
"Connection": "keep-alive",
"X-Download-Options": "noopen",
"X-XSS-Protection": "1; mode=block;",
"X-IAM-Trace-Id": "token_[redacted]_null_af2953f2bcc67a42325a69a19e6c32a2",
"Date": "Tue, 21 May 2024 09:03:40 GMT",
"Content-Type": "application/json; charset=utf8"
```


响应消息体

响应消息体通常以结构化格式返回，与响应消息头中Content-type对应，传递除响应消息头之外的内容。

对于**管理员创建IAM用户**接口，返回如下消息体。为篇幅起见，这里只展示部分内容。

```
{
    "user": {
        "id": "c131886aec...",
        "name": "IAMUser",
        "description": "IAM User Description",
        "areacode": "",
        "phone": "",
        "email": "***@***.com",
        "status": null,
        "enabled": true,
        "pwd_status": false,
        "access_mode": "default",
        "is_domain_owner": false,
        "xuser_id": "",
        "xuser_type": "",
        "password_expires_at": null,
        "create_time": "2024-05-21T09:03:41.000000",
        "domain_id": "d78cbac1.....",
        "xdomain_id": "30086000.....",
        "xdomain_type": "",
        "default_project_id": null
    }
}
```

当接口调用出错时，会返回错误码及错误信息说明，错误响应的Body体格式如下所示。

```
{
    "error_msg": "Request body is invalid.",
    "error_code": "IAM.0011"
}
```

其中，error_code表示错误码，error_msg表示错误描述信息。

3 API 概览

数据加密服务提供的接口类型，以及接口对应的性能参数，您可以完整地使用数据加密服务的所有功能。

性能参数类型包括共享流控以及基础流控，其中：

- 共享流控：本类型分类接口共同使用流量。
- 基础流控：仅由该接口的流量使用。

说明

未单独标识基础流控接口，均为共享流控，例如创建凭据版本接口。

类型	说明
管理加密密钥	加密密钥管理接口，包括密钥的创建，查询，修改，删除等接口。
凭据管理服务	凭据管理接口，包括凭据的创建，查询，修改，删除等接口。
管理SSH密钥对	SSH密钥对管理接口（最新版本），包括SSH密钥对的创建，查询，修改，删除等接口。
历史全局	SSH密钥对管理接口（V2.1和V2版本），包括SSH密钥对的创建，查询，修改，删除等接口。

管理加密密钥

类型	名称	说明	性能参数
查询全局版本信息	查询版本信息列表	查询API版本信息列表。	-
	查询指定版本信息	查指定API版本信息。	

类型	名称	说明	性能参数
生命周期管理	创建密钥	创建用户主密钥，用户主密钥可以为对称密钥或非对称密钥。	单用户20次/秒 全局100次/秒
	启用密钥	启用密钥，密钥启用后才可以使用。	
	禁用密钥	禁用密钥，密钥禁用后不可以使用。	
	计划删除密钥	计划删除一个指定密钥，可设置7天～1096天内删除密钥。密钥将被彻底删除后，使用该密钥加密的数据将无法解密。	
	取消计划删除密钥	取消计划删除密钥，取消后，用户可以正常使用该密钥。	
	修改密钥别名	修改用户主密钥别名。	
	修改密钥描述	修改用户主密钥描述信息。	
数据密钥管理	创建随机数	生成8~8192bit范围内的随机数。	单用户800次/秒 全局1000次/秒
	创建数据密钥	创建数据密钥，返回结果包含明文和密文。	
	创建不含明文数据密钥	创建数据密钥，返回结果只包含密文。	
	加密数据密钥	使用指定的用户主密钥加密数据密钥。	
	解密数据密钥	使用指定的用户主密钥解密数据密钥。	
导入密钥管理	获取密钥导入参数	获取导入密钥的必要参数，包括密钥导入令牌和密钥加密公钥。	单用户20次/秒 全局100次/秒
	导入密钥材料	导入指定密钥的密钥材料。	
	删除密钥材料	删除指定密钥的密钥材料。	
授权管理	创建授权	被授权用户可以对授权密钥进行操作。	单用户20次/秒 全局100次/秒
	撤销授权	授权用户撤销被授权用户操作密钥的权限。	
	退役授权	被授权用户不再具有授权密钥的操作权。	
	查询授权列表	查询密钥的授权列表。	
	查询可退役授权列表	查询用户可以退役的授权列表。	

类型	名称	说明	性能参数
小数据加解密	加密数据	使用指定的用户主密钥加密数据。	单用户20次/秒 全局100次/秒
	解密数据	解密数据。	
签名验签	签名数据	使用非对称密钥的私钥对消息或消息摘要进行数字签名。	单用户300次/秒 全局500次/秒
	验证签名	使用非对称密钥的公钥对消息或消息摘要进行签名验证。	
轮换管理	开启密钥轮换	开启用户主密钥轮换，默认主密钥及外部导入密钥不支持轮换操作。	单用户20次/秒 全局100次/秒
	关闭密钥轮换	关闭用户主密钥轮换。	
	修改密钥轮换周期	修改用户主密钥轮换周期。	
	查询密钥轮换状态	查询用户主密钥轮换状态。	
标签管理	查询密钥实例	通过标签过滤，查询指定用户主密钥的详细信息。	单用户20次/秒 全局100次/秒
	查询密钥标签	查询指定密钥的标签信息。	
	查询项目标签	查询用户在指定项目下的所有标签集合。	
	批量添加删除密钥标签	批量添加或删除密钥标签。	
	添加密钥标签	添加密钥标签。	
	删除密钥标签	删除密钥标签。	
查询接口	查询密钥列表	查询用户所有密钥列表。	单用户160次/秒 全局200次/秒
	查询密钥信息	查询指定密钥的详细信息。	
	查询实例数	获取用户已经创建的用户主密钥数量，不包含默认主密钥。	单用户80次/秒 全局200次/秒
	查询配额	查询用户可以创建的用户主密钥配额总数及当前使用量信息，不包含默认主密钥。	
专属密钥库管理	创建专属密钥库	创建租户专属密钥库，专属密钥库使用DHSM实例作为密钥的存储。	单用户20次/秒 全局100次/秒
	查询专属密钥库列表	查询租户专属密钥库列表。	

类型	名称	说明	性能参数
	获取专属密钥库	获取租户专属密钥库。	
	删除专属密钥库	删除租户专属密钥库。	
	启用专属密钥库	启用租户专属密钥库。	
	禁用专属密钥库	禁用租户专属密钥库。	

凭据管理服务

类型	名称	说明	配额
生命周期管理	创建凭据	创建新的凭据，并将凭据值存入凭据的初始版本。	单用户300次/分钟 全局4800次/分钟
	查询凭据列表	查询当前用户在本项目下创建的所有凭据。	单用户100次/秒 全局 200次/秒
	查询凭据	查询指定凭据的信息。	单用户1200次/分钟 全局 4800次/分钟
	更新凭据	更新指定凭据的元数据信息。	单用户300次/分钟 全局4800次/分钟
	立即删除凭据	立即删除指定的凭据，且无法恢复。	
	恢复凭据对象	通过上传凭据备份文件，恢复凭据对象。	
	下载凭据备份	下载指定凭据的备份文件。	
	创建凭据的定时删除任务	指定延迟删除时间，创建删除凭据的定时任务，可设置7~30天的的延迟删除时间。	
	取消凭据的定时删除任务	取消凭据的定时删除任务，凭据对象恢复可使用状态。	

类型	名称	说明	配额
	轮转凭据	立即执行轮转凭据。在指定的凭据中，创建一个新的凭据版本，用于加密存储后台随机产生的凭据值。同时将新创建的凭据版本标记为SYSCURRENT状态。	
凭据版本管理	创建凭据版本	在指定的凭据中，创建一个新的凭据版本，用于加密保管新的凭据值。默认情况下，新创建的凭据版本被标记为SYSCURRENT状态，而SYSCURRENT标记的前一个凭据版本被标记为SYSPREVIOUS状态。您可以通过指定VersionStage参数来覆盖默认行为。	基础流控： 单用户80次/秒 全局 200次/秒 应用80次/秒 IP 80次/秒
	查询凭据的版本列表	查询指定凭据下的版本列表信息。	单用户300次/分钟 全局4800次/分钟
	更新凭据版本	当前支持更新指定凭据版本的有效期，只能更新ENABLED状态的凭据。在关联订阅的事件包含“版本过期”基础事件类型时，每次更新版本有效期后仅会触发一次事件通知。	
	查询凭据的版本与凭据值	查询指定凭据版本的信息和版本中的明文凭据值，只能查询ENABLED状态的凭据。 通过/v1/{project_id}/secrets/{secret_name}/versions/latest（即将当前接口URL中的{version_id}赋值为latest）可访问凭据最新版本的凭据值。	基础流控： 单用户160次/秒 全局 200次/秒 应用160次/秒 IP 160次/秒
凭据版本状态管理	更新凭据的版本状态	更新凭据的版本状态。	单用户300次/分钟 全局4800次/分钟
	查询凭据的版本状态	查询指定凭据版本状态标记的版本信息。	
	删除凭据的版本状态	删除指定的凭据版本状态。	
凭据标签管理	查询凭据实例	查询凭据实例。通过标签过滤，筛选用户凭据，返回凭据列表。	单用户300次/分钟 全局4800次/分钟

类型	名称	说明	配额
	批量添加或删除凭据标签	批量添加或删除凭据标签。	
	查询凭据标签	查询凭据标签。	
	添加凭据标签	添加凭据标签。	
	删除凭据标签	删除凭据标签。	
	查询项目标签	查询用户在指定项目下的所有凭据标签集合。	
事件管理	创建事件	创建事件，事件可配置在一个或多个凭据对象上。当事件为启用状态且包含的基础事件类型在凭据对象上触发时，云服务会将对应的事件通知发送至事件指定的通知主题上。	单用户300次/分钟 全局4800次/分钟
	查询事件	查询指定事件的信息。	
	查询事件列表	查询当前用户在本项目下创建的所有事件。	
	更新事件	更新指定事件的元数据信息。支持更新的元数据包含事件启用状态、基础类型列表、通知主题。	
	立即删除事件	立即删除指定的事件，且无法恢复。如事件存在凭据引用，则无法删除，请先解除关联。	
	查询已触发的事件通知记录	查询三个月内所有已触发的事件通知记录。	

管理 SSH 密钥对

类型	名称	说明	配额
密钥对管理	创建和导入SSH密钥对	创建和导入SSH密钥对。	单用户300次/分钟 全局4800次/分钟
	清除私钥	清除SSH密钥对私钥。	

类型	名称	说明	配额
	查询SSH密钥对列表	查询SSH密钥对列表。	基础流控： 单用户160次/秒 全局 200次/秒 应用160次/秒 IP 160次/秒
	查询SSH密钥对详细信息	查询SSH密钥对详细信息。	
	删除SSH密钥对	删除SSH密钥对。	单用户300次/分钟 全局4800次/分钟
	更新SSH密钥对描述	更新SSH密钥对描述。	
	导入私钥	导入私钥到指定密钥对。	
	导出私钥	导出指定密钥对的私钥。	
密钥对任务管理	绑定SSH密钥对	给指定的虚拟机绑定（替换或重置，替换需提供虚拟机已配置的SSH密钥对私钥；重置不需要提供虚拟机的SSH密钥对私钥）新的SSH密钥对。	单用户300次/分钟 全局4800次/分钟
	解绑SSH密钥对	给指定的虚拟机解除绑定SSH密钥对并恢复SSH密码登录。	
	批量绑定SSH密钥对	给指定的虚拟机批量绑定新的SSH密钥对。	基础流控： 单用户10次/分钟 全局 20次/分钟 应用10次/分钟 IP 10次/分钟
	查询任务信息	根据SSH密钥对接口返回的task_id，查询SSH密钥对当前任务的执行状态。	单用户300次/分钟 全局4800次/分钟
	查询正在处理的任务信息	查询正在处理的任务信息。	
	查询失败的任务信息	查询绑定、解绑等操作失败的任务信息。	
	删除所有失败的任务	删除操作失败的任务信息。	
	删除失败的任务	删除失败的任务。	

历史全局

全局分类	说明
管理SSH密钥对(V2.1)	查询SSH密钥对列表。
	查询SSH密钥对详细信息。
	创建SSH密钥对和导入SSH密钥对，同时可选择对私钥进行托管。
	根据SSH密钥对的名称，删除指定SSH密钥对。
	根据SSH密钥对的名称，修改指定SSH密钥对的描述信息。
管理SSH密钥对(V2.0)	查询SSH密钥对信息列表。
	根据SSH密钥对的名称查询指定SSH密钥对。
	创建SSH密钥对，或把公钥导入华为云中，生成SSH密钥对。 创建SSH密钥对成功后，请把响应数据中的私钥内容保存到本地文件，单用户使用该私钥登录云服务器。为保证云服务器安全，私钥数据只能下载一次，请妥善保管。
	根据SSH密钥对的名称，删除指定SSH密钥对。
	在同一个租户下可能包含多个单用户账号，将同一租户下目标单用户账号下的密钥对复制到当前单用户账号下。

4 API 说明

4.1 管理加密密钥

4.1.1 密钥生命周期管理

4.1.1.1 创建密钥

功能介绍

创建用户主密钥，用户主密钥可以为对称密钥或非对称密钥。

- 对称密钥为长度为256位AES密钥或者128位SM4密钥，可用于小量数据的加密或者用于加密数据密钥。
- 非对称密钥可以为RSA密钥对或者ECC密钥对（包含SM2密钥对），可用于加解密数据、数字签名及验签。

接口约束

别名“/default”为服务默认主密钥的后缀名，由服务自动创建。因此用户创建的主密钥别名不能与服务默认主密钥的别名相同，即后缀名不能为“/default”。

对于开通企业项目的用户，服务默认主密钥属于且只能属于默认企业项目下，且不支持企业资源的迁入迁出。服务默认主密钥为用户提供基础的云上加密功能，满足合规要求。因此，在企业多项目下，其他非默认企业项目下的用户均可使用该密钥。若客户有企业管理资源诉求，请自行创建和使用密钥。

调用方法

请参见**如何调用API**。

URI

POST /v1.0/{project_id}/kms/create-key

表 4-1 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-2 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-3 请求 Body 参数

参数	是否必选	参数类型	描述
key_alias	是	String	非默认主密钥别名，取值范围为1到255个字符，满足正则匹配“^[a-zA-Z0-9:/_-]{1,255}$”，且不与系统服务创建的默认主密钥别名重名。
key_spec	否	String	密钥生成算法，默认为“AES_256”，枚举如下： ● AES_256 ● SM4 ● RSA_2048 ● RSA_3072 ● RSA_4096 ● EC_P256 ● EC_P384 ● SM2
key_usage	否	String	密钥用途，对称密钥默认为“ENCRYPT_DECRYPT”，非对称密钥默认为“SIGN_VERIFY”，枚举如下： ● ENCRYPT_DECRYPT ● SIGN_VERIFY

参数	是否必选	参数类型	描述
key_description	否	String	密钥描述，取值0到255字符。
origin	否	String	密钥来源，默认为“kms”，枚举如下： • kms：表示密钥材料由kms生成。 • external：表示密钥材料由外部导入。
enterprise_project_id	否	String	企业多项目ID。 • 用户未开通企业多项目时，不需要输入该字段。 • 用户开通企业多项目时，创建资源可以输入该字段。若用户户不输入该字段，默认创建属于默认企业多项目ID（ID为“0”）的资源。 注意：若用户没有默认企业多项目ID（ID为“0”）下的创建权限，则接口报错。
sequence	否	String	请求消息序列号，36字节序列号。 例如：919c82d4-8046-4722-9094-35c3c6524cff
keystore_id	否	String	密钥库ID，默认使用KMS默认密钥库

响应参数

状态码： 200

表 4-4 响应 Body 参数

参数	参数类型	描述
key_info	**KeKInfo** object	密钥详细信息。

表 4-5 KeKInfo

参数	参数类型	描述
key_id	String	密钥ID。
domain_id	String	用户域ID。

状态码： 400

表 4-6 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-7 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-8 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-9 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-10 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-11 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-12 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-13 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-14 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-15 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-16 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-17 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-18 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-19 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

创建一个别名为"test"的密钥。

```
{
  "key_alias" : "test"
}
```

响应示例

状态码： 200

请求已成功

```
{
  "key_info" : {
    "key_id" : "bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e",
    "domain_id" : "b168fe00ff56492495a7d22974df2d0b"
```

```
    }
}
```

SDK 代码示例

SDK代码示例如下。

Java

创建一个别名为"test"的密钥。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;


public class CreateKeySolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        CreateKeyRequest request = new CreateKeyRequest();
        CreateKeyRequestBody body = new CreateKeyRequestBody();
        body.withKeyAlias("test");
        request.withBody(body);
        try {
            CreateKeyResponse response = client.createKey(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

创建一个别名为"test"的密钥。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = CreateKeyRequest()
        request.body = CreateKeyRequestBody(
            key_alias="test"
        )
        response = client.create_key(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

创建一个别名为"test"的密钥。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
```

```
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.CreateKeyRequest{}
    request.Body = &model.CreateKeyRequestBody{
        KeyAlias: "test",
    }
    response, err := client.CreateKey(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.1.2 启用密钥

功能介绍

- 功能介绍：启用密钥，密钥启用后才可以使用。
- 说明：密钥为禁用状态才能启用密钥。

调用方法

请参见**如何调用API**。

URI

POST /v1.0/{project_id}/kms/enable-key

表 4-20 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-21 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-22 请求 Body 参数

参数	是否必选	参数类型	描述
key_id	是	String	密钥ID，36字节，满足正则匹配“^[0-9a-z]{8}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{12}$”。 例如：0d0466b0-e727-4d9c-b35d-f84bb474a37f。
sequence	否	String	请求消息序列号，36字节序列号。 例如：919c82d4-8046-4722-9094-35c3c6524cff

响应参数

状态码： 200

表 4-23 响应 Body 参数

参数	参数类型	描述
key_info	KeyStatusInfo object	密钥状态信息。

表 4-24 KeyStatusInfo

参数	参数类型	描述
key_id	String	密钥ID
key_state	String	密钥状态： • 2为启用状态 • 3为禁用状态 • 4为计划删除状态 • 5为等待导入状态 • 7为冻结状态

状态码： 400

表 4-25 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-26 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-27 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-28 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-29 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-30 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-31 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-32 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-33 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-34 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-35 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-36 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-37 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-38 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

启用ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥。

```
{
  "key_id" : "0d0466b0-e727-4d9c-b35d-f84bb474a37f"
}
```

响应示例

状态码： 200

请求已成功

```
{
  "key_info" : {
    "key_id" : "bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e",
    "key_state" : "2"
  }
}
```

SDK 代码示例

SDK代码示例如下。

Java

启用ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;


public class EnableKeySolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        EnableKeyRequest request = new EnableKeyRequest();
        OperateKeyRequestBody body = new OperateKeyRequestBody();
        body.withKeyId("0d0466b0-e727-4d9c-b35d-f84bb474a37f");
        request.withBody(body);
```

```
        try {
            EnableKeyResponse response = client.enableKey(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

启用ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = EnableKeyRequest()
        request.body = OperateKeyRequestBody(
            key_id="0d0466b0-e727-4d9c-b35d-f84bb474a37f"
        )
        response = client.enable_key(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

启用ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
```

```
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.EnableKeyRequest{}
    request.Body = &model.OperateKeyRequestBody{
        KeyId: "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
    }
    response, err := client.EnableKey(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.1.3 禁用密钥

功能介绍

- 功能介绍：禁用密钥，密钥禁用后不可以使用。
- 说明：密钥为启用状态才能禁用密钥。

调用方法

请参见**如何调用API**。

URI

POST /v1.0/{project_id}/kms/disable-key

表 4-39 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-40 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-41 请求 Body 参数

参数	是否必选	参数类型	描述
key_id	是	String	密钥ID，36字节，满足正则匹配“^[0-9a-z]{8}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{12}$”。 例如：0d0466b0-e727-4d9c-b35d-f84bb474a37f。

参数	是否必选	参数类型	描述
sequence	否	String	请求消息序列号，36字节序列号。 例如：919c82d4-8046-4722-9094-35c3c6524cff

响应参数

状态码： 200

表 4-42 响应 Body 参数

参数	参数类型	描述
key_info	**KeyStatusInfo** object	密钥状态信息。

表 4-43 KeyStatusInfo

参数	参数类型	描述
key_id	String	密钥ID
key_state	String	密钥状态： • 2为启用状态 • 3为禁用状态 • 4为计划删除状态 • 5为等待导入状态 • 7为冻结状态

状态码： 400

表 4-44 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-45 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-46 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-47 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-48 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-49 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-50 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-51 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-52 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-53 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-54 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-55 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-56 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-57 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

禁用ID为“0d0466b0-e727-4d9c-b35d-f84bb474a37f”的密钥。

```
{
  "key_id" : "0d0466b0-e727-4d9c-b35d-f84bb474a37f"
}
```

响应示例

状态码： 200

请求已成功

```
{
  "key_info" : {
    "key_id" : "bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e",
    "key_state" : "3"
  }
}
```

SDK 代码示例

SDK代码示例如下。

Java

禁用ID为“0d0466b0-e727-4d9c-b35d-f84bb474a37f”的密钥。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;
```

```
public class DisableKeySolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        DisableKeyRequest request = new DisableKeyRequest();
        OperateKeyRequestBody body = new OperateKeyRequestBody();
        body.withKeyId("0d0466b0-e727-4d9c-b35d-f84bb474a37f");
        request.withBody(body);
        try {
            DisableKeyResponse response = client.disableKey(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

禁用ID为“0d0466b0-e727-4d9c-b35d-f84bb474a37f”的密钥。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
```

```
        .build()

    try:
        request = DisableKeyRequest()
        request.body = OperateKeyRequestBody(
            key_id="0d0466b0-e727-4d9c-b35d-f84bb474a37f"
        )
        response = client.disable_key(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

禁用ID为“0d0466b0-e727-4d9c-b35d-f84bb474a37f”的密钥。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.DisableKeyRequest{}
    request.Body = &model.OperateKeyRequestBody{
        KeyId: "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
    }
    response, err := client.DisableKey(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.1.4 计划删除密钥

功能介绍

- 功能介绍：计划多少天后删除密钥，可设置7天～1096天内删除密钥。

调用方法

请参见**如何调用API**。

URI

POST /v1.0/{project_id}/kms/schedule-key-deletion

表 4-58 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-59 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-60 请求 Body 参数

参数	是否必选	参数类型	描述
key_id	是	String	密钥ID，36字节，满足正则匹配“^[0-9a-z]{8}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{12}$”。 例如：0d0466b0-e727-4d9c-b35d-f84bb474a37f。
pending_days	是	String	计划多少天后删除密钥，取值为7到1096。
sequence	否	String	请求消息序列号，36字节序列号。 例如：919c82d4-8046-4722-9094-35c3c6524cff

响应参数

状态码： 200

表 4-61 响应 Body 参数

参数	参数类型	描述
key_id	String	密钥ID
key_state	String	密钥状态： • 2为启用状态 • 3为禁用状态 • 4为计划删除状态 • 5为等待导入状态 • 7为冻结状态

状态码： 400

表 4-62 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-63 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-64 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-65 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-66 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-67 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码

参数	参数类型	描述
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-68 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-69 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-70 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-71 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-72 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-73 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-74 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-75 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

计划删除ID为“0d0466b0-e727-4d9c-b35d-f84bb474a37f”的密钥，延期删除7天。

```
{
  "key_id" : "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
  "pending_days" : "7"
}
```

响应示例

状态码： 200

请求已成功

```
{
  "key_id" : "bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e",
  "key_state" : "4"
}
```

SDK 代码示例

SDK代码示例如下。

Java

计划删除ID为“0d0466b0-e727-4d9c-b35d-f84bb474a37f”的密钥，延期删除7天。

```
package com.huaweicloud.sdk.test;
```

```
import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;


public class DeleteKeySolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        DeleteKeyRequest request = new DeleteKeyRequest();
        ScheduleKeyDeletionRequestBody body = new ScheduleKeyDeletionRequestBody();
        body.withPendingDays("7");
        body.withKeyId("0d0466b0-e727-4d9c-b35d-f84bb474a37f");
        request.withBody(body);
        try {
            DeleteKeyResponse response = client.deleteKey(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

计划删除ID为“0d0466b0-e727-4d9c-b35d-f84bb474a37f”的密钥，延期删除7天。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
```

```
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = DeleteKeyRequest()
        request.body = ScheduleKeyDeletionRequestBody(
            pending_days="7",
            key_id="0d0466b0-e727-4d9c-b35d-f84bb474a37f"
        )
        response = client.delete_key(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

计划删除ID为“0d0466b0-e727-4d9c-b35d-f84bb474a37f”的密钥，延期删除7天。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.DeleteKeyRequest{}
    request.Body = &model.ScheduleKeyDeletionRequestBody{
        PendingDays: "7",
        KeyId: "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
    }
    response, err := client.DeleteKey(request)
```

```
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.1.5 取消计划删除密钥

功能介绍

- 功能介绍：取消计划删除密钥。
- 说明：密钥处于“计划删除”状态才能取消计划删除密钥。

调用方法

请参见**如何调用API**。

URI

POST /v1.0/{project_id}/kms/cancel-key-deletion

表 4-76 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-77 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-78 请求 Body 参数

参数	是否必选	参数类型	描述
key_id	是	String	密钥ID，36字节，满足正则匹配“^[0-9a-z]{8}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{12}$”。 例如：0d0466b0-e727-4d9c-b35d-f84bb474a37f。
sequence	否	String	请求消息序列号，36字节序列号。 例如：919c82d4-8046-4722-9094-35c3c6524cff

响应参数

状态码： 200

表 4-79 响应 Body 参数

参数	参数类型	描述
key_id	String	密钥ID

参数	参数类型	描述
key_state	String	密钥状态： • 2为启用状态 • 3为禁用状态 • 4为计划删除状态 • 5为等待导入状态 • 7为冻结状态

状态码： 400

表 4-80 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-81 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-82 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-83 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-84 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-85 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-86 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-87 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-88 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-89 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-90 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-91 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-92 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-93 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

取消已计划删除状态ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥。

```
{
  "key_id" : "0d0466b0-e727-4d9c-b35d-f84bb474a37f"
}
```

响应示例

状态码： 200

请求已成功

```
{
  "key_id" : "bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e",
  "key_state" : "3"
}
```

SDK 代码示例

SDK代码示例如下。

Java

取消已计划删除状态ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;


public class CancelKeyDeletionSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        CancelKeyDeletionRequest request = new CancelKeyDeletionRequest();
        OperateKeyRequestBody body = new OperateKeyRequestBody();
        body.withKeyId("0d0466b0-e727-4d9c-b35d-f84bb474a37f");
        request.withBody(body);
        try {
            CancelKeyDeletionResponse response = client.cancelKeyDeletion(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

取消已计划删除状态ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥。

```
# coding: utf-8
```

```
import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = CancelKeyDeletionRequest()
        request.body = OperateKeyRequestBody(
            key_id="0d0466b0-e727-4d9c-b35d-f84bb474a37f"
        )
        response = client.cancel_key_deletion(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

取消已计划删除状态ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
```

```
        WithCredential(auth).
        Build())

    request := &model.CancelKeyDeletionRequest{}
    request.Body = &model.OperateKeyRequestBody{
        KeyId: "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
    }
    response, err := client.CancelKeyDeletion(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.1.6 修改密钥别名

功能介绍

- 功能介绍：修改用户主密钥别名。
- 说明：
  - 服务默认主密钥（密钥别名后缀为“/default”）不可以修改。
  - 密钥处于“计划删除”状态，密钥别名不可以修改。

调用方法

请参见**如何调用API**。

URI

POST /v1.0/{project_id}/kms/update-key-alias

表 4-94 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-95 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-96 请求 Body 参数

参数	是否必选	参数类型	描述
key_id	是	String	密钥ID，36字节，满足正则匹配“^[0-9a-z]{8}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{12}$”。 例如：0d0466b0-e727-4d9c-b35d-f84bb474a37f。
key_alias	是	String	非默认主密钥别名，取值1到255字符，满足正则匹配“^[a-zA-Z0-9:/_-]{1,255}$”且后缀不可以为“/default”。
sequence	否	String	请求消息序列号，36字节序列号。 例如：919c82d4-8046-4722-9094-35c3c6524cff

响应参数

状态码： 200

表 4-97 响应 Body 参数

参数	参数类型	描述
key_info	**KeyAliasInfo** object	密钥别名信息。

表 4-98 KeyAliasInfo

参数	参数类型	描述
key_id	String	密钥ID。
key_alias	String	密钥别名。

状态码： 400

表 4-99 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-100 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-101 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-102 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码

参数	参数类型	描述
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-103 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-104 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-105 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-106 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-107 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-108 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-109 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-110 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-111 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-112 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

更新ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥的别名为"test"。

```
{
  "key_id" : "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
```

```
  "key_alias" : "test"
}
```

响应示例

状态码： 200

请求已成功

```
{
  "key_info" : {
    "key_id" : "bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e",
    "key_alias" : "test"
  }
}
```

SDK 代码示例

SDK代码示例如下。

Java

更新ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥的别名为"test"。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;


public class UpdateKeyAliasSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        UpdateKeyAliasRequest request = new UpdateKeyAliasRequest();
        UpdateKeyAliasRequestBody body = new UpdateKeyAliasRequestBody();
        body.withKeyAlias("test");
        body.withKeyId("0d0466b0-e727-4d9c-b35d-f84bb474a37f");
        request.withBody(body);
        try {
            UpdateKeyAliasResponse response = client.updateKeyAlias(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
```

```
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

更新ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥的别名为"test"。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = UpdateKeyAliasRequest()
        request.body = UpdateKeyAliasRequestBody(
            key_alias="test",
            key_id="0d0466b0-e727-4d9c-b35d-f84bb474a37f"
        )
        response = client.update_key_alias(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

更新ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥的别名为"test"。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
```

```
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.UpdateKeyAliasRequest{}
    request.Body = &model.UpdateKeyAliasRequestBody{
        KeyAlias: "test",
        KeyId: "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
    }
    response, err := client.UpdateKeyAlias(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.1.7 修改密钥描述

功能介绍

- 功能介绍：修改用户主密钥描述信息。
- 说明：
  - 服务默认主密钥（密钥别名后缀为“/default”）不可以修改。
  - 密钥处于“计划删除”状态，密钥描述不可以修改。

调用方法

请参见**如何调用API**。

URI

POST /v1.0/{project_id}/kms/update-key-description

表 4-113 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-114 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-115 请求 Body 参数

参数	是否必选	参数类型	描述
key_id	是	String	密钥ID，36字节，满足正则匹配“^[0-9a-z]{8}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{12}$”。 例如：0d0466b0-e727-4d9c-b35d-f84bb474a37f。
key_description	是	String	密钥描述，取值0到255字符。
sequence	否	String	请求消息序列号，36字节序列号。 例如：919c82d4-8046-4722-9094-35c3c6524cff

响应参数

状态码： 200

表 4-116 响应 Body 参数

参数	参数类型	描述
key_info	**KeyDescriptionInfo** object	密钥描述信息。

表 4-117 KeyDescriptionInfo

参数	参数类型	描述
key_id	String	密钥ID。
key_description	String	密钥描述。

状态码： 400

表 4-118 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-119 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-120 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-121 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-122 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-123 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-124 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-125 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-126 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-127 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-128 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-129 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-130 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-131 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

更新ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥的描述为"test"。

```
{
  "key_id" : "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
  "key_description" : "test"
}
```

响应示例

状态码： 200

请求已成功

```
{
  "key_info" : {
    "key_id" : "bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e",
    "key_description" : "test"
  }
}
```

SDK 代码示例

SDK代码示例如下。

Java

更新ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥的描述为"test"。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;
```

```
public class UpdateKeyDescriptionSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        UpdateKeyDescriptionRequest request = new UpdateKeyDescriptionRequest();
        UpdateKeyDescriptionRequestBody body = new UpdateKeyDescriptionRequestBody();
        body.withKeyDescription("test");
        body.withKeyId("0d0466b0-e727-4d9c-b35d-f84bb474a37f");
        request.withBody(body);
        try {
            UpdateKeyDescriptionResponse response = client.updateKeyDescription(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

更新ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥的描述为"test"。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
```

```
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = UpdateKeyDescriptionRequest()
        request.body = UpdateKeyDescriptionRequestBody(
            key_description="test",
            key_id="0d0466b0-e727-4d9c-b35d-f84bb474a37f"
        )
        response = client.update_key_description(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

更新ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥的描述为"test"。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.UpdateKeyDescriptionRequest{}
    request.Body = &model.UpdateKeyDescriptionRequestBody{
        KeyDescription: "test",
        KeyId: "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
    }
    response, err := client.UpdateKeyDescription(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.2 数据密钥管理

4.1.2.1 创建随机数

功能介绍

- 功能介绍：
  生成8~8192bit范围内的随机数。

调用方法

请参见**如何调用API**。

URI

POST /v1.0/{project_id}/kms/gen-random

表 4-132 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-133 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-134 请求 Body 参数

参数	是否必选	参数类型	描述
random_data_length	是	String	随机数的bit位长度。 取值为8的倍数，取值范围为8~8192。 随机数的bit位长度，取值为“512”。
sequence	否	String	请求消息序列号，36字节序列号。 例如：919c82d4-8046-4722-9094-35c3c6524cff

响应参数

状态码： 200

表 4-135 响应 Body 参数

参数	参数类型	描述
random_data	String	随机数16进制表示，两位表示1byte。随机数的长度与用户传入的参数“random_data_length”的长度保持一致。

状态码： 400

表 4-136 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-137 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-138 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-139 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-140 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-141 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-142 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-143 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-144 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-145 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-146 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-147 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-148 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-149 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

生成长度为512位的随机数。

```
{
  "random_data_length" : "512"
}
```

响应示例

状态码： 200

请求已成功

```
{
  "random_data" : "5791C223E87120BE4B98D168F47A58BB2A88834EEADC"
}
```

SDK 代码示例

SDK代码示例如下。

Java

生成长度为512位的随机数。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;


public class CreateRandomSolution {

    public static void main(String[] args) {
```

```
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        CreateRandomRequest request = new CreateRandomRequest();
        GenRandomRequestBody body = new GenRandomRequestBody();
        body.withRandomDataLength("512");
        request.withBody(body);
        try {
            CreateRandomResponse response = client.createRandom(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

生成长度为512位的随机数。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
```

```
        request = CreateRandomRequest()
        request.body = GenRandomRequestBody(
            random_data_length="512"
        )
        response = client.create_random(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

生成长度为512位的随机数。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.CreateRandomRequest{}
    request.Body = &model.GenRandomRequestBody{
        RandomDataLength: "512",
    }
    response, err := client.CreateRandom(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.2.2 创建数据密钥

功能介绍

- 功能介绍：创建数据密钥，返回结果包含明文和密文。

调用方法

请参见**如何调用API**。

URI

POST /v1.0/{project_id}/kms/create-datakey

表 4-150 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-151 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-152 请求 Body 参数

参数	是否必选	参数类型	描述
key_id	是	String	密钥ID，36字节，满足正则匹配“^[0-9a-z]{8}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{12}$”。 例如：0d0466b0-e727-4d9c-b35d-f84bb474a37f。
key_spec	否	String	指定生成的密钥bit位长度。有效值：AES_256、AES_128。 ● AES_256：表示256比特的对称密钥。 ● AES_128：表示128比特的对称密钥。 说明： datakey_length和key_spec二选一。 ● 若datakey_length和key_spec都为空，默认生成256bit的密钥。 ● 若datakey_length和key_spec都指定了值，仅datakey_length生效。

参数	是否必选	参数类型	描述
datakey_length	否	String	密钥bit位长度。取值为8的倍数，取值范围为8~8192。 说明： datakey_length和key_spec二选一。 • 若datakey_length和key_spec都为空，默认生成256bit的密钥。 • 若datakey_length和key_spec都指定了值，仅datakey_length生效。
additional_authenticated_data	否	String	身份验证的非敏感额外数据。任意字符串，长度不超过128字节。
sequence	否	String	请求消息序列号，36字节序列号。 例如：919c82d4-8046-4722-9094-35c3c6524cff

响应参数

状态码： 200

表 4-153 响应 Body 参数

参数	参数类型	描述
key_id	String	密钥ID。
plain_text	String	DEK明文16进制，两位表示1byte。
cipher_text	String	DEK密文16进制，两位表示1byte。

状态码： 400

表 4-154 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-155 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-156 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-157 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-158 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-159 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-160 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-161 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-162 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-163 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-164 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-165 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-166 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-167 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

创建"512"位长度的明文数据密钥，并使用ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥进行加密，返回明文密钥,附加数据为“123add”。

```
{
  "key_id" : "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
  "datakey_length" : "512",
  "additional_authenticated_data" : "123aad"
}
```

响应示例

状态码： 200

请求已成功

```
{
  "key_id" : "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
  "plain_text" : "8151014275E426C72EE7D44267XXXXX...",
  "cipher_text" : "020098009EEAFCE122CAA5927D2XXX..."
}
```

SDK 代码示例

SDK代码示例如下。

Java

创建"512"位长度的明文数据密钥，并使用ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥进行加密，返回明文密钥,附加数据为“123add”。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
```

```
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;


public class CreateDatakeySolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        CreateDatakeyRequest request = new CreateDatakeyRequest();
        CreateDatakeyRequestBody body = new CreateDatakeyRequestBody();
        body.withAdditionalAuthenticatedData("123aad");
        body.withDatakeyLength("512");
        body.withKeyId("0d0466b0-e727-4d9c-b35d-f84bb474a37f");
        request.withBody(body);
        try {
            CreateDatakeyResponse response = client.createDatakey(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

创建"512"位长度的明文数据密钥，并使用ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥进行加密，返回明文密钥,附加数据为“123add”。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
```

```
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = CreateDatakeyRequest()
        request.body = CreateDatakeyRequestBody(
            additional_authenticated_data="123aad",
            datakey_length="512",
            key_id="0d0466b0-e727-4d9c-b35d-f84bb474a37f"
        )
        response = client.create_datakey(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

创建"512"位长度的明文数据密钥，并使用ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥进行加密，返回明文密钥,附加数据为“123add”。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.CreateDatakeyRequest{}
    additionalAuthenticatedDataCreateDatakeyRequestBody:= "123aad"
    datakeyLengthCreateDatakeyRequestBody:= "512"
    request.Body = &model.CreateDatakeyRequestBody{
        AdditionalAuthenticatedData: &additionalAuthenticatedDataCreateDatakeyRequestBody,
        DatakeyLength: &datakeyLengthCreateDatakeyRequestBody,
        KeyId: "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
    }
```

```
    response, err := client.CreateDatakey(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.2.3 创建不含明文数据密钥

功能介绍

- 功能介绍：创建数据密钥，返回结果只包含密文。

调用方法

请参见**如何调用API**。

URI

POST /v1.0/{project_id}/kms/create-datakey-without-plaintext

表 4-168 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-169 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-170 请求 Body 参数

参数	是否必选	参数类型	描述
key_id	是	String	密钥ID，36字节，满足正则匹配“^[0-9a-z]{8}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{12}$”。 例如：0d0466b0-e727-4d9c-b35d-f84bb474a37f。
key_spec	否	String	指定生成的密钥bit位长度。有效值：AES_256、AES_128。 ● AES_256：表示256比特的对称密钥。 ● AES_128：表示128比特的对称密钥。 说明： datakey_length和key_spec二选一。 ● 若datakey_length和key_spec都为空，默认生成256bit的密钥。 ● 若datakey_length和key_spec都指定了值，仅datakey_length生效。

参数	是否必选	参数类型	描述
datakey_length	否	String	密钥bit位长度。取值为8的倍数，取值范围为8~8192。 说明： datakey_length和key_spec二选一。 ● 若datakey_length和key_spec都为空，默认生成256bit的密钥。 ● 若datakey_length和key_spec都指定了值，仅datakey_length生效。
additional_authenticated_data	否	String	身份验证的非敏感额外数据。任意字符串，长度不超过128字节。
sequence	否	String	请求消息序列号，36字节序列号。 例如：919c82d4-8046-4722-9094-35c3c6524cff

响应参数

状态码： 200

表 4-171 响应 Body 参数

参数	参数类型	描述
key_id	String	密钥ID。
cipher_text	String	加密后的密文，使用了Base64编码

状态码： 400

表 4-172 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-173 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-174 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-175 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-176 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-177 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-178 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-179 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-180 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-181 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-182 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-183 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-184 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-185 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

创建"512"位长度的明文数据密钥，并使用ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥进行加密，不返回明文密钥,附加数据为“123add”。

```
{
  "key_id" : "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
  "datakey_length" : "512",
  "additional_authenticated_data" : "123aad"
}
```

响应示例

状态码： 200

请求已成功

```
{
  "key_id" : "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
  "cipher_text" : "020098009EEAFCE122CAA5927D2XXX..."
}
```

SDK 代码示例

SDK代码示例如下。

Java

创建"512"位长度的明文数据密钥，并使用ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥进行加密，不返回明文密钥,附加数据为“123add”。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
```

```
import com.huaweicloud.sdk.kms.v2.model.*;


public class CreateDatakeyWithoutPlaintextSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        CreateDatakeyWithoutPlaintextRequest request = new CreateDatakeyWithoutPlaintextRequest();
        CreateDatakeyRequestBody body = new CreateDatakeyRequestBody();
        body.withAdditionalAuthenticatedData("123aad");
        body.withDatakeyLength("512");
        body.withKeyId("0d0466b0-e727-4d9c-b35d-f84bb474a37f");
        request.withBody(body);
        try {
            CreateDatakeyWithoutPlaintextResponse response = client.createDatakeyWithoutPlaintext(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

创建"512"位长度的明文数据密钥，并使用ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥进行加密，不返回明文密钥,附加数据为“123add”。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"
```

```
credentials = BasicCredentials(ak, sk, projectId)

client = KmsClient.new_builder() \
    .with_credentials(credentials) \
    .with_region(KmsRegion.value_of("<YOUR REGION>")) \
    .build()

try:
    request = CreateDatakeyWithoutPlaintextRequest()
    request.body = CreateDatakeyRequestBody(
        additional_authenticated_data="123aad",
        datakey_length="512",
        key_id="0d0466b0-e727-4d9c-b35d-f84bb474a37f"
    )
    response = client.create_datakey_without_plaintext(request)
    print(response)
except exceptions.ClientRequestException as e:
    print(e.status_code)
    print(e.request_id)
    print(e.error_code)
    print(e.error_msg)
```

Go

创建"512"位长度的明文数据密钥，并使用ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥进行加密，不返回明文密钥,附加数据为“123add”。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.CreateDatakeyWithoutPlaintextRequest{}
    additionalAuthenticatedDataCreateDatakeyRequestBody:= "123aad"
    datakeyLengthCreateDatakeyRequestBody:= "512"
    request.Body = &model.CreateDatakeyRequestBody{
        AdditionalAuthenticatedData: &additionalAuthenticatedDataCreateDatakeyRequestBody,
        DatakeyLength: &datakeyLengthCreateDatakeyRequestBody,
        KeyId: "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
    }
    response, err := client.CreateDatakeyWithoutPlaintext(request)
```

```
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.2.4 加密数据密钥

功能介绍

- 功能介绍：加密数据密钥，用指定的主密钥加密数据密钥。

调用方法

请参见**如何调用API**。

URI

POST /v1.0/{project_id}/kms/encrypt-datakey

表 4-186 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-187 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-188 请求 Body 参数

参数	是否必选	参数类型	描述
key_id	是	String	密钥ID，36字节，满足正则匹配“^[0-9a-z]{8}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{12}$”。 例如：0d0466b0-e727-4d9c-b35d-f84bb474a37f。
plain_text	是	String	使用的主密钥算法为AES时，传入数据密钥的明文+数据密钥明文的SHA256值（32字节）；使用的主密钥算法为SM4时，传入数据密钥的明文+数据密钥明文的SM3值（32字节），均为16进制字符串表示。本参数中的两部分，通过字符串拼接后作为参数即可。
datakey_plain_length	是	String	DEK明文字节长度，取值范围为1~1024。 DEK明文字节长度，取值为“64”。
additional_authenticated_data	否	String	身份验证的非敏感额外数据。任意字符串，长度不超过128字节。

参数	是否必选	参数类型	描述
sequence	否	String	请求消息序列号，36字节序列号。 例如：919c82d4-8046-4722-9094-35c3c6524cff

响应参数

状态码： 200

表 4-189 响应 Body 参数

参数	参数类型	描述
key_id	String	密钥ID
cipher_text	String	DEK密文16进制，两位表示1byte。
datakey_length	String	DEK字节长度。

状态码： 400

表 4-190 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-191 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-192 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-193 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-194 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-195 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-196 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-197 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-198 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-199 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-200 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-201 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-202 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-203 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

加密明文数据密钥，使用ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥，加密512位长度的明文密钥,附加数据为“123add”。

```
{
  "key_id" : "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
  "plain_text" :
"7549d9aea901767bf3c0b3e14b10722eaf6f59053bbd82045d04e075e809a0fe6ccab48f8e5efe74e4b18ff0512
525e527b10331100f357bf42125d8d5ced94ffbc8ac72b0785ca7fe33eb6776ce3990b11e32b299d9c0a9ee0305f
b9540f797",
  "datakey_plain_length" : "64",
  "additional_authenticated_data" : "123aad"
}
```

响应示例

状态码： 200

请求已成功

```
{
  "key_id" : "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
  "datakey_length" : "64",
  "cipher_text" : "020098009EEAFCE122CAA5927D2XXX..."
}
```

SDK 代码示例

SDK代码示例如下。

Java

加密明文数据密钥，使用ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥，加密512位长度的明文密钥,附加数据为“123add”。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;


public class EncryptDatakeySolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);
```

```
        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        EncryptDatakeyRequest request = new EncryptDatakeyRequest();
        EncryptDatakeyRequestBody body = new EncryptDatakeyRequestBody();
        body.withAdditionalAuthenticatedData("123aad");
        body.withDatakeyPlainLength("64");

body.withPlainText("7549d9aea901767bf3c0b3e14b10722eaf6f59053bbd82045d04e075e809a0fe6ccab48f8e
5efe74e4b18ff0512525e527b10331100f357bf42125d8d5ced94ffbc8ac72b0785ca7fe33eb6776ce3990b11e32
b299d9c0a9ee0305fb9540f797");
        body.withKeyId("0d0466b0-e727-4d9c-b35d-f84bb474a37f");
        request.withBody(body);
        try {
            EncryptDatakeyResponse response = client.encryptDatakey(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

加密明文数据密钥，使用ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥，加密512位长度的明文密钥,附加数据为“123add”。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this
example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = EncryptDatakeyRequest()
        request.body = EncryptDatakeyRequestBody(
            additional_authenticated_data="123aad",
            datakey_plain_length="64",

plain_text="7549d9aea901767bf3c0b3e14b10722eaf6f59053bbd82045d04e075e809a0fe6ccab48f8e5efe74e4
b18ff0512525e527b10331100f357bf42125d8d5ced94ffbc8ac72b0785ca7fe33eb6776ce3990b11e32b299d9c0
```

```
a9ee0305fb9540f797",
            key_id="0d0466b0-e727-4d9c-b35d-f84bb474a37f"
        )
        response = client.encrypt_datakey(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

加密明文数据密钥，使用ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥，加密512位长度的明文密钥,附加数据为“123add”。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.EncryptDatakeyRequest{}
    additionalAuthenticatedDataEncryptDatakeyRequestBody:= "123aad"
    request.Body = &model.EncryptDatakeyRequestBody{
        AdditionalAuthenticatedData: &additionalAuthenticatedDataEncryptDatakeyRequestBody,
        DatakeyPlainLength: "64",
        PlainText: "7549d9aea901767bf3c0b3e14b10722eaf6f59053bbd82045d04e075e809a0fe6ccab48f8e5efe74e4b18ff0512525e527b10331100f357bf42125d8d5ced94ffbc8ac72b0785ca7fe33eb6776ce3990b11e32b299d9c0a9ee0305fb9540f797",
        KeyId: "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
    }
    response, err := client.EncryptDatakey(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.2.5 解密数据密钥

功能介绍

- 功能介绍：解密数据密钥，用指定的主密钥解密数据密钥。

调用方法

请参见**如何调用API**。

URI

POST /v1.0/{project_id}/kms/decrypt-datakey

表 4-204 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-205 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-206 请求 Body 参数

参数	是否必选	参数类型	描述
key_id	是	String	密钥ID，36字节，满足正则匹配“^[0-9a-z]{8}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{12}$”。 例如：0d0466b0-e727-4d9c-b35d-f84bb474a37f。
cipher_text	是	String	DEK密文及元数据的16进制字符串。取值为加密数据密钥结果中的cipher_text的值。
datakey_cipher_length	是	String	密钥字节长度，取值范围为1~1024。 密钥字节长度，取值为“64”。
additional_authenticated_data	否	String	身份验证的非敏感额外数据。任意字符串，长度不超过128字节。
sequence	否	String	请求消息序列号，36字节序列号。 例如：919c82d4-8046-4722-9094-35c3c6524cff

响应参数

状态码： 200

表 4-207 响应 Body 参数

参数	参数类型	描述
data_key	String	DEK明文的16进制字符串。
datakey_length	String	DEK明文字节长度。
datakey_dgst	String	DEK明文的SHA256值对应的16进制字符串。

状态码： 400

表 4-208 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-209 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-210 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-211 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-212 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-213 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-214 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-215 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-216 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-217 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-218 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-219 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-220 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-221 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

解密数据密钥，使用ID为“0d0466b0-e727-4d9c-b35d-f84bb474a37f”的密钥，解密明文长度为64字节的密文数据密钥，附加数据为“123aad”。

```
{
  "key_id" : "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
  "cipher_text" : "020098005273E14E6E8E95F5463BECDC27E80AFxxxxxxxxxx...",
  "datakey_cipher_length" : "64",
  "additional_authenticated_data" : "123aad"
}
```

响应示例

状态码： 200

请求已成功

```
{
  "data_key" : "000000e724d9cb35df84bb474a37fXXX...",
  "datakey_length" : "64",
  "datakey_dgst" : "F5A5FD42D16A20302798EF6ED3099XXX..."
}
```

SDK 代码示例

SDK代码示例如下。

Java

解密数据密钥，使用ID为“0d0466b0-e727-4d9c-b35d-f84bb474a37f”的密钥，解密明文长度为64字节的密文数据密钥，附加数据为“123aad”。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;


public class DecryptDatakeySolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        DecryptDatakeyRequest request = new DecryptDatakeyRequest();
        DecryptDatakeyRequestBody body = new DecryptDatakeyRequestBody();
        body.withAdditionalAuthenticatedData("123aad");
        body.withDatakeyCipherLength("64");
        body.withCipherText("020098005273E14E6E8E95F5463BECDC27E80AFxxxxxxxxxx...");
        body.withKeyId("0d0466b0-e727-4d9c-b35d-f84bb474a37f");
        request.withBody(body);
        try {
            DecryptDatakeyResponse response = client.decryptDatakey(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
```

```
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

解密数据密钥，使用ID为“0d0466b0-e727-4d9c-b35d-f84bb474a37f”的密钥，解密明文长度为64字节的密文数据密钥，附加数据为“123aad”。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = DecryptDatakeyRequest()
        request.body = DecryptDatakeyRequestBody(
            additional_authenticated_data="123aad",
            datakey_cipher_length="64",
            cipher_text="020098005273E14E6E8E95F5463BECDC27E80AFxxxxxxxxxx...",
            key_id="0d0466b0-e727-4d9c-b35d-f84bb474a37f"
        )
        response = client.decrypt_datakey(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

解密数据密钥，使用ID为“0d0466b0-e727-4d9c-b35d-f84bb474a37f”的密钥，解密明文长度为64字节的密文数据密钥，附加数据为“123aad”。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
```

```
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.DecryptDatakeyRequest{}
    additionalAuthenticatedDataDecryptDatakeyRequestBody:= "123aad"
    request.Body = &model.DecryptDatakeyRequestBody{
        AdditionalAuthenticatedData: &additionalAuthenticatedDataDecryptDatakeyRequestBody,
        DatakeyCipherLength: "64",
        CipherText: "020098005273E14E6E8E95F5463BECDC27E80AFxxxxxxxxxx...",
        KeyId: "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
    }
    response, err := client.DecryptDatakey(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.3 导入密钥管理

4.1.3.1 获取密钥导入参数

功能介绍

- 功能介绍：获取导入密钥的必要参数，包括密钥导入令牌和密钥加密公钥。
- 说明：返回的公钥类型默认为RSA_2048。

调用方法

请参见**如何调用API**。

URI

POST /v1.0/{project_id}/kms/get-parameters-for-import

表 4-222 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-223 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-224 请求 Body 参数

参数	是否必选	参数类型	描述
key_id	是	String	密钥ID，36字节，满足正则匹配“^[0-9a-z]{8}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{12}$”。 例如：0d0466b0-e727-4d9c-b35d-f84bb474a37f。
wrapping_algorithm	是	String	密钥材料加密算法，枚举如下： ● RSAES_OAEP_SHA_256 ● SM2_ENCRYPT，部分局点不支持该导入类型
sequence	否	String	请求消息序列号，36字节序列号。 例如：919c82d4-8046-4722-9094-35c3c6524cff

响应参数

状态码： 200

表 4-225 响应 Body 参数

参数	参数类型	描述
key_id	String	密钥ID。
import_token	String	密钥导入令牌。
expiration_time	Long	导入参数到期时间，时间戳，即从1970年1月1日至该时间的总秒数。
public_key	String	加密密钥材料的公钥，base64格式。

状态码： 400

表 4-226 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-227 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-228 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-229 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-230 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-231 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-232 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-233 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-234 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-235 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-236 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-237 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-238 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-239 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

获取ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"密钥包装材料，密钥包装算法使用"RSAES_OAEP_SHA_256"。

```
{
  "key_id" : "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
  "wrapping_algorithm" : "RSAES_OAEP_SHA_256"
}
```

响应示例

状态码： 200

请求已成功

```
{
  "key_id" : "bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e",
  "import_token" : "AACIBjY2ZTQxYjBmLTY3ZWItNDU4Ny04OTIxLWVhZXXX...",
  "expiration_time" : 1501578672,
  "public_key" : "MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCXXX..."
}
```

SDK 代码示例

SDK代码示例如下。

Java

获取ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"密钥包装材料，密钥包装算法使用"RSAES_OAEP_SHA_256"。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
```

```
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;


public class CreateParametersForImportSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        CreateParametersForImportRequest request = new CreateParametersForImportRequest();
        GetParametersForImportRequestBody body = new GetParametersForImportRequestBody();

body.withWrappingAlgorithm(GetParametersForImportRequestBody.WrappingAlgorithmEnum.fromValue("RSAES_OAEP_SHA_256"));
        body.withKeyId("0d0466b0-e727-4d9c-b35d-f84bb474a37f");
        request.withBody(body);
        try {
            CreateParametersForImportResponse response = client.createParametersForImport(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

获取ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"密钥包装材料，密钥包装算法使用"RSAES_OAEP_SHA_256"。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
```

```
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = CreateParametersForImportRequest()
        request.body = GetParametersForImportRequestBody(
            wrapping_algorithm="RSAES_OAEP_SHA_256",
            key_id="0d0466b0-e727-4d9c-b35d-f84bb474a37f"
        )
        response = client.create_parameters_for_import(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

获取ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"密钥包装材料，密钥包装算法使用"RSAES_OAEP_SHA_256"。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.CreateParametersForImportRequest{}
    request.Body = &model.GetParametersForImportRequestBody{
        WrappingAlgorithm: model.GetGetParametersForImportRequestBodyWrappingAlgorithmEnum().RSAES_OAEP_SHA_256,
        KeyId: "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
    }
    response, err := client.CreateParametersForImport(request)
    if err == nil {
```

```
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.3.2 导入密钥材料

功能介绍

- 功能介绍：导入密钥材料。

调用方法

请参见**如何调用API**。

URI

POST /v1.0/{project_id}/kms/import-key-material

表 4-240 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-241 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-242 请求 Body 参数

参数	是否必选	参数类型	描述
key_id	是	String	密钥ID，36字节，满足正则匹配“^[0-9a-z]{8}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{12}$”。 例如：0d0466b0-e727-4d9c-b35d-f84bb474a37f。
import_token	是	String	密钥导入令牌，base64格式，满足正则匹配“^[0-9a-zA-Z+/=]{200,6144}$”。
encrypted_key_material	是	String	加密后的对称密钥材料，base64格式，满足正则匹配“^[0-9a-zA-Z+/=]{344,360}$”。若导入非对称密钥，则该参数为用于加密私钥的临时中间密钥。
encrypted_privatekey	否	String	使用临时中间密钥加密后的私钥，导入非对称密钥需要该参数，base64格式，满足正则匹配“^[0-9a-zA-Z+/=]{200,6144}$”。
expiration_time	否	Long	密钥材料到期时间，时间戳，即从1970年1月1日至该时间的总秒数，KMS会在该时间的24小时内删除密钥材料。 例如：1550291833

参数	是否必选	参数类型	描述
sequence	否	String	请求消息序列号，36字节序列号。 例如：919c82d4-8046-4722-9094-35c3c6524cff

响应参数

状态码： 400

表 4-243 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-244 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-245 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-246 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-247 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-248 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-249 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-250 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-251 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-252 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-253 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-254 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-255 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-256 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

导入密文密钥材料到ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥中。

```
{
  "key_id" : "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
  "import_token" : "AACIBjY2ZTQxYItNDU4Ny04OTIxLWVhZTVhZjg5NDZm....",
  "encrypted_key_material" : "e0wTU/YJT/HDxsEv2NE+3CKT1..."
}
```

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

导入密文密钥材料到ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥中。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;


public class ImportKeyMaterialSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        ImportKeyMaterialRequest request = new ImportKeyMaterialRequest();
        ImportKeyMaterialRequestBody body = new ImportKeyMaterialRequestBody();
        body.withEncryptedKeyMaterial("e0wTU/YJT/HDxsEv2NE+3CKT1...");
        body.withImportToken("AACIBjY2ZTQxYItNDU4Ny04OTIxLWVhZTVhZjg5NDZm....");
        body.withKeyId("0d0466b0-e727-4d9c-b35d-f84bb474a37f");
        request.withBody(body);
        try {
            ImportKeyMaterialResponse response = client.importKeyMaterial(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

导入密文密钥材料到ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥中。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
```

```
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ImportKeyMaterialRequest()
        request.body = ImportKeyMaterialRequestBody(
            encrypted_key_material="e0wTU/YJT/HDxsEv2NE+3CKT1...",
            import_token="AACIBjY2ZTQxYItNDU4Ny04OTIxLWVhZTVhZjg5NDZm....",
            key_id="0d0466b0-e727-4d9c-b35d-f84bb474a37f"
        )
        response = client.import_key_material(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

导入密文密钥材料到ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥中。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
```

```
            WithCredential(auth).
            Build())

    request := &model.ImportKeyMaterialRequest{}
    request.Body = &model.ImportKeyMaterialRequestBody{
        EncryptedKeyMaterial: "e0wTU/YJT/HDxsEv2NE+3CKT1...",
        ImportToken: "AACIBjY2ZTQxYItNDU4Ny04OTIxLWVhZTVhZjg5NDZm....",
        KeyId: "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
    }
    response, err := client.ImportKeyMaterial(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.3.3 删除密钥材料

功能介绍

- 功能介绍：删除密钥材料信息。

接口约束

导入的非对称密钥不支持删除密钥材料。若要删除非对称密钥，请使用计划删除密钥功能。

调用方法

请参见**如何调用API**。

URI

POST /v1.0/{project_id}/kms/delete-imported-key-material

表 4-257 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-258 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-259 请求 Body 参数

参数	是否必选	参数类型	描述
key_id	是	String	密钥ID，36字节，满足正则匹配“^[0-9a-z]{8}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{12}$”。 例如：0d0466b0-e727-4d9c-b35d-f84bb474a37f。
sequence	否	String	请求消息序列号，36字节序列号。 例如：919c82d4-8046-4722-9094-35c3c6524cff

响应参数

状态码： 400

表 4-260 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-261 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-262 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-263 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-264 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-265 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-266 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-267 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-268 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-269 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-270 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-271 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码

参数	参数类型	描述
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-272 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-273 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

删除ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥的密钥材料。

```
{
  "key_id" : "0d0466b0-e727-4d9c-b35d-f84bb474a37f"
}
```

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

删除ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥的密钥材料。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;
```

```
public class DeleteImportedKeyMaterialSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        DeleteImportedKeyMaterialRequest request = new DeleteImportedKeyMaterialRequest();
        OperateKeyRequestBody body = new OperateKeyRequestBody();
        body.withKeyId("0d0466b0-e727-4d9c-b35d-f84bb474a37f");
        request.withBody(body);
        try {
            DeleteImportedKeyMaterialResponse response = client.deleteImportedKeyMaterial(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

删除ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥的密钥材料。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
```

```
        .build()

    try:
        request = DeleteImportedKeyMaterialRequest()
        request.body = OperateKeyRequestBody(
            key_id="0d0466b0-e727-4d9c-b35d-f84bb474a37f"
        )
        response = client.delete_imported_key_material(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

删除ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥的密钥材料。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.DeleteImportedKeyMaterialRequest{}
    request.Body = &model.OperateKeyRequestBody{
        KeyId: "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
    }
    response, err := client.DeleteImportedKeyMaterial(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.4 密钥授权管理

4.1.4.1 创建授权

功能介绍

- 功能介绍：创建授权，被授权用户可以对授权密钥进行操作。
- 说明：
  - 服务默认主密钥（密钥别名后缀为“/default”）不可以授权。

调用方法

请参见**如何调用API**。

URI

POST /v1.0/{project_id}/kms/create-grant

表 4-274 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-275 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-276 请求 Body 参数

参数	是否必选	参数类型	描述
key_id	是	String	密钥ID，36字节，满足正则匹配“^[0-9a-z]{8}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{12}$”。 例如：0d0466b0-e727-4d9c-b35d-f84bb474a37f。
grantee_principal	是	String	被授权用户ID，1~64字节，满足正则匹配“^[a-zA-Z0-9]{1，64}$”。 例如：0d0466b00d0466b00d0466b00d0466b0

参数	是否必选	参数类型	描述
operations	是	Array of strings	授权允许的操作列表。 有效的值：“create-datakey”，“create-datakey-without-plaintext”，“encrypt-datakey”，“decrypt-datakey”，“describe-key”，“create-grant”，“retire-grant”，“encrypt-data”，“decrypt-data”。 有效值不能仅为“create-grant”。 ● “create-datakey” 创建数据密钥 ● “create-datakey-without-plaintext” 创建不含明文数据密钥 ● “encrypt-datakey” 加密数据密钥 ● “decrypt-datakey” 解密数据密钥 ● “describe-key” 查询密钥信息 ● “retire-grant” 退役授权 ● “encrypt-data” 加密数据 ● “decrypt-data” 解密数据
name	否	String	授权名称，取值1到255字符，满足正则匹配“^[a-zA-Z0-9:/_-]{1,255}$”。
retiring_principal	否	String	可退役授权的用户ID，1~64字节，满足正则匹配“^[a-zA-Z0-9]{1，64}$”。 例如：0d0466b00d0466b00d0466b00d0466b0
grantee_principal_type	否	String	授权类型。有效值：“user”，“domain”。默认值为“user”。
sequence	否	String	请求消息序列号，36字节序列号。 例如：919c82d4-8046-4722-9094-35c3c6524cff

响应参数

状态码： 200

表 4-277 响应 Body 参数

参数	参数类型	描述
grant_id	String	授权ID，64字节。

状态码： 400

表 4-278 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-279 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-280 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-281 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-282 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-283 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-284 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-285 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-286 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-287 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-288 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-289 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-290 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-291 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

授权给用户"13gg44z4g2sglzk0egw0u726zoyzvrs8"ID为 "0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥操作权限，授权操作为查询密钥、创建数据密钥、加密数据密钥。

```
{
  "key_id" : "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
  "operations" : [ "describe-key", "create-datakey", "encrypt-datakey" ],
  "grantee_principal" : "13gg44z4g2sglzk0egw0u726zoyzvrs8",
  "grantee_principal_type" : "user",
  "retiring_principal" : "13gg44z4g2sglzk0egw0u726zoyzvrs8"
}
```

响应示例

状态码： 200

请求已成功

```
{
  "grant_id" : "7c9a3286af4fcca5f0a385ad13e1d21a50e27b6dbcab50f37f30f93b8939827d"
}
```

SDK 代码示例

SDK代码示例如下。

Java

授权给用户"13gg44z4g2sglzk0egw0u726zoyzvrs8"ID为 "0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥操作权限，授权操作为查询密钥、创建数据密钥、加密数据密钥。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;

import java.util.List;
import java.util.ArrayList;

public class CreateGrantSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        CreateGrantRequest request = new CreateGrantRequest();
        CreateGrantRequestBody body = new CreateGrantRequestBody();
        List<String> listbodyOperations = new ArrayList<>();
        listbodyOperations.add("describe-key");
        listbodyOperations.add("create-datakey");
        listbodyOperations.add("encrypt-datakey");

body.withGranteePrincipalType(CreateGrantRequestBody.GranteePrincipalTypeEnum.fromValue("user"));
        body.withRetiringPrincipal("13gg44z4g2sglzk0egw0u726zoyzvrs8");
        body.withOperations(listbodyOperations);
        body.withGranteePrincipal("13gg44z4g2sglzk0egw0u726zoyzvrs8");
        body.withKeyId("0d0466b0-e727-4d9c-b35d-f84bb474a37f");
        request.withBody(body);
        try {
            CreateGrantResponse response = client.createGrant(request);
            System.out.println(response.toString());
```

```
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

授权给用户"13gg44z4g2sglzk0egw0u726zoyzvrs8"ID为 "0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥操作权限，授权操作为查询密钥、创建数据密钥、加密数据密钥。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = CreateGrantRequest()
        listOperationsbody = [
            "describe-key",
            "create-datakey",
            "encrypt-datakey"
        ]
        request.body = CreateGrantRequestBody(
            grantee_principal_type="user",
            retiring_principal="13gg44z4g2sglzk0egw0u726zoyzvrs8",
            operations=listOperationsbody,
            grantee_principal="13gg44z4g2sglzk0egw0u726zoyzvrs8",
            key_id="0d0466b0-e727-4d9c-b35d-f84bb474a37f"
        )
        response = client.create_grant(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

授权给用户"13gg44z4g2sglzk0egw0u726zoyzvrs8"ID为 "0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥操作权限，授权操作为查询密钥、创建数据密钥、加密数据密钥。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.CreateGrantRequest{}
    var listOperationsbody = []string{
        "describe-key",
        "create-datakey",
        "encrypt-datakey",
    }
    granteePrincipalTypeCreateGrantRequestBody:= model.GetCreateGrantRequestBodyGranteePrincipalTypeEnum().USER
    retiringPrincipalCreateGrantRequestBody:= "13gg44z4g2sglzk0egw0u726zoyzvrs8"
    request.Body = &model.CreateGrantRequestBody{
        GranteePrincipalType: &granteePrincipalTypeCreateGrantRequestBody,
        RetiringPrincipal: &retiringPrincipalCreateGrantRequestBody,
        Operations: listOperationsbody,
        GranteePrincipal: "13gg44z4g2sglzk0egw0u726zoyzvrs8",
        KeyId: "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
    }
    response, err := client.CreateGrant(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.4.2 撤销授权

功能介绍

- 功能介绍：撤销授权，授权用户撤销被授权用户操作密钥的权限。
- 说明：
  - 创建密钥的用户才能撤销该密钥授权。

调用方法

请参见**如何调用API**。

URI

POST /v1.0/{project_id}/kms/revoke-grant

表 4-292 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-293 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-294 请求 Body 参数

参数	是否必选	参数类型	描述
key_id	是	String	密钥ID，36字节，满足正则匹配“^[0-9a-z]{8}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{12}$”。 例如：0d0466b0-e727-4d9c-b35d-f84bb474a37f。
grant_id	是	String	授权ID，64字节，满足正则匹配“^[A-Fa-f0-9]{64}$”。 例如：7c9a3286af4fcca5f0a385ad13e1d21a50e27b6dbcab50f37f30f93b8939827d
sequence	否	String	请求消息序列号，36字节序列号。例如：919c82d4-8046-4722-9094-35c3c6524cff

响应参数

状态码： 400

表 4-295 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-296 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-297 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-298 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-299 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-300 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-301 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-302 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-303 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-304 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-305 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-306 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-307 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-308 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

撤销ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥中ID为“7c9a3286af4fcca5f0a385ad13e1d21a50e27b6dbcab50f37f30f93b8939827d”的授权。

```
{
  "key_id" : "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
  "grant_id" : "7c9a3286af4fcca5f0a385ad13e1d21a50e27b6dbcab50f37f30f93b8939827d"
}
```

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

撤销ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥中ID为“7c9a3286af4fcca5f0a385ad13e1d21a50e27b6dbcab50f37f30f93b8939827d”的授权。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;


public class CancelGrantSolution {
```

```
    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        CancelGrantRequest request = new CancelGrantRequest();
        RevokeGrantRequestBody body = new RevokeGrantRequestBody();
        body.withGrantId("7c9a3286af4fcca5f0a385ad13e1d21a50e27b6dbcab50f37f30f93b8939827d");
        body.withKeyId("0d0466b0-e727-4d9c-b35d-f84bb474a37f");
        request.withBody(body);
        try {
            CancelGrantResponse response = client.cancelGrant(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

撤销ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥中ID为“7c9a3286af4fcca5f0a385ad13e1d21a50e27b6dbcab50f37f30f93b8939827d”的授权。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
```

```
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = CancelGrantRequest()
        request.body = RevokeGrantRequestBody(
            grant_id="7c9a3286af4fcca5f0a385ad13e1d21a50e27b6dbcab50f37f30f93b8939827d",
            key_id="0d0466b0-e727-4d9c-b35d-f84bb474a37f"
        )
        response = client.cancel_grant(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

撤销ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥中ID为“7c9a3286af4fcca5f0a385ad13e1d21a50e27b6dbcab50f37f30f93b8939827d”的授权。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.CancelGrantRequest{}
    request.Body = &model.RevokeGrantRequestBody{
        GrantId: "7c9a3286af4fcca5f0a385ad13e1d21a50e27b6dbcab50f37f30f93b8939827d",
        KeyId: "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
    }
    response, err := client.CancelGrant(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.4.3 退役授权

功能介绍

- 功能介绍：退役授权，表示被授权用户不再具有授权密钥的操作权。
  例如：用户A授权用户B可以操作密钥A/key，同时授权用户C可以撤销该授权，那么用户A、B、C均可退役该授权，退役授权后，用户B不再可以使用A/key。
- 须知：
  可执行退役授权的主体包括：
  - 创建授权的用户；
  - 授权中retiring_principal指向的用户；
  - 当授权的操作列表中包含retire-grant时，grantee_principal指向的用户。

调用方法

请参见**如何调用API**。

URI

POST /v1.0/{project_id}/kms/retire-grant

表 4-309 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-310 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-311 请求 Body 参数

参数	是否必选	参数类型	描述
key_id	是	String	密钥ID，36字节，满足正则匹配“^[0-9a-z]{8}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{12}$”。 例如：0d0466b0-e727-4d9c-b35d-f84bb474a37f。
grant_id	是	String	授权ID，64字节，满足正则匹配“^[A-Fa-f0-9]{64}$”。 例如：7c9a3286af4fcca5f0a385ad13e1d21a50e27b6dbcab50f37f30f93b8939827d
sequence	否	String	请求消息序列号，36字节序列号。例如：919c82d4-8046-4722-9094-35c3c6524cff

响应参数

状态码： 400

表 4-312 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-313 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-314 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-315 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-316 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-317 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-318 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-319 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-320 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-321 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-322 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-323 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码

参数	参数类型	描述
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-324 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-325 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

删除ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥中ID为”7c9a3286af4fcca5f0a385ad13e1d21a50e27b6dbcab50f37f30f93b8939827d“的授权。

```
{
  "key_id" : "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
  "grant_id" : "7c9a3286af4fcca5f0a385ad13e1d21a50e27b6dbcab50f37f30f93b8939827d"
}
```

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

删除ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥中ID为”7c9a3286af4fcca5f0a385ad13e1d21a50e27b6dbcab50f37f30f93b8939827d“的授权。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
```

```
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;


public class CancelSelfGrantSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        CancelSelfGrantRequest request = new CancelSelfGrantRequest();
        RevokeGrantRequestBody body = new RevokeGrantRequestBody();
        body.withGrantId("7c9a3286af4fcca5f0a385ad13e1d21a50e27b6dbcab50f37f30f93b8939827d");
        body.withKeyId("0d0466b0-e727-4d9c-b35d-f84bb474a37f");
        request.withBody(body);
        try {
            CancelSelfGrantResponse response = client.cancelSelfGrant(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

删除ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥中ID为”7c9a3286af4fcca5f0a385ad13e1d21a50e27b6dbcab50f37f30f93b8939827d“的授权。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this
```

```
example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = CancelSelfGrantRequest()
        request.body = RevokeGrantRequestBody(
            grant_id="7c9a3286af4fcca5f0a385ad13e1d21a50e27b6dbcab50f37f30f93b8939827d",
            key_id="0d0466b0-e727-4d9c-b35d-f84bb474a37f"
        )
        response = client.cancel_self_grant(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

删除ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥中ID为”7c9a3286af4fcca5f0a385ad13e1d21a50e27b6dbcab50f37f30f93b8939827d“的授权。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.CancelSelfGrantRequest{}
    request.Body = &model.RevokeGrantRequestBody{
        GrantId: "7c9a3286af4fcca5f0a385ad13e1d21a50e27b6dbcab50f37f30f93b8939827d",
        KeyId: "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
```

```
    }
    response, err := client.CancelSelfGrant(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.4.4 查询授权列表

功能介绍

- 功能介绍：查询密钥的授权列表。

调用方法

请参见**如何调用API**。

URI

POST /v1.0/{project_id}/kms/list-grants

表 4-326 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-327 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-328 请求 Body 参数

参数	是否必选	参数类型	描述
key_id	是	String	密钥ID，36字节，满足正则匹配“^[0-9a-z]{8}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{12}$”。 例如：0d0466b0-e727-4d9c-b35d-f84bb474a37f。
limit	否	String	指定查询返回记录条数，默认值为1000。 取值范围：授权最大个数范围以内，例如：100 如果指定查询记录条数小于存在的条数，响应参数“truncated”将返回“true”，表示存在分页。
marker	否	String	分页查询起始位置标识。 分页查询收到的响应参数“truncated”为“true”时，可以发送连续的请求获取更多的记录条数，“marker”设置为响应的“next_marker”的值。例如：10。

参数	是否必选	参数类型	描述
sequence	否	String	请求消息序列号，36字节序列号。 例如：919c82d4-8046-4722-9094-35c3c6524cff

响应参数

状态码： 200

表 4-329 响应 Body 参数

参数	参数类型	描述
grants	Array of **Grants** objects	grant列表，详情请参见grants字段数据结构说明。
next_marker	String	获取下一页所需要传递的marker值。 当“truncated”为“false”时，“next_marker”为空。
truncated	String	是否还有下一页： ● “true”表示还有数据。 ● “false”表示已经是最后一页。
total	Integer	grant总条数。

表 4-330 Grants

参数	参数类型	描述
key_id	String	密钥ID。
grant_id	String	授权ID，64字节。
grantee_principal	String	被授权用户ID，1~64字节，满足正则匹配“^[a-zA-Z0-9]{1，64}$”。 例如：0d0466b00d0466b00d0466b00d0466b0
grantee_principal_type	String	授权类型。 有效值：“user”，“domain”。

参数	参数类型	描述
operations	Array of strings	授权允许的操作列表。 有效的值：“create-datakey”，“create-datakey-without-plaintext”，“encrypt-datakey”，“decrypt-datakey”，“describe-key”，“create-grant”，“retire-grant”，“encrypt-data”，“decrypt-data”。 有效值不能仅为“create-grant”。 ● “create-datakey” 创建数据密钥 ● “create-datakey-without-plaintext” 创建不含明文数据密钥 ● “encrypt-datakey” 加密数据密钥 ● “decrypt-datakey” 解密数据密钥 ● “describe-key” 查询密钥信息 ● “retire-grant” 退役授权 ● “encrypt-data” 加密数据 ● “decrypt-data” 解密数据
issuing_principal	String	创建授权用户ID，1~64字节，满足正则匹配“^[a-zA-Z0-9]{1，64}$”。 例如：0d0466b00d0466b00d0466b00d0466b0
creation_date	String	创建时间，时间戳，即从1970年1月1日至该时间的总秒数。 例如：1497341531000
name	String	授权名字，取值1到255字符，满足正则匹配“^[a-zA-Z0-9:/_-]{1,255}$”。
retiring_principal	String	可退役授权的用户ID，1~64字节，满足正则匹配“^[a-zA-Z0-9]{1，64}$”。 例如：0d0466b00d0466b00d0466b00d0466b0

状态码： 400

表 4-331 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-332 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-333 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-334 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-335 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-336 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-337 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-338 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-339 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-340 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-341 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-342 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-343 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-344 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

查询ID为 "0d0466b0-e727-4d9c-b35d-f84bb474a37f"当前的授权列表信息。

```
{
  "key_id" : "0d0466b0-e727-4d9c-b35d-f84bb474a37f"
}
```

响应示例

状态码： 200

请求已成功

```
{
  "grants" : [ {
    "operations" : [ "create-datakey", "describe-key" ],
    "issuing_principal" : "8b961fb414344d59825ba0c8c008c815",
    "key_id" : "737fd52b-36c4-4c91-972e-f6e202de9f6e",
    "grant_id" : "dd3f03e9229a5e47a41be6c27a630e60d5cbdbad2be89465d63109ad034db7d8",
    "grantee_principal" : "13gg44z4g2sglzk0egw0u726zoyzvrs8",
    "name" : "13gg44z4g2sglzk0egw0u726zoyzvrs8",
    "creation_date" : "1597062260000",
    "grantee_principal_type" : "user"
  } ],
  "next_marker" : "",
  "total" : 1,
  "truncated" : "false"
}
```

SDK 代码示例

SDK代码示例如下。

Java

查询ID为 "0d0466b0-e727-4d9c-b35d-f84bb474a37f"当前的授权列表信息。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
```

```
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;


public class ListGrantsSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        ListGrantsRequest request = new ListGrantsRequest();
        ListGrantsRequestBody body = new ListGrantsRequestBody();
        body.withKeyId("0d0466b0-e727-4d9c-b35d-f84bb474a37f");
        request.withBody(body);
        try {
            ListGrantsResponse response = client.listGrants(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

查询ID为 "0d0466b0-e727-4d9c-b35d-f84bb474a37f"当前的授权列表信息。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
```

```
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ListGrantsRequest()
        request.body = ListGrantsRequestBody(
            key_id="0d0466b0-e727-4d9c-b35d-f84bb474a37f"
        )
        response = client.list_grants(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

查询ID为 "0d0466b0-e727-4d9c-b35d-f84bb474a37f"当前的授权列表信息。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ListGrantsRequest{}
    request.Body = &model.ListGrantsRequestBody{
        KeyId: "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
    }
    response, err := client.ListGrants(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
```

```
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.4.5 查询可退役授权列表

功能介绍

- 功能介绍：查询用户可以退役的授权列表。

调用方法

请参见**如何调用API**。

URI

POST /v1.0/{project_id}/kms/list-retirable-grants

表 4-345 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-346 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-347 请求 Body 参数

参数	是否必选	参数类型	描述
limit	否	String	指定查询可退役授权返回记录条数，如果查询记录条数小于存在的条数，响应参数“truncated”将返回“true”，表示存在分页。 取值在授权最大个数范围以内。例如：100
marker	否	String	分页查询起始位置标识。 分页查询收到的响应参数“truncated”为“true”时，可以发送连续的请求获取更多的记录条数，“marker”设置为响应的“next_marker”的值。例如：10。
sequence	否	String	请求消息序列号，36字节序列号。 例如：919c82d4-8046-4722-9094-35c3c6524cff

响应参数

状态码： 200

表 4-348 响应 Body 参数

参数	参数类型	描述
grants	Array of **Grants** objects	grant列表，详情请参见grants字段数据结构说明。

参数	参数类型	描述
next_marker	String	获取下一页所需要传递的marker值。 当“truncated”为“false”时，“next_marker”为空。
total	Integer	可退役授权总条数。
truncated	String	是否还有下一页： ● “true”表示还有数据。 ● “false”表示已经是最后一页。

表 4-349 Grants

参数	参数类型	描述
key_id	String	密钥ID。
grant_id	String	授权ID，64字节。
grantee_principal	String	被授权用户ID，1~64字节，满足正则匹配“^[a-zA-Z0-9]{1，64}$”。 例如：0d0466b00d0466b00d0466b00d0466b0
grantee_principal_type	String	授权类型。 有效值：“user”，“domain”。
operations	Array of strings	授权允许的操作列表。 有效的值：“create-datakey”，“create-datakey-without-plaintext”，“encrypt-datakey”，“decrypt-datakey”，“describe-key”，“create-grant”，“retire-grant”，“encrypt-data”，“decrypt-data”。 有效值不能仅为“create-grant”。 ● “create-datakey” 创建数据密钥 ● “create-datakey-without-plaintext” 创建不含明文数据密钥 ● “encrypt-datakey” 加密数据密钥 ● “decrypt-datakey” 解密数据密钥 ● “describe-key” 查询密钥信息 ● “retire-grant” 退役授权 ● “encrypt-data” 加密数据 ● “decrypt-data” 解密数据
issuing_principal	String	创建授权用户ID，1~64字节，满足正则匹配“^[a-zA-Z0-9]{1，64}$”。 例如：0d0466b00d0466b00d0466b00d0466b0

参数	参数类型	描述
creation_date	String	创建时间，时间戳，即从1970年1月1日至该时间的总秒数。 例如：1497341531000
name	String	授权名字，取值1到255字符，满足正则匹配“^[a-zA-Z0-9:/_-]{1,255}$”。
retiring_principal	String	可退役授权的用户ID，1~64字节，满足正则匹配“^[a-zA-Z0-9]{1，64}$”。 例如：0d0466b00d0466b00d0466b00d0466b0

状态码： 400

表 4-350 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-351 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-352 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-353 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-354 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-355 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-356 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-357 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-358 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-359 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码

参数	参数类型	描述
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-360 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-361 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-362 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-363 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

查询至多1000条当前项目可退役的授权列表。

```
{
  "limit" : "1000"
}
```

响应示例

状态码： 200

请求已成功

```
{
  "grants" : [ {
    "operations" : [ "create-datakey", "describe-key" ],
    "issuing_principal" : "8b961fb414344d59825ba0c8c008c815",
    "key_id" : "737fd52b-36c4-4c91-972e-f6e202de9f6e",
    "grant_id" : "dd3f03e9229a5e47a41be6c27a630e60d5cbdbad2be89465d63109ad034db7d8",
    "grantee_principal" : "13gg44z4g2sglzk0egw0u726zoyzvrs8",
    "name" : "13gg44z4g2sglzk0egw0u726zoyzvrs8",
    "creation_date" : "1597062260000",
    "grantee_principal_type" : "user"
  } ],
  "next_marker" : "",
  "total" : 1,
  "truncated" : "false"
}
```

SDK 代码示例

SDK代码示例如下。

Java

查询至多1000条当前项目可退役的授权列表。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;


public class ListRetirableGrantsSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        ListRetirableGrantsRequest request = new ListRetirableGrantsRequest();
        ListRetirableGrantsRequestBody body = new ListRetirableGrantsRequestBody();
        body.withLimit("1000");
        request.withBody(body);
```

```
        try {
            ListRetirableGrantsResponse response = client.listRetirableGrants(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

查询至多1000条当前项目可退役的授权列表。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ListRetirableGrantsRequest()
        request.body = ListRetirableGrantsRequestBody(
            limit="1000"
        )
        response = client.list_retirable_grants(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

查询至多1000条当前项目可退役的授权列表。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
```

```
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ListRetirableGrantsRequest{}
    limitListRetirableGrantsRequestBody:= "1000"
    request.Body = &model.ListRetirableGrantsRequestBody{
        Limit: &limitListRetirableGrantsRequestBody,
    }
    response, err := client.ListRetirableGrants(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.5 小数据加解密

4.1.5.1 加密数据

功能介绍

- 功能介绍：加密数据，用指定的用户主密钥加密数据。

接口约束

- 使用非对称密钥加密数据时，请记录选择的密钥ID和加密算法。解密数据时，您将需要提供相同的密钥ID和加密算法。如果指定的密钥和加密算法与用于加密数据的值不匹配，解密操作将失败。
- 使用对称密钥解密时，不需要提供密钥ID和加密算法，KMS会将此信息存储在密文中。KMS无法将元数据存储在使用非对称密钥生成的密文中，非对称密钥密文的标准格式不包括可配置字段。

调用方法

请参见**如何调用API**。

URI

POST /v1.0/{project_id}/kms/encrypt-data

表 4-364 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-365 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-366 请求 Body 参数

参数	是否必选	参数类型	描述
key_id	是	String	密钥ID，36字节，满足正则匹配“^[0-9a-z]{8}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{12}$”。 例如：0d0466b0-e727-4d9c-b35d-f84bb474a37f。
plain_text	是	String	明文数据，1~4096字节，满足正则匹配“^.{1,4096}$”，且转化为byte数组后长度取值范围为1~4096字节。
encryption_algorithm	否	String	数据加密算法，仅使用非对称密钥需要指定该参数，默认值为“SYMMETRIC_DEFAULT”，合法枚举值如下： ● SYMMETRIC_DEFAULT ● RSAES_OAEP_SHA_256 ● SM2_ENCRYPT
additional_authenticated_data	否	String	身份验证的非敏感额外数据。任意字符串，长度不超过128字节。
sequence	否	String	请求消息序列号，36字节序列号。 例如：919c82d4-8046-4722-9094-35c3c6524cff

响应参数

状态码： 200

表 4-367 响应 Body 参数

参数	参数类型	描述
key_id	String	密钥ID。
cipher_text	String	加密后的密文，使用了Base64编码

状态码： 400

表 4-368 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-369 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-370 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-371 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-372 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-373 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-374 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-375 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-376 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-377 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-378 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-379 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码

参数	参数类型	描述
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-380 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-381 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

使用ID为“0d0466b0-e727-4d9c-b35d-f84bb474a37f”的用户主密钥加密明文数据“hello world”，数据加密算法为"SYMMETRIC_DEFAULT"，附加数据为“123aad”。

```
{
  "key_id" : "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
  "plain_text" : "hello world",
  "encryption_algorithm" : "SYMMETRIC_DEFAULT",
  "additional_authenticated_data" : "123aad"
}
```

响应示例

状态码： 200

请求已成功

```
{
  "key_id" : "bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e",
  "cipher_text" : "AgDoAG7EsEc2OHpQxz4gDFDH54CqwaelpTdEl+RFXXX..."
}
```

SDK 代码示例

SDK代码示例如下。

Java

使用ID为“0d0466b0-e727-4d9c-b35d-f84bb474a37f”的用户主密钥加密明文数据“hello world”，数据加密算法为"SYMMETRIC_DEFAULT"，附加数据为“123aad”。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;


public class EncryptDataSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        EncryptDataRequest request = new EncryptDataRequest();
        EncryptDataRequestBody body = new EncryptDataRequestBody();
        body.withAdditionalAuthenticatedData("123aad");

body.withEncryptionAlgorithm(EncryptDataRequestBody.EncryptionAlgorithmEnum.fromValue("SYMMETRIC_DEFAULT"));
        body.withPlainText("hello world");
        body.withKeyId("0d0466b0-e727-4d9c-b35d-f84bb474a37f");
        request.withBody(body);
        try {
            EncryptDataResponse response = client.encryptData(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

使用ID为“0d0466b0-e727-4d9c-b35d-f84bb474a37f”的用户主密钥加密明文数据“hello world”，数据加密算法为"SYMMETRIC_DEFAULT"，附加数据为“123aad”。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = EncryptDataRequest()
        request.body = EncryptDataRequestBody(
            additional_authenticated_data="123aad",
            encryption_algorithm="SYMMETRIC_DEFAULT",
            plain_text="hello world",
            key_id="0d0466b0-e727-4d9c-b35d-f84bb474a37f"
        )
        response = client.encrypt_data(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

使用ID为“0d0466b0-e727-4d9c-b35d-f84bb474a37f”的用户主密钥加密明文数据“hello world”，数据加密算法为"SYMMETRIC_DEFAULT"，附加数据为“123aad”。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
```

```
    // In this example, AK and SK are stored in environment variables for authentication. Before running this
example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.EncryptDataRequest{}
    additionalAuthenticatedDataEncryptDataRequestBody:= "123aad"
    encryptionAlgorithmEncryptDataRequestBody:=
model.GetEncryptDataRequestBodyEncryptionAlgorithmEnum().SYMMETRIC_DEFAULT
    request.Body = &model.EncryptDataRequestBody{
        AdditionalAuthenticatedData: &additionalAuthenticatedDataEncryptDataRequestBody,
        EncryptionAlgorithm: &encryptionAlgorithmEncryptDataRequestBody,
        PlainText: "hello world",
        KeyId: "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
    }
    response, err := client.EncryptData(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.5.2 解密数据

功能介绍

- 功能介绍：解密数据。

接口约束

- 解密非对称密钥加密的数据时，需要指定密钥ID和加密算法。如果指定的密钥和加密算法与用于加密数据的值不匹配，解密操作将失败。

调用方法

请参见**如何调用API**。

URI

POST /v1.0/{project_id}/kms/decrypt-data

表 4-382 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-383 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-384 请求 Body 参数

参数	是否必选	参数类型	描述
cipher_text	是	String	被加密数据密文。取值为加密数据结果中的cipher_text的值，满足正则匹配“^[0-9a-zA-Z+/=]{128,5648}$”。

参数	是否必选	参数类型	描述
encryption_algorithm	否	String	数据加密算法，仅使用非对称密钥需要指定该参数，默认值为“SYMMETRIC_DEFAULT”，合法枚举值如下： ● SYMMETRIC_DEFAULT ● RSAES_OAEP_SHA_256 ● SM2_ENCRYPT
key_id	否	String	密钥ID，36字节，满足正则匹配“^[0-9a-f]{8}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{12}$”。仅当密文使用非对称密钥加密时才需要此参数。 例如：0d0466b0-e727-4d9c-b35d-f84bb474a37f。
additional_authenticated_data	否	String	身份验证的非敏感额外数据。任意字符串，长度不超过128字节。
sequence	否	String	请求消息序列号，36字节序列号。 例如：919c82d4-8046-4722-9094-35c3c6524cff

响应参数

状态码： 200

表 4-385 响应 Body 参数

参数	参数类型	描述
key_id	String	密钥ID。
plain_text	String	明文。
plain_text_base64	String	明文的Base64值，在非对称加密场景下，若加密的明文中含有不可见字符，则解密结果以该值为准。

状态码： 400

表 4-386 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-387 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-388 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-389 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-390 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-391 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-392 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-393 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-394 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-395 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-396 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-397 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码

参数	参数类型	描述
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-398 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-399 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

解密密文“AgDoAG7EsEc2OHpQxz4gDFDH54Cqwaelxxxxxxx“，数据加密算法为”SYMMETRIC_DEFAULT“，附加数据为“123aad”。

```
{
  "cipher_text" : "AgDoAG7EsEc2OHpQxz4gDFDH54Cqwaelxxxxxxx",
  "encryption_algorithm" : "SYMMETRIC_DEFAULT",
  "additional_authenticated_data" : "123aad"
}
```

响应示例

状态码： 200

请求已成功

```
{
  "key_id" : "bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e",
  "plain_text" : "hello world",
  "plain_text_base64" : "aGVsbG8gd29ybGQ="
}
```

SDK 代码示例

SDK代码示例如下。

Java

解密密文“AgDoAG7EsEc2OHpQxz4gDFDH54Cqwaelxxxxxxx“，数据加密算法为”SYMMETRIC_DEFAULT“，附加数据为“123aad”。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;


public class DecryptDataSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        DecryptDataRequest request = new DecryptDataRequest();
        DecryptDataRequestBody body = new DecryptDataRequestBody();
        body.withAdditionalAuthenticatedData("123aad");

body.withEncryptionAlgorithm(DecryptDataRequestBody.EncryptionAlgorithmEnum.fromValue("SYMMETRIC_DEFAULT"));
        body.withCipherText("AgDoAG7EsEc2OHpQxz4gDFDH54Cqwaelxxxxxxx");
        request.withBody(body);
        try {
            DecryptDataResponse response = client.decryptData(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

解密密文“AgDoAG7EsEc2OHpQxz4gDFDH54Cqwaelxxxxxxx“，数据加密算法为”SYMMETRIC_DEFAULT“，附加数据为“123aad”。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
```

```
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = DecryptDataRequest()
        request.body = DecryptDataRequestBody(
            additional_authenticated_data="123aad",
            encryption_algorithm="SYMMETRIC_DEFAULT",
            cipher_text="AgDoAG7EsEc2OHpQxz4gDFDH54Cqwaelxxxxxxx"
        )
        response = client.decrypt_data(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

解密密文“AgDoAG7EsEc2OHpQxz4gDFDH54Cqwaelxxxxxxx“，数据加密算法为”SYMMETRIC_DEFAULT“，附加数据为“123aad”。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
```

```
        Build())

    request := &model.DecryptDataRequest{}
    additionalAuthenticatedDataDecryptDataRequestBody:= "123aad"
    encryptionAlgorithmDecryptDataRequestBody:=
model.GetDecryptDataRequestBodyEncryptionAlgorithmEnum().SYMMETRIC_DEFAULT
    request.Body = &model.DecryptDataRequestBody{
        AdditionalAuthenticatedData: &additionalAuthenticatedDataDecryptDataRequestBody,
        EncryptionAlgorithm: &encryptionAlgorithmDecryptDataRequestBody,
        CipherText: "AgDoAG7EsEc2OHpQxz4gDFDH54Cqwaelxxxxxxx",
    }
    response, err := client.DecryptData(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.6 签名验签

4.1.6.1 签名数据

功能介绍

- 功能介绍：使用非对称密钥的私钥对消息或消息摘要进行数字签名。

接口约束

- 仅支持key_usage为SIGN_VERIFY的非对称密钥进行签名操作。
- 使用SM2密钥签名时，仅支持对消息摘要签名。根据GBT32918国家标准，计算SM2签名值时，消息摘要不是对原始消息直接计算SM3摘要，而是对Z(A)和M的拼接值计算的摘要，其中M是待签名的原始消息，Z(A)是GBT32918中定义的用户A的杂凑值。

调用方法

请参见**如何调用API**。

URI

POST /v1.0/{project_id}/kms/sign

表 4-400 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-401 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-402 请求 Body 参数

参数	是否必选	参数类型	描述
key_id	是	String	密钥ID，36字节，满足正则匹配“^[0-9a-f]{8}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{12}$”。 例如：0d0466b0-e727-4d9c-b35d-f84bb474a37f。
message	是	String	待签名的消息摘要或者消息，消息长度要求小于4096字节，使用Base64编码。

参数	是否必选	参数类型	描述
signing_algorithm	是	String	签名算法，枚举如下： ● RSASSA_PSS_SHA_256 ● RSASSA_PSS_SHA_384 ● RSASSA_PSS_SHA_512 ● RSASSA_PKCS1_V1_5_SHA_256 ● RSASSA_PKCS1_V1_5_SHA_384 ● RSASSA_PKCS1_V1_5_SHA_512 ● ECDSA_SHA_256 ● ECDSA_SHA_384 ● ECDSA_SHA_512 ● SM2DSA_SM3
message_type	否	String	消息类型，默认为“DIGEST”，枚举如下： ● DIGEST 表示消息摘要 ● RAW 表示消息原文
sequence	否	String	请求消息序列号，36字节序列号。 例如：919c82d4-8046-4722-9094-35c3c6524cff。

响应参数

状态码： 200

表 4-403 响应 Body 参数

参数	参数类型	描述
key_id	String	密钥ID。
signature	String	签名值，使用base64编码。

状态码： 400

表 4-404 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-405 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-406 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-407 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-408 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-409 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-410 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-411 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-412 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-413 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-414 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-415 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码

参数	参数类型	描述
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-416 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-417 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

签名消息信息，使用ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥，采用"RSASSA_PKCS1_V1_5_SHA_256"签名算法，对摘要消息进行签名。

```
{
  "key_id" : "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
  "signing_algorithm" : "RSASSA_PKCS1_V1_5_SHA_256",
  "message" : "MmFiZWE0ZjI3ZGIxYTkzY2RmYmEzM2YwMTA1YmJjYw=="
}
```

响应示例

状态码： 200

请求已成功

```
{
  "key_id" : "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
  "signature" : "jFUqQESGBc0j6k9BozzrP9YL4qk8/W9DZRvK6XXX..."
}
```

SDK 代码示例

SDK代码示例如下。

Java

签名消息信息，使用ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥，采用"RSASSA_PKCS1_V1_5_SHA_256"签名算法，对摘要消息进行签名。

```
package com.huaweicloud.sdk.test;
```

```
import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;


public class SignSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        SignRequest request = new SignRequest();
        SignRequestBody body = new SignRequestBody();
        body.withSigningAlgorithm(SignRequestBody.SigningAlgorithmEnum.fromValue("RSASSA_PKCS1_V1_5_SHA_256"));
        body.withMessage("MmFiZWE0ZjI3ZGIxYTkzY2RmYmEzM2YwMTA1YmJjYw==");
        body.withKeyId("0d0466b0-e727-4d9c-b35d-f84bb474a37f");
        request.withBody(body);
        try {
            SignResponse response = client.sign(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

签名消息信息，使用ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥，采用"RSASSA_PKCS1_V1_5_SHA_256"签名算法，对摘要消息进行签名。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *
```

```
if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = SignRequest()
        request.body = SignRequestBody(
            signing_algorithm="RSASSA_PKCS1_V1_5_SHA_256",
            message="MmFiZWE0ZjI3ZGIxYTkzY2RmYmEzM2YwMTA1YmJjYw==",
            key_id="0d0466b0-e727-4d9c-b35d-f84bb474a37f"
        )
        response = client.sign(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

签名消息信息，使用ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥，采用"RSASSA_PKCS1_V1_5_SHA_256"签名算法，对摘要消息进行签名。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())
```

```
    request := &model.SignRequest{}
    request.Body = &model.SignRequestBody{
        SigningAlgorithm: model.GetSignRequestBodySigningAlgorithmEnum().RSASSA_PKCS1_V1_5_SHA_256,
        Message: "MmFiZWE0ZjI3ZGIxYTkzY2RmYmEzM2YwMTA1YmJjYw==",
        KeyId: "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
    }
    response, err := client.Sign(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.6.2 验证签名

功能介绍

- 功能介绍：使用非对称密钥的公钥对消息或消息摘要进行签名验证。

接口约束

- 仅支持key_usage为SIGN_VERIFY的非对称密钥进行验签操作。
- 使用SM2密钥签名时，仅支持对消息摘要签名。根据GBT32918国家标准，计算SM2签名值时，消息摘要不是对原始消息直接计算SM3摘要，而是对Z(A)和M的拼接值计算的摘要，其中M是待签名的原始消息，Z(A)是GBT32918中定义的用户A的杂凑值。

调用方法

请参见**如何调用API**。

URI

POST /v1.0/{project_id}/kms/verify

表 4-418 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-419 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-420 请求 Body 参数

参数	是否必选	参数类型	描述
key_id	是	String	密钥ID，36字节，满足正则匹配“^[0-9a-f]{8}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{12}$”。 例如：0d0466b0-e727-4d9c-b35d-f84bb474a37f。
message	是	String	待签名的消息摘要或者消息，消息长度要求小于4096字节，使用Base64编码。
signature	是	String	待验证的签名值，使用Base64编码。

参数	是否必选	参数类型	描述
signing_algorithm	是	String	签名算法，枚举如下： ● RSASSA_PSS_SHA_256 ● RSASSA_PSS_SHA_384 ● RSASSA_PSS_SHA_512 ● RSASSA_PKCS1_V1_5_SHA_256 ● RSASSA_PKCS1_V1_5_SHA_384 ● RSASSA_PKCS1_V1_5_SHA_512 ● ECDSA_SHA_256 ● ECDSA_SHA_384 ● ECDSA_SHA_512 ● SM2DSA_SM3
message_type	否	String	消息类型，默认为“DIGEST”，枚举如下： ● DIGEST 表示消息摘要 ● RAW 表示消息原文
sequence	否	String	请求消息序列号，36字节序列号。 例如：919c82d4-8046-4722-9094-35c3c6524cff。

响应参数

状态码： 200

表 4-421 响应 Body 参数

参数	参数类型	描述
key_id	String	密钥ID。
signature_valid	String	签名验证合法性，“true”表示验证签名合法，“false”表示验证签名非法。

状态码： 400

表 4-422 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-423 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-424 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-425 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-426 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-427 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-428 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-429 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-430 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-431 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-432 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-433 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码

参数	参数类型	描述
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-434 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-435 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

验签消息信息，使用ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥，采用"RSASSA_PKCS1_V1_5_SHA_256"签名算法，对摘要消息和签名信息进行验签。

```
{
  "key_id" : "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
  "signing_algorithm" : "RSASSA_PKCS1_V1_5_SHA_256",
  "signature" : "jFUqQESGBc0j6k9BozzrP9YL4qk8/W9DZRvK6XXX...",
  "message" : "MmFiZWE0ZjI3ZGIxYTkzY2RmYmEzM2YwMTA1YmJjYw=="
}
```

响应示例

状态码： 200

请求已成功

```
{
  "key_id" : "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
  "signature_valid" : "true"
}
```

SDK 代码示例

SDK代码示例如下。

Java

验签消息信息，使用ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥，采用"RSASSA_PKCS1_V1_5_SHA_256"签名算法，对摘要消息和签名信息进行验签。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;


public class ValidateSignatureSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        ValidateSignatureRequest request = new ValidateSignatureRequest();
        VerifyRequestBody body = new VerifyRequestBody();

body.withSigningAlgorithm(VerifyRequestBody.SigningAlgorithmEnum.fromValue("RSASSA_PKCS1_V1_5_SHA_256"));
        body.withSignature("jFUqQESGBc0j6k9BozzrP9YL4qk8/W9DZRvK6XXX...");
        body.withMessage("MmFiZWE0ZjI3ZGIxYTkzY2RmYmEzM2YwMTA1YmJjYw==");
        body.withKeyId("0d0466b0-e727-4d9c-b35d-f84bb474a37f");
        request.withBody(body);
        try {
            ValidateSignatureResponse response = client.validateSignature(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

验签消息信息，使用ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥，采用"RSASSA_PKCS1_V1_5_SHA_256"签名算法，对摘要消息和签名信息进行验签。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
```

```
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ValidateSignatureRequest()
        request.body = VerifyRequestBody(
            signing_algorithm="RSASSA_PKCS1_V1_5_SHA_256",
            signature="jFUqQESGBc0j6k9BozzrP9YL4qk8/W9DZRvK6XXX...",
            message="MmFiZWE0ZjI3ZGIxYTkzY2RmYmEzM2YwMTA1YmJjYw==",
            key_id="0d0466b0-e727-4d9c-b35d-f84bb474a37f"
        )
        response = client.validate_signature(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

验签消息信息，使用ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥，采用"RSASSA_PKCS1_V1_5_SHA_256"签名算法，对摘要消息和签名信息进行验签。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
```

```
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ValidateSignatureRequest{}
    request.Body = &model.VerifyRequestBody{
        SigningAlgorithm:
model.GetVerifyRequestBodySigningAlgorithmEnum().RSASSA_PKCS1_V1_5_SHA_256,
        Signature: "jFUqQESGBc0j6k9BozzrP9YL4qk8/W9DZRvK6XXX...",
        Message: "MmFiZWE0ZjI3ZGIxYTkzY2RmYmEzM2YwMTA1YmJjYw==",
        KeyId: "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
    }
    response, err := client.ValidateSignature(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.7 密钥轮换管理

4.1.7.1 开启密钥轮换

功能介绍

- 功能介绍：开启用户主密钥轮换。

- 说明：
  - 开启密钥轮换后，默认轮换间隔时间为365天。
  - 默认主密钥及外部导入密钥不支持轮换操作。

调用方法

请参见**如何调用API**。

URI

POST /v1.0/{project_id}/kms/enable-key-rotation

表 4-436 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-437 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-438 请求 Body 参数

参数	是否必选	参数类型	描述
key_id	是	String	密钥ID，36字节，满足正则匹配“^[0-9a-z]{8}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{12}$”。 例如：0d0466b0-e727-4d9c-b35d-f84bb474a37f。
sequence	否	String	请求消息序列号，36字节序列号。 例如：919c82d4-8046-4722-9094-35c3c6524cff

响应参数

状态码： 400

表 4-439 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-440 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-441 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-442 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-443 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-444 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-445 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-446 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-447 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-448 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-449 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-450 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-451 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-452 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

开启ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的密钥轮换。

```
{
  "key_id" : "0d0466b0-e727-4d9c-b35d-f84bb474a37f"
}
```

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;
```

```
import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;


public class EnableKeyRotationSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        EnableKeyRotationRequest request = new EnableKeyRotationRequest();
        OperateKeyRequestBody body = new OperateKeyRequestBody();
        body.withKeyId("0d0466b0-e727-4d9c-b35d-f84bb474a37f");
        request.withBody(body);
        try {
            EnableKeyRotationResponse response = client.enableKeyRotation(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
```

```
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = EnableKeyRotationRequest()
        request.body = OperateKeyRequestBody(
            key_id="0d0466b0-e727-4d9c-b35d-f84bb474a37f"
        )
        response = client.enable_key_rotation(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.EnableKeyRotationRequest{}
    request.Body = &model.OperateKeyRequestBody{
        KeyId: "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
    }
    response, err := client.EnableKeyRotation(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.7.2 关闭密钥轮换

功能介绍

- 功能介绍：关闭用户主密钥轮换。

调用方法

请参见**如何调用API**。

URI

POST /v1.0/{project_id}/kms/disable-key-rotation

表 4-453 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-454 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-455 请求 Body 参数

参数	是否必选	参数类型	描述
key_id	是	String	密钥ID，36字节，满足正则匹配“^[0-9a-z]{8}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{12}$”。 例如：0d0466b0-e727-4d9c-b35d-f84bb474a37f。
sequence	否	String	请求消息序列号，36字节序列号。 例如：919c82d4-8046-4722-9094-35c3c6524cff

响应参数

状态码： 400

表 4-456 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-457 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-458 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-459 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-460 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-461 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-462 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-463 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码

参数	参数类型	描述
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-464 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-465 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-466 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-467 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-468 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-469 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

关闭key_id为0d0466b0-e727-4d9c-b35d-f84bb474a37f的密钥轮换

```
{
  "key_id" : "0d0466b0-e727-4d9c-b35d-f84bb474a37f"
}
```

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

关闭key_id为0d0466b0-e727-4d9c-b35d-f84bb474a37f的密钥轮换

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;


public class DisableKeyRotationSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
```

```
        DisableKeyRotationRequest request = new DisableKeyRotationRequest();
        OperateKeyRequestBody body = new OperateKeyRequestBody();
        body.withKeyId("0d0466b0-e727-4d9c-b35d-f84bb474a37f");
        request.withBody(body);
        try {
            DisableKeyRotationResponse response = client.disableKeyRotation(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

关闭key_id为0d0466b0-e727-4d9c-b35d-f84bb474a37f的密钥轮换

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = DisableKeyRotationRequest()
        request.body = OperateKeyRequestBody(
            key_id="0d0466b0-e727-4d9c-b35d-f84bb474a37f"
        )
        response = client.disable_key_rotation(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

关闭key_id为0d0466b0-e727-4d9c-b35d-f84bb474a37f的密钥轮换

```
package main
```

```
import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.DisableKeyRotationRequest{}
    request.Body = &model.OperateKeyRequestBody{
        KeyId: "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
    }
    response, err := client.DisableKeyRotation(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应

状态码	描述
504	网关超时

错误码

请参见**错误码**。

4.1.7.3 修改密钥轮换周期

功能介绍

- 功能介绍：修改用户主密钥轮换周期。

调用方法

请参见**如何调用API**。

URI

POST /v1.0/{project_id}/kms/update-key-rotation-interval

表 4-470 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-471 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-472 请求 Body 参数

参数	是否必选	参数类型	描述
key_id	是	String	密钥ID，36字节，满足正则匹配“^[0-9a-z]{8}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{12}$”。 例如：0d0466b0-e727-4d9c-b35d-f84bb474a37f。
rotation_interval	是	Integer	轮换周期，取值范围为30~365的整数。 周期范围设置根据密钥使用频率进行，若密钥使用频率高，建议设置为短周期；反之，则设置为长周期。
sequence	否	String	请求消息序列号，36字节序列号。 例如：919c82d4-8046-4722-9094-35c3c6524cff

响应参数

状态码： 400

表 4-473 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-474 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-475 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-476 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-477 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-478 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-479 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-480 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-481 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-482 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-483 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-484 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-485 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-486 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码

参数	参数类型	描述
error_msg	String	错误请求返回的错误信息

请求示例

修改用户主密钥ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的轮换周期为30天。

```
{
  "key_id" : "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
  "rotation_interval" : 30
}
```

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

修改用户主密钥ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的轮换周期为30天。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;


public class UpdateKeyRotationIntervalSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        UpdateKeyRotationIntervalRequest request = new UpdateKeyRotationIntervalRequest();
```

```
        UpdateKeyRotationIntervalRequestBody body = new UpdateKeyRotationIntervalRequestBody();
        body.withRotationInterval(30);
        body.withKeyId("0d0466b0-e727-4d9c-b35d-f84bb474a37f");
        request.withBody(body);
        try {
            UpdateKeyRotationIntervalResponse response = client.updateKeyRotationInterval(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

修改用户主密钥ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的轮换周期为30天。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = UpdateKeyRotationIntervalRequest()
        request.body = UpdateKeyRotationIntervalRequestBody(
            rotation_interval=30,
            key_id="0d0466b0-e727-4d9c-b35d-f84bb474a37f"
        )
        response = client.update_key_rotation_interval(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

修改用户主密钥ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的轮换周期为30天。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.UpdateKeyRotationIntervalRequest{}
    request.Body = &model.UpdateKeyRotationIntervalRequestBody{
        RotationInterval: int32(30),
        KeyId: "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
    }
    response, err := client.UpdateKeyRotationInterval(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码

状态码	描述
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.7.4 查询密钥轮换状态

功能介绍

- 功能介绍：查询用户主密钥轮换状态。

调用方法

请参见**如何调用API**。

URI

POST /v1.0/{project_id}/kms/get-key-rotation-status

表 4-487 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-488 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-489 请求 Body 参数

参数	是否必选	参数类型	描述
key_id	是	String	密钥ID，36字节，满足正则匹配“^[0-9a-z]{8}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{12}$”。 例如：0d0466b0-e727-4d9c-b35d-f84bb474a37f。
sequence	否	String	请求消息序列号，36字节序列号。 例如：919c82d4-8046-4722-9094-35c3c6524cff

响应参数

状态码： 200

表 4-490 响应 Body 参数

参数	参数类型	描述
key_rotation_enabled	Boolean	密钥轮换状态，默认为“false”，表示关闭密钥轮换功能。
rotation_interval	Integer	轮换周期，取值范围为30~365的整数。 周期范围设置根据密钥使用频率进行，若密钥使用频率高，建议设置为短周期；反之，则设置为长周期。
last_rotation_time	String	上一次密钥轮换时间。时间戳，即从1970年1月1日至该时间的总秒数。
number_of_rotations	Integer	密钥轮换次数。

状态码： 400

表 4-491 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-492 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-493 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-494 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-495 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-496 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-497 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-498 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-499 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-500 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-501 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-502 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-503 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-504 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

查询用户主密钥ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的轮换状态。

```
{
  "key_id" : "0d0466b0-e727-4d9c-b35d-f84bb474a37f"
}
```

响应示例

状态码： 200

请求已成功

```
{
  "key_rotation_enabled" : true,
  "rotation_interval" : 30,
  "last_rotation_time" : "1501578672000",
  "number_of_rotations" : 3
}
```

SDK 代码示例

SDK代码示例如下。

Java

查询用户主密钥ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的轮换状态。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;
```

```
public class ShowKeyRotationStatusSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        ShowKeyRotationStatusRequest request = new ShowKeyRotationStatusRequest();
        OperateKeyRequestBody body = new OperateKeyRequestBody();
        body.withKeyId("0d0466b0-e727-4d9c-b35d-f84bb474a37f");
        request.withBody(body);
        try {
            ShowKeyRotationStatusResponse response = client.showKeyRotationStatus(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

查询用户主密钥ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的轮换状态。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
```

```
        .build()

    try:
        request = ShowKeyRotationStatusRequest()
        request.body = OperateKeyRequestBody(
            key_id="0d0466b0-e727-4d9c-b35d-f84bb474a37f"
        )
        response = client.show_key_rotation_status(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

查询用户主密钥ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"的轮换状态。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ShowKeyRotationStatusRequest{}
    request.Body = &model.OperateKeyRequestBody{
        KeyId: "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
    }
    response, err := client.ShowKeyRotationStatus(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.8 密钥标签管理

4.1.8.1 查询密钥实例

功能介绍

- 功能介绍：查询密钥实例。通过标签过滤，查询指定用户主密钥的详细信息。

调用方法

请参见**如何调用API**。

URI

POST /v1.0/{project_id}/kms/{resource_instances}/action

表 4-505 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID
resource_instances	是	String	资源实例，固定值为resource_instances

请求参数

表 4-506 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-507 请求 Body 参数

参数	是否必选	参数类型	描述
limit	否	String	查询记录数（“action”为“count”时，无需设置此参数），如果“action”为“filter”，默认为“10”。 limit的取值范围为“1-1000”。
offset	否	String	索引位置。从offset指定的下一条数据开始查询。查询第一页数据时，将查询前一页数据时响应体中的值带入此参数（“action”为“count”时，无需设置此参数）。如果“action”为“filter”，offset默认为“0”。 offset必须为数字，不能为负数。
action	否	String	操作标识（可设置为“filter”或者“count”）。 • filter：表示过滤。 • count：表示查询总条数。
tags	否	Array of **Tag** objects	标签列表，key和value键值对的集合。 • key：表示标签键，一个密钥下最多包含10个key，key不能为空，不能重复，同一个key中value不能重复。key最大长度为36个字符。 • value：表示标签值。每个值最大长度43个字符，value之间为“与”的关系。

参数	是否必选	参数类型	描述
matches	否	Array of **TagItem** objects	搜索字段。 • key为要匹配的字段，例如：resource_name等。 • value为匹配的值，最大长度为255个字符，不能为空。
sequence	否	String	请求消息序列号，36字节序列号。 例如：919c82d4-8046-4722-9094-35c3c6524cff

表 4-508 Tag

参数	是否必选	参数类型	描述
key	否	String	键。 最大长度36个unicode字符。key不能为空。不能包含非打印字符“ASCII(0-31)”、“*”、“<”、“>”、“\”、“=”。
values	否	Array of strings	标签值集合

表 4-509 TagItem

参数	是否必选	参数类型	描述
key	是	String	键。 最大长度36个unicode字符。key不能为空。不能包含非打印字符“ASCII(0-31)”、“*”、“<”、“>”、“\”、“=”。
value	否	String	值。 每个值最大长度43个unicode字符，可以为空字符串。 不能包含非打印字符“ASCII(0-31)”、“*”、“<”、“>”、“\”、“=”。

响应参数

状态码： 200

表 4-510 响应 Body 参数

参数	参数类型	描述
resources	Array of **ActionResources** objects	资源实例列表，详情请参见resource字段数据结构说明。
total_count	Integer	总记录数。

表 4-511 ActionResources

参数	参数类型	描述
resource_id	String	资源ID。
resource_detail	**KeyDetails** object	密钥详情。
resource_name	String	资源名称，默认为空字符串。
tags	Array of **TagItem** objects	标签列表，没有标签，数组默认为空。

表 4-512 KeyDetails

参数	参数类型	描述
key_id	String	密钥ID。
domain_id	String	用户域ID。
key_alias	String	密钥别名。
realm	String	密钥区域。

参数	参数类型	描述
key_spec	String	密钥生成算法。 ● AES_256 ● SM4 ● RSA_2048 ● RSA_3072 ● RSA_4096 ● EC_P256 ● EC_P384 ● SM2
key_usage	String	密钥用途。 ● ENCRYPT_DECRYPT ● SIGN_VERIFY
key_description	String	密钥描述。
creation_date	String	密钥创建时间，时间戳，即从1970年1月1日至该时间的总秒数。
scheduled_deletion_date	String	密钥计划删除时间，时间戳，即从1970年1月1日至该时间的总秒数。
key_state	String	密钥状态，满足正则匹配“^[1-5]{1}$”，枚举如下： ● “1”表示待激活状态 ● “2”表示启用状态 ● “3”表示禁用状态 ● “4”表示计划删除状态 ● “5”表示等待导入状态
default_key_flag	String	默认主密钥标识，默认主密钥标识为1，非默认标识为0。
key_type	String	密钥类型。
expiration_time	String	密钥材料失效时间，时间戳，即从1970年1月1日至该时间的总秒数。
origin	String	密钥来源，默认为“kms”，枚举如下： ● kms表示密钥材料由kms生成 ● external表示密钥材料由外部导入
key_rotation_enabled	String	密钥轮换状态，默认为“false”，表示关闭密钥轮换功能。

参数	参数类型	描述
sys_enterprise_project_id	String	企业项目ID，默认为“0”。 ● 对于开通企业项目的用户，表示资源处于默认企业项目下。 ● 对于未开通企业项目的用户，表示资源未处于企业项目下。
keystore_id	String	密钥库ID

表 4-513 TagItem

参数	参数类型	描述
key	String	键。 最大长度36个unicode字符。 key不能为空。不能包含非打印字符“ASCII(0-31)”、“*”、“<”、“>”、“\”、“=”。
value	String	值。 每个值最大长度43个unicode字符，可以为空字符串。 不能包含非打印字符“ASCII(0-31)”、“*”、“<”、“>”、“\”、“=”。

状态码： 400

表 4-514 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-515 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-516 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-517 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-518 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-519 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-520 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-521 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-522 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-523 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-524 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-525 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-526 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-527 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码

参数	参数类型	描述
error_msg	String	错误请求返回的错误信息

请求示例

查询标签键为”key1“，标签值为”value1“或”value2“的密钥实例，起始位置为100，显示条数100。

```
{
  "offset" : "100",
  "limit" : "100",
  "action" : "filter",
  "tags" : [ {
    "key" : "key1",
    "values" : [ "value1", "value2" ]
  } ]
}
```

响应示例

状态码： 200

请求已成功

```
{
  "resources" : [ {
    "resource_id" : "90c03e67-5534-4ed0-acfa-89780e47a535",
    "resource_detail" : [ {
      "key_id" : "90c03e67-5534-4ed0-acfa-89780e47a535",
      "domain_id" : "4B688Fb77412Aee5570E7ecdbeB5afdc",
      "key_alias" : "tagTest_xmdmi",
      "key_description" : "123",
      "creation_date" : 1521449277000,
      "scheduled_deletion_date" : "",
      "key_state" : 2,
      "default_key_flag" : 0,
      "key_type" : 1,
      "key_rotation_enabled" : false,
      "expiration_time" : "",
      "origin" : "kms",
      "sys_enterprise_project_id" : "0",
      "realm" : "test"
    } ],
    "resource_name" : "tagTest_xmdmi",
    "tags" : [ {
      "key" : "key",
      "value" : "testValue!"
    }, {
      "key" : "haha",
      "value" : "testValue"
    } ]
  } ],
  "total_count" : 1
}
```

SDK 代码示例

SDK代码示例如下。

Java

查询标签键为”key1“，标签值为”value1“或”value2“的密钥实例，起始位置为100，显示条数100。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;

import java.util.List;
import java.util.ArrayList;

public class ListKmsByTagsSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        ListKmsByTagsRequest request = new ListKmsByTagsRequest();
        request.withResourceInstances("{resource_instances}");
        ListKmsByTagsRequestBody body = new ListKmsByTagsRequestBody();
        List<String> listTagsValues = new ArrayList<>();
        listTagsValues.add("value1");
        listTagsValues.add("value2");
        List<Tag> listbodyTags = new ArrayList<>();
        listbodyTags.add(
            new Tag()
                .withKey("key1")
                .withValues(listTagsValues)
        );
        body.withTags(listbodyTags);
        body.withAction("filter");
        body.withOffset("100");
        body.withLimit("100");
        request.withBody(body);
        try {
            ListKmsByTagsResponse response = client.listKmsByTags(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
```

```
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

查询标签键为”key1“，标签值为”value1“或”value2“的密钥实例，起始位置为100，显示条数100。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ListKmsByTagsRequest()
        request.resource_instances = "{resource_instances}"
        listValuesTags = [
            "value1",
            "value2"
        ]
        listTagsbody = [
            Tag(
                key="key1",
                values=listValuesTags
            )
        ]
        request.body = ListKmsByTagsRequestBody(
            tags=listTagsbody,
            action="filter",
            offset="100",
            limit="100"
        )
        response = client.list_kms_by_tags(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

查询标签键为”key1“，标签值为”value1“或”value2“的密钥实例，起始位置为100，显示条数100。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ListKmsByTagsRequest{}
    request.ResourceInstances = "{resource_instances}"
    var listValuesTags = []string{
        "value1",
        "value2",
    }
    keyTags:= "key1"
    var listTagsbody = []model.Tag{
        {
            Key: &keyTags,
            Values: &listValuesTags,
        },
    }
    actionListKmsByTagsRequestBody:= "filter"
    offsetListKmsByTagsRequestBody:= "100"
    limitListKmsByTagsRequestBody:= "100"
    request.Body = &model.ListKmsByTagsRequestBody{
        Tags: &listTagsbody,
        Action: &actionListKmsByTagsRequestBody,
        Offset: &offsetListKmsByTagsRequestBody,
        Limit: &limitListKmsByTagsRequestBody,
    }
    response, err := client.ListKmsByTags(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.8.2 查询密钥标签

功能介绍

- 功能介绍：查询密钥标签。

调用方法

请参见**如何调用API**。

URI

GET /v1.0/{project_id}/kms/{key_id}/tags

表 4-528 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID
key_id	是	String	密钥ID

请求参数

表 4-529 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码： 200

表 4-530 响应 Body 参数

参数	参数类型	描述
tags	Array of **TagItem** objects	标签列表，key和value键值对的集合。 • key：表示标签键，一个密钥下最多包含10个key，key不能为空，不能重复，同一个key中value不能重复。key最大长度为36个字符。 • value：表示标签值。每个值最大长度43个字符，value之间为“与”的关系。
existTagsNum	Integer	密钥的标签个数。。

表 4-531 TagItem

参数	参数类型	描述
key	String	键。 最大长度36个unicode字符。 key不能为空。不能包含非打印字符“ASCII(0-31)”、“*”、“<”、“>”、“\”、“=”。
value	String	值。 每个值最大长度43个unicode字符，可以为空字符串。 不能包含非打印字符“ASCII(0-31)”、“*”、“<”、“>”、“\”、“=”。

状态码： 400

表 4-532 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-533 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-534 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-535 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-536 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-537 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-538 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-539 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-540 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-541 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-542 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-543 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码

参数	参数类型	描述
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-544 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-545 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

无

响应示例

状态码： 200

请求已成功

```
{
  "tags" : [ {
    "key" : "key1",
    "value" : "value1"
  }, {
    "key" : "key2",
    "value" : "value2"
  } ],
  "existTagsNum" : 2
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
```

```
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;


public class ShowKmsTagsSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        ShowKmsTagsRequest request = new ShowKmsTagsRequest();
        request.withKeyId("{key_id}");
        try {
            ShowKmsTagsResponse response = client.showKmsTags(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
```

```
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ShowKmsTagsRequest()
        request.key_id = "{key_id}"
        response = client.show_kms_tags(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ShowKmsTagsRequest{}
    request.KeyId = "{key_id}"
    response, err := client.ShowKmsTags(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.8.3 添加密钥标签

功能介绍

- 功能介绍：添加密钥标签。

调用方法

请参见**如何调用API**。

URI

POST /v1.0/{project_id}/kms/{key_id}/tags

表 4-546 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID
key_id	是	String	密钥ID

请求参数

表 4-547 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-548 请求 Body 参数

参数	是否必选	参数类型	描述
tag	否	TagItem object	标签。
sequence	否	String	请求消息序列号，36字节序列号。 例如：919c82d4-8046-4722-9094-35c3c6524cff

表 4-549 TagItem

参数	是否必选	参数类型	描述
key	是	String	键。 最大长度36个unicode字符。key不能为空。不能包含非打印字符“ASCII(0-31)”、“*”、“<”、“>”、“\”、“=”。
value	否	String	值。 每个值最大长度43个unicode字符，可以为空字符串。 不能包含非打印字符“ASCII(0-31)”、“*”、“<”、“>”、“\”、“=”。

响应参数

状态码： 400

表 4-550 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-551 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-552 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-553 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-554 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-555 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-556 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-557 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-558 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-559 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-560 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-561 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码

参数	参数类型	描述
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-562 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-563 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

添加密钥标签，标签键为"DEV"，标签值为"DEV1"。

```
{
  "tag" : {
    "key" : "DEV",
    "value" : "DEV1"
  }
}
```

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

添加密钥标签，标签键为"DEV"，标签值为"DEV1"。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
```

```
import com.huaweicloud.sdk.kms.v2.model.*;


public class CreateKmsTagSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        CreateKmsTagRequest request = new CreateKmsTagRequest();
        request.withKeyId("{key_id}");
        CreateKmsTagRequestBody body = new CreateKmsTagRequestBody();
        TagItem tagbody = new TagItem();
        tagbody.withKey("DEV")
            .withValue("DEV1");
        body.withTag(tagbody);
        request.withBody(body);
        try {
            CreateKmsTagResponse response = client.createKmsTag(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

添加密钥标签，标签键为"DEV"，标签值为"DEV1"。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
```

```
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = CreateKmsTagRequest()
        request.key_id = "{key_id}"
        tagbody = TagItem(
            key="DEV",
            value="DEV1"
        )
        request.body = CreateKmsTagRequestBody(
            tag=tagbody
        )
        response = client.create_kms_tag(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

添加密钥标签，标签键为"DEV"，标签值为"DEV1"。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.CreateKmsTagRequest{}
    request.KeyId = "{key_id}"
    valueTag:= "DEV1"
    tagbody := &model.TagItem{
        Key: "DEV",
        Value: &valueTag,
```

```
    }
    request.Body = &model.CreateKmsTagRequestBody{
        Tag: tagbody,
    }
    response, err := client.CreateKmsTag(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
204	No Content
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.8.4 查询项目标签

功能介绍

- 功能介绍：查询用户在指定项目下的所有标签集合。

调用方法

请参见**如何调用API**。

URI

GET /v1.0/{project_id}/kms/tags

表 4-564 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-565 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码： 200

表 4-566 响应 Body 参数

参数	参数类型	描述
tags	Array of **Tag** objects	标签列表，key和value键值对的集合。 • key：表示标签键，一个密钥下最多包含10个key，key不能为空，不能重复，同一个key中value不能重复。key最大长度为36个字符。 • value：表示标签值。每个值最大长度43个字符，value之间为“与”的关系。

表 4-567 Tag

参数	参数类型	描述
key	String	键。 最大长度36个unicode字符。 key不能为空。不能包含非打印字符“ASCII(0-31)”、“*”、“<”、“>”、“\”、“=”。
values	Array of strings	标签值集合

状态码： 400

表 4-568 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-569 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-570 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-571 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-572 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-573 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-574 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-575 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-576 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-577 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-578 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-579 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码

参数	参数类型	描述
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-580 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-581 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

无

响应示例

状态码： 200

请求已成功

```
{
  "tags" : [ {
    "key" : "key1",
    "values" : [ "value1", "value2" ]
  }, {
    "key" : "key2",
    "values" : [ "value1", "value2" ]
  } ]
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
```

```
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;


public class ListKmsTagsSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        ListKmsTagsRequest request = new ListKmsTagsRequest();
        try {
            ListKmsTagsResponse response = client.listKmsTags(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
```

```
        .build()

    try:
        request = ListKmsTagsRequest()
        response = client.list_kms_tags(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ListKmsTagsRequest{}
    response, err := client.ListKmsTags(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功

状态码	描述
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.8.5 批量添加删除密钥标签

功能介绍

- 功能介绍：批量添加删除密钥标签。

调用方法

请参见**如何调用API**。

URI

POST /v1.0/{project_id}/kms/{key_id}/tags/action

表 4-582 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID
key_id	是	String	密钥ID

请求参数

表 4-583 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-584 请求 Body 参数

参数	是否必选	参数类型	描述
tags	是	Array of **TagItem** objects	标签列表，key和value键值对的集合。
action	是	String	操作标识： 仅限于“create”和“delete”。
sequence	否	String	请求消息序列号，36字节序列号。 例如：919c82d4-8046-4722-9094-35c3c6524cff

表 4-585 TagItem

参数	是否必选	参数类型	描述
key	是	String	键。 最大长度36个unicode字符。key不能为空。不能包含非打印字符“ASCII(0-31)”、“*”、“<”、“>”、“\”、“=”。
value	否	String	值。 每个值最大长度43个unicode字符，可以为空字符串。 不能包含非打印字符“ASCII(0-31)”、“*”、“<”、“>”、“\”、“=”。

响应参数

状态码： 400

表 4-586 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-587 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-588 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-589 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-590 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-591 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-592 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-593 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-594 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-595 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-596 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-597 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-598 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-599 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

添加多个密钥标签，标签1标签键为”key1“，标签值为”value1“。标签2标签键为”key“，标签值为”value3“。

```
{
  "action" : "create",
  "tags" : [ {
    "key" : "key1",
    "value" : "value1"
  }, {
    "key" : "key",
    "value" : "value3"
  } ]
}
```

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

添加多个密钥标签，标签1标签键为”key1“，标签值为”value1“。标签2标签键为”key“，标签值为”value3“。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;

import java.util.List;
import java.util.ArrayList;

public class BatchCreateKmsTagsSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        BatchCreateKmsTagsRequest request = new BatchCreateKmsTagsRequest();
        request.withKeyId("{key_id}");
        BatchCreateKmsTagsRequestBody body = new BatchCreateKmsTagsRequestBody();
        List<TagItem> listbodyTags = new ArrayList<>();
        listbodyTags.add(
            new TagItem()
                .withKey("key1")
                .withValue("value1")
        );
        listbodyTags.add(
            new TagItem()
                .withKey("key")
                .withValue("value3")
        );
        body.withAction("create");
        body.withTags(listbodyTags);
        request.withBody(body);
        try {
            BatchCreateKmsTagsResponse response = client.batchCreateKmsTags(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
```

```
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

添加多个密钥标签，标签1标签键为”key1“，标签值为”value1“。标签2标签键为”key“，标签值为”value3“。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = BatchCreateKmsTagsRequest()
        request.key_id = "{key_id}"
        listTagsbody = [
            TagItem(
                key="key1",
                value="value1"
            ),
            TagItem(
                key="key",
                value="value3"
            )
        ]
        request.body = BatchCreateKmsTagsRequestBody(
            action="create",
            tags=listTagsbody
        )
        response = client.batch_create_kms_tags(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

添加多个密钥标签，标签1标签键为”key1“，标签值为”value1“。标签2标签键为”key“，标签值为”value3“。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.BatchCreateKmsTagsRequest{}
    request.KeyId = "{key_id}"
    valueTags:= "value1"
    valueTags1:= "value3"
    var listTagsbody = []model.TagItem{
        {
            Key: "key1",
            Value: &valueTags,
        },
        {
            Key: "key",
            Value: &valueTags1,
        },
    }
    request.Body = &model.BatchCreateKmsTagsRequestBody{
        Action: "create",
        Tags: listTagsbody,
    }
    response, err := client.BatchCreateKmsTags(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
204	No Content
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.8.6 删除密钥标签

功能介绍

- 功能介绍：删除密钥标签。

调用方法

请参见**如何调用API**。

URI

DELETE /v1.0/{project_id}/kms/{key_id}/tags/{key}

表 4-600 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID
key_id	是	String	密钥ID
key	是	String	标签键的值

请求参数

表 4-601 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码： 400

表 4-602 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-603 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-604 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-605 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-606 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-607 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-608 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-609 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-610 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-611 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码

参数	参数类型	描述
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-612 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-613 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-614 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-615 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

无

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;


public class DeleteTagSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        DeleteTagRequest request = new DeleteTagRequest();
        request.withKeyId("{key_id}");
        request.withKey("{key}");
        try {
            DeleteTagResponse response = client.deleteTag(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *
```

```
if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = DeleteTagRequest()
        request.key_id = "{key_id}"
        request.key = "{key}"
        response = client.delete_tag(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.DeleteTagRequest{}
    request.KeyId = "{key_id}"
    request.Key = "{key}"
    response, err := client.DeleteTag(request)
    if err == nil {
```

```
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
204	No Content
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.9 密钥查询

4.1.9.1 查询密钥列表

功能介绍

- 功能介绍：查询用户所有密钥列表。

调用方法

请参见**如何调用API**。

URI

POST /v1.0/{project_id}/kms/list-keys

表 4-616 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-617 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-618 请求 Body 参数

参数	是否必选	参数类型	描述
limit	否	String	指定查询返回记录条数，默认值为2000。 取值范围：密钥最大个数范围以内，例如：20 如果指定查询记录条数小于存在的条数，响应参数“truncated”将返回“true”，表示存在分页。
marker	否	String	分页查询起始位置标识。分页查询收到的响应参数“truncated”为“true”时，可以发送连续的请求获取更多的记录条数，“marker”设置为响应的next_marker的值。例如：10
key_state	否	String	密钥状态，满足正则匹配“^[1-5]{1}$”，枚举如下： ● “1”表示待激活状态 ● “2”表示启用状态 ● “3”表示禁用状态 ● “4”表示计划删除状态 ● “5”表示等待导入状态

参数	是否必选	参数类型	描述
key_spec	否	String	密钥生成算法，默认为“AES_256”。查询所有（包含非对称）密钥需要指定参数“ALL”。 ● AES_256 ● SM4 ● RSA_2048 ● RSA_3072 ● RSA_4096 ● EC_P256 ● EC_P384 ● SM2 ● ALL
enterprise_project_id	否	String	企业多项目ID。用户未开通企业多项目时，不需要输入该字段。 用户开通企业多项目时，查询资源可以输入该字段。 若用户不输入该字段，默认查询租户所有有权限的企业多项目下的资源。 此时“enterprise_project_id”取值为“all”。 若用户输入该字段，取值满足以下任一条件. ● 取值为“all” ● 取值为“0” ● 满足正则匹配：“^[0-9a-z]{8}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{12}$”
sequence	否	String	请求消息序列号，36字节序列号。 例如：919c82d4-8046-4722-9094-35c3c6524cff

响应参数

状态码： 200

表 4-619 响应 Body 参数

参数	参数类型	描述
keys	Array of strings	key_id列表。
key_details	Array of **KeyDetails** objects	密钥详情列表。详情参见KeyDetails
next_marker	String	获取下一页所需要传递的“marker”值。当“truncated”为“false”时，“next_marker”为空。
truncated	String	是否还有下一页： ● “true”表示还有数据。 ● “false”表示已经是最后一页。
total	Integer	密钥总条数。

表 4-620 KeyDetails

参数	参数类型	描述
key_id	String	密钥ID。
domain_id	String	用户域ID。
key_alias	String	密钥别名。
realm	String	密钥区域。
key_spec	String	密钥生成算法。 ● AES_256 ● SM4 ● RSA_2048 ● RSA_3072 ● RSA_4096 ● EC_P256 ● EC_P384 ● SM2
key_usage	String	密钥用途。 ● ENCRYPT_DECRYPT ● SIGN_VERIFY
key_description	String	密钥描述。

参数	参数类型	描述
creation_date	String	密钥创建时间，时间戳，即从1970年1月1日至该时间的总秒数。
scheduled_deletion_date	String	密钥计划删除时间，时间戳，即从1970年1月1日至该时间的总秒数。
key_state	String	密钥状态，满足正则匹配“^[1-5]{1}$”，枚举如下： ● “1”表示待激活状态 ● “2”表示启用状态 ● “3”表示禁用状态 ● “4”表示计划删除状态 ● “5”表示等待导入状态
default_key_flag	String	默认主密钥标识，默认主密钥标识为1，非默认标识为0。
key_type	String	密钥类型。
expiration_time	String	密钥材料失效时间，时间戳，即从1970年1月1日至该时间的总秒数。
origin	String	密钥来源，默认为“kms”，枚举如下： ● kms表示密钥材料由kms生成 ● external表示密钥材料由外部导入
key_rotation_enabled	String	密钥轮换状态，默认为“false”，表示关闭密钥轮换功能。
sys_enterprise_project_id	String	企业项目ID，默认为“0”。 ● 对于开通企业项目的用户，表示资源处于默认企业项目下。 ● 对于未开通企业项目的用户，表示资源未处于企业项目下。
keystore_id	String	密钥库ID

状态码： 400

表 4-621 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-622 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-623 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-624 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-625 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-626 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-627 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-628 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-629 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-630 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-631 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-632 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-633 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-634 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

查询别名以"1"开始的密钥，至多返回2条密钥列表信息。

```
{
  "limit" : "2",
  "marker" : "1"
}
```

响应示例

状态码： 200

请求已成功

```
{
  "keys" : [ "0d0466b0-e727-4d9c-b35d-f84bb474a37f", "2e258389-bb1e-4568-a1d5-e1f50adf70ea" ],
  "key_details" : [ {
    "key_id" : "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
    "domain_id" : "00074811d5c27c4f8d48bb91e4a1dcfd",
    "key_alias" : "test",
    "realm" : "cn-north-7",
    "key_description" : "key_description",
    "creation_date" : "1502799822000",
    "scheduled_deletion_date" : "",
    "key_spec" : "AES_256",
    "key_usage" : "ENCRYPT_DECRYPT",
    "key_state" : "2",
    "default_key_flag" : "0",
    "key_type" : "1",
    "expiration_time" : "1501578672000",
    "origin" : "kms",
    "key_rotation_enabled" : "true",
    "sys_enterprise_project_id" : "0"
  }, {
    "key_id" : "2e258389-bb1e-4568-a1d5-e1f50adf70ea",
    "domain_id" : "00074811d5c27c4f8d48bb91e4a1dcfd",
    "key_alias" : "test",
    "realm" : "realm",
    "key_description" : "key_description",
    "creation_date" : "1502799822000",
    "scheduled_deletion_date" : "",
    "key_spec" : "AES_256",
```

```
    "key_usage" : "ENCRYPT_DECRYPT",
    "key_state" : "2",
    "default_key_flag" : "0",
    "key_type" : "1",
    "expiration_time" : "1501578672000",
    "origin" : "kms",
    "key_rotation_enabled" : "true",
    "sys_enterprise_project_id" : "0"
  } ],
  "next_marker" : "",
  "truncated" : "false",
  "total" : 2
}
```

SDK 代码示例

SDK代码示例如下。

Java

查询别名以"1"开始的密钥，至多返回2条密钥列表信息。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;


public class ListKeysSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        ListKeysRequest request = new ListKeysRequest();
        ListKeysRequestBody body = new ListKeysRequestBody();
        body.withMarker("1");
        body.withLimit("2");
        request.withBody(body);
        try {
            ListKeysResponse response = client.listKeys(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
```

```
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

查询别名以"1"开始的密钥，至多返回2条密钥列表信息。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ListKeysRequest()
        request.body = ListKeysRequestBody(
            marker="1",
            limit="2"
        )
        response = client.list_keys(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

查询别名以"1"开始的密钥，至多返回2条密钥列表信息。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
```

```
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ListKeysRequest{}
    markerListKeysRequestBody:= "1"
    limitListKeysRequestBody:= "2"
    request.Body = &model.ListKeysRequestBody{
        Marker: &markerListKeysRequestBody,
        Limit: &limitListKeysRequestBody,
    }
    response, err := client.ListKeys(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.9.2 查询密钥信息

功能介绍

- 功能介绍：查询密钥详细信息。

调用方法

请参见**如何调用API**。

URI

POST /v1.0/{project_id}/kms/describe-key

表 4-635 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-636 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-637 请求 Body 参数

参数	是否必选	参数类型	描述
key_id	是	String	密钥ID，36字节，满足正则匹配“^[0-9a-z]{8}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{12}$”。 例如：0d0466b0-e727-4d9c-b35d-f84bb474a37f。

参数	是否必选	参数类型	描述
sequence	否	String	请求消息序列号，36字节序列号。 例如：919c82d4-8046-4722-9094-35c3c6524cff

响应参数

状态码： 200

表 4-638 响应 Body 参数

参数	参数类型	描述
key_info	KeyDetails object	密钥详情。

表 4-639 KeyDetails

参数	参数类型	描述
key_id	String	密钥ID。
domain_id	String	用户域ID。
key_alias	String	密钥别名。
realm	String	密钥区域。
key_spec	String	密钥生成算法。 • AES_256 • SM4 • RSA_2048 • RSA_3072 • RSA_4096 • EC_P256 • EC_P384 • SM2
key_usage	String	密钥用途。 • ENCRYPT_DECRYPT • SIGN_VERIFY
key_description	String	密钥描述。

参数	参数类型	描述
creation_date	String	密钥创建时间，时间戳，即从1970年1月1日至该时间的总秒数。
scheduled_deletion_date	String	密钥计划删除时间，时间戳，即从1970年1月1日至该时间的总秒数。
key_state	String	密钥状态，满足正则匹配“^[1-5]{1}$”，枚举如下： ● “1”表示待激活状态 ● “2”表示启用状态 ● “3”表示禁用状态 ● “4”表示计划删除状态 ● “5”表示等待导入状态
default_key_flag	String	默认主密钥标识，默认主密钥标识为1，非默认标识为0。
key_type	String	密钥类型。
expiration_time	String	密钥材料失效时间，时间戳，即从1970年1月1日至该时间的总秒数。
origin	String	密钥来源，默认为“kms”，枚举如下： ● kms表示密钥材料由kms生成 ● external表示密钥材料由外部导入
key_rotation_enabled	String	密钥轮换状态，默认为“false”，表示关闭密钥轮换功能。
sys_enterprise_project_id	String	企业项目ID，默认为“0”。 ● 对于开通企业项目的用户，表示资源处于默认企业项目下。 ● 对于未开通企业项目的用户，表示资源未处于企业项目下。
keystore_id	String	密钥库ID

状态码： 400

表 4-640 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-641 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-642 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-643 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-644 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-645 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-646 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-647 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-648 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-649 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-650 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-651 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-652 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-653 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

查询ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"密钥的详细信息。

```
{
  "key_id" : "0d0466b0-e727-4d9c-b35d-f84bb474a37f"
}
```

响应示例

状态码： 200

请求已成功

```
{
  "key_info" : {
    "key_id" : "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
    "domain_id" : "00074811d5c27c4f8d48bb91e4a1dcfd",
    "key_alias" : "test",
    "realm" : "test",
    "key_description" : "key_description",
    "creation_date" : "1502799822000",
    "scheduled_deletion_date" : "",
    "key_spec" : "AES_256",
    "key_usage" : "ENCRYPT_DECRYPT",
    "key_state" : "2",
    "default_key_flag" : "0",
    "key_type" : "1",
    "expiration_time" : "1501578672000",
    "origin" : "kms",
    "key_rotation_enabled" : "false",
    "sys_enterprise_project_id" : "0"
  }
}
```

SDK 代码示例

SDK代码示例如下。

Java

查询ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"密钥的详细信息。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;


public class ListKeyDetailSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        ListKeyDetailRequest request = new ListKeyDetailRequest();
        OperateKeyRequestBody body = new OperateKeyRequestBody();
        body.withKeyId("0d0466b0-e727-4d9c-b35d-f84bb474a37f");
        request.withBody(body);
        try {
            ListKeyDetailResponse response = client.listKeyDetail(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

查询ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"密钥的详细信息。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
```

```
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ListKeyDetailRequest()
        request.body = OperateKeyRequestBody(
            key_id="0d0466b0-e727-4d9c-b35d-f84bb474a37f"
        )
        response = client.list_key_detail(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

查询ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"密钥的详细信息。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ListKeyDetailRequest{}
```

```
    request.Body = &model.OperateKeyRequestBody{
        KeyId: "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
    }
    response, err := client.ListKeyDetail(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.9.3 查询公钥信息

功能介绍

- 查询用户指定非对称密钥的公钥信息。

接口约束

对称密钥无法使用该操作进行查询。

调用方法

请参见**如何调用API**。

URI

POST /v1.0/{project_id}/kms/get-publickey

表 4-654 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-655 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-656 请求 Body 参数

参数	是否必选	参数类型	描述
key_id	是	String	密钥ID，36字节，满足正则匹配“^[0-9a-z]{8}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{12}$”。 例如：0d0466b0-e727-4d9c-b35d-f84bb474a37f。
sequence	否	String	请求消息序列号，36字节序列号。 例如：919c82d4-8046-4722-9094-35c3c6524cff

响应参数

状态码： 200

表 4-657 响应 Body 参数

参数	参数类型	描述
key_id	String	密钥ID。

参数	参数类型	描述
public_key	String	公钥信息。

状态码： 400

表 4-658 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-659 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-660 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-661 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-662 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-663 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-664 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-665 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-666 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-667 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-668 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-669 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-670 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-671 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

查询ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"公钥信息。

```
{
  "key_id" : "0d0466b0-e727-4d9c-b35d-f84bb474a37f"
}
```

响应示例

状态码： 200

请求已成功

```
{
  "key_id" : "bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e",
  "public_key" : "-----BEGIN PUBLIC KEY-----\r\nMIICCgKCAgEA3RQAXXwya9k4zV1/
Q3AFr37GO8JgFobDKZioAklLQdgElHZ/uxmP\r\n4bveNHpY6OI0Okk/1Ov8oJf+9W10VqVxbzihWa5n/
RMN0720DzLV7KuH4YylCGDb\r\n3JH/+bMbhF2qRArrrKod0kR9rYHrdPkI7O5fYQQprZ3kWnPgrhDoDFC8ja
+OelOg\r\nn4MMOGGYA/DAOb0XyxPnGl26PnUtvvF7aZbMW5x/Yq2yAVFE1cjqLaH7/j1C8KYE\r
\naOSYtl2nOif28WoweFavXpgVsb/iICTfqgC91BtCSFC5pT8vqZCimfoHmJCAkZa5\r
```

```
\nZ8QIqkOOf9F6iMqqIz7pGKgQSUmoKKY9j6DK3OwXDOB5gKu0vyuz+gW3b4SZn+Xa\r
\nKkEN8ZpXsdQdEGpe4SwIzSVyUGYNBOCLrsydBcPR7jWgQ6gs56IJrV2pdAtmBwKd\r\n6l33z1tQ7+/
h3IrZxXuuej/fRRUMlbVcmhTS2l6vle7HXgZj/dWzPsLLg9MGHu0+\r\no9PRr+brxTbrf5e2Zdr1ad35X/
b86gx7Grg1sYPkly2aEI4fsnDGPgFrudG+Hzx/\r\nABHejYfJEI6P0SXCzB/oDMkjw6XKhTSojMzuncAP/AM
+0LVYQxQe750qkb3hjBT0\r\nq/HBl/
4zMXA03tMb9QySnLK63uo64JMJiBsEe7wPLhHB3VzBZk9SvvECAwEAAQ==\r\n-----END PUBLIC KEY-----\r\n"
}
```

SDK 代码示例

SDK代码示例如下。

Java

查询ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"公钥信息。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;


public class ShowPublicKeySolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        ShowPublicKeyRequest request = new ShowPublicKeyRequest();
        OperateKeyRequestBody body = new OperateKeyRequestBody();
        body.withKeyId("0d0466b0-e727-4d9c-b35d-f84bb474a37f");
        request.withBody(body);
        try {
            ShowPublicKeyResponse response = client.showPublicKey(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
```

```
    }
}
```

Python

查询ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"公钥信息。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ShowPublicKeyRequest()
        request.body = OperateKeyRequestBody(
            key_id="0d0466b0-e727-4d9c-b35d-f84bb474a37f"
        )
        response = client.show_public_key(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

查询ID为"0d0466b0-e727-4d9c-b35d-f84bb474a37f"公钥信息。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"
```

```
    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ShowPublicKeyRequest{}
    request.Body = &model.OperateKeyRequestBody{
        KeyId: "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
    }
    response, err := client.ShowPublicKey(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.9.4 查询实例数

功能介绍

- 功能介绍：查询实例数，获取用户已经创建的用户主密钥数量。

调用方法

请参见**如何调用API**。

URI

GET /v1.0/{project_id}/kms/user-instances

表 4-672 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-673 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码： 200

表 4-674 响应 Body 参数

参数	参数类型	描述
instance_num	Integer	非默认用户主密钥个数。

状态码： 400

表 4-675 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-676 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-677 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-678 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-679 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-680 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-681 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-682 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-683 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-684 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-685 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-686 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-687 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-688 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

查询项目ID为0dea2644dc80d5d22ff1c01e3ebea6fc已创建的实例个数。

```
/v1.0/0dea2644dc80d5d22ff1c01e3ebea6fc/kms/user-instances
```

响应示例

状态码： 200

请求已成功

```
{
  "instance_num" : 15
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;


public class ShowUserInstancesSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running
```

```
this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        ShowUserInstancesRequest request = new ShowUserInstancesRequest();
        try {
            ShowUserInstancesResponse response = client.showUserInstances(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ShowUserInstancesRequest()
        response = client.show_user_instances(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ShowUserInstancesRequest{}
    response, err := client.ShowUserInstances(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误

状态码	描述
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.9.5 查询配额

功能介绍

- 功能介绍：查询配额，查询用户可以创建的用户主密钥配额总数及当前使用量信息。

调用方法

请参见**如何调用API**。

URI

GET /v1.0/{project_id}/kms/user-quotas

表 4-689 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-690 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码： 200

表 4-691 响应 Body 参数

参数	参数类型	描述
quotas	**Quotas** object	配额详情。

表 4-692 Quotas

参数	参数类型	描述
resources	Array of **Resources** objects	资源配额列表，详情请参见Resources

表 4-693 Resources

参数	参数类型	描述
type	String	配额类型。枚举值说明: • CMK，用户主密钥 • grant_per_CMK，单个用户主密钥可创建授权数
used	Integer	已使用配额数。
quota	Integer	配额总数。

状态码： 400

表 4-694 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-695 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-696 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-697 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-698 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-699 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-700 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-701 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-702 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-703 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-704 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-705 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-706 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-707 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码

参数	参数类型	描述
error_msg	String	错误请求返回的错误信息

请求示例

查询项目ID为0dea2644dc80d5d22ff1c01e3ebea6fc已使用的配额。

```
/v1.0/0dea2644dc80d5d22ff1c01e3ebea6fc/kms/user-quotas
```

响应示例

状态码： 200

请求已成功

```
{
  "quotas" : {
    "resources" : [ {
      "quota" : 20,
      "used" : 20,
      "type" : "CMK"
    }, {
      "quota" : 100,
      "used" : 0,
      "type" : "grant_per_CMK"
    } ]
  }
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;


public class ShowUserQuotasSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
```

```
            .withAk(ak)
            .withSk(sk);

    KmsClient client = KmsClient.newBuilder()
            .withCredential(auth)
            .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
            .build();
    ShowUserQuotasRequest request = new ShowUserQuotasRequest();
    try {
        ShowUserQuotasResponse response = client.showUserQuotas(request);
        System.out.println(response.toString());
    } catch (ConnectionException e) {
        e.printStackTrace();
    } catch (RequestTimeoutException e) {
        e.printStackTrace();
    } catch (ServiceResponseException e) {
        e.printStackTrace();
        System.out.println(e.getHttpStatusCode());
        System.out.println(e.getRequestId());
        System.out.println(e.getErrorCode());
        System.out.println(e.getErrorMsg());
    }
  }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ShowUserQuotasRequest()
        response = client.show_user_quotas(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
```

```
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ShowUserQuotasRequest{}
    response, err := client.ShowUserQuotas(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.10 查询密钥 API 版本信息

4.1.10.1 查询版本信息列表

功能介绍

- 功能介绍：查询API版本信息列表。

调用方法

请参见**如何调用API**。

URI

GET /

请求参数

无

响应参数

状态码： 200

表 4-708 响应 Body 参数

参数	参数类型	描述
versions	Array of **ApiVersionDetail** objects	描述version 相关对象的列表，详情请参见versions字段数据结构说明。

表 4-709 ApiVersionDetail

参数	参数类型	描述
id	String	版本ID（版本号），如“v1.0”。
links	Array of **ApiLink** objects	JSON对象，详情请参见links字段数据结构说明。
version	String	若该版本API支持微版本，则填支持的最大微版本号，如果不支持微版本，则返回空字符串。

参数	参数类型	描述
status	String	版本状态，包含如下3种： ● CURRENT：表示该版本为主推版本。 ● SUPPORTED：表示为老版本，但是现在还继续支持。 ● DEPRECATED：表示为废弃版本，存在后续删除的可能。
updated	String	版本发布时间，要求用UTC时间表示。如v1.发布的时间2014-06-28T12:20:21Z。
min_version	String	若该版本API 支持微版本，则填支持的最小微版本号，如果不支持微版本，则返回空字符串。

表 4-710 ApiLink

参数	参数类型	描述
href	String	API的URL地址。
rel	String	默认值self。

状态码： 500

表 4-711 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-712 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-713 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-714 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-715 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-716 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

```
无
```

响应示例

状态码： 200

请求已成功

```
{
  "versions" : [ {
    "min_version" : "",
    "links" : [ {
      "rel" : "self",
      "href" : "https://kms.region_id.domain.com/v1.0/"
    } ],
    "id" : "v1.0",
    "version" : "",
    "updated" : "2016-10-29T02:00:00Z",
    "status" : "CURRENT"
  } ]
}
```

状态码

状态码	描述
200	请求已成功
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.10.2 查询指定版本信息

功能介绍

- 功能介绍：查指定API版本信息。

调用方法

请参见**如何调用API**。

URI

GET /{version_id}

表 4-717 路径参数

参数	是否必选	参数类型	描述
version_id	是	String	API版本号

请求参数

无

响应参数

状态码： 200

表 4-718 响应 Body 参数

参数	参数类型	描述
version	Object	描述version 对象的列表，详情请参见ApiVersionDetail字段数据结构说明。

表 4-719 ApiVersionDetail

参数	参数类型	描述
id	String	版本ID（版本号），如“v1.0”。
links	Array of **ApiLink** objects	JSON对象，详情请参见links字段数据结构说明。
version	String	若该版本API支持微版本，则填支持的最大微版本号，如果不支持微版本，则返回空字符串。
status	String	版本状态，包含如下3种： ● CURRENT：表示该版本为主推版本。 ● SUPPORTED：表示为老版本，但是现在还继续支持。 ● DEPRECATED：表示为废弃版本，存在后续删除的可能。
updated	String	版本发布时间，要求用UTC时间表示。如v1.发布的时间2014-06-28T12:20:21Z。
min_version	String	若该版本API 支持微版本，则填支持的最小微版本号，如果不支持微版本，则返回空字符串。

表 4-720 ApiLink

参数	参数类型	描述
href	String	API的URL地址。
rel	String	默认值self。

状态码： 500

表 4-721 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-722 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-723 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-724 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-725 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-726 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

查询当前局点的V1.0版本接口信息。

```
/v1.0
```

响应示例

状态码： 200

查指定API版本信息

```
{
  "version" : {
    "min_version" : "",
    "links" : [ {
      "rel" : "self",
      "href" : "https://kms.region_id.domain.com/v1.0/"
    } ],
    "id" : "v1.0",
    "version" : "",
    "updated" : "2016-10-29T02:00:00Z",
    "status" : "CURRENT"
  }
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;


public class ShowVersionSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        ShowVersionRequest request = new ShowVersionRequest();
        request.withVersionId("{version_id}");
        try {
            ShowVersionResponse response = client.showVersion(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
```

```
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]

    credentials = BasicCredentials(ak, sk)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ShowVersionRequest()
        request.version_id = "{version_id}"
        response = client.show_version(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
```

```
        WithSk(sk).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ShowVersionRequest{}
    request.VersionId = "{version_id}"
    response, err := client.ShowVersion(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	查指定API版本信息
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.11 专属密钥库管理

4.1.11.1 创建专属密钥库

功能介绍

- "创建租户专属密钥库，专属密钥库使用DHSM实例作为密钥的存储"

调用方法

请参见**如何调用API**。

URI

POST /v1.0/{project_id}/keystores

表 4-727 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-728 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-729 请求 Body 参数

参数	是否必选	参数类型	描述
keystore_alias	是	String	专属密钥库别名，取值范围为1到255个字符，满足正则匹配“^[a-zA-Z0-9:/_-]{1,255}$”，且不与已有的专属密钥库别名重名。
hsm_cluster_id	是	String	DHSM集群Id，要求集群当前未创建专属密钥库。
hsm_ca_cert	是	String	DHSM集群的CA证书

响应参数

状态码： 200

表 4-730 响应 Body 参数

参数	参数类型	描述
keystore	**KeystoreInfo** object	密钥库信息。

表 4-731 KeystoreInfo

参数	参数类型	描述
keystore_id	String	密钥库ID
domain_id	String	用户域ID

状态码： 400

表 4-732 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-733 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-734 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-735 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-736 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-737 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-738 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-739 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-740 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-741 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-742 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-743 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-744 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-745 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

创建别名为"keystore_alia1"，集群ID为"hsm_cluster_id"的租户专属密钥库。

```
{
  "keystore_alias" : "keystore_alia1",
  "hsm_cluster_id" : "hsm_cluster_id",
  "hsm_ca_cert" : "-----BEGIN CERTIFICATE---******----END CERTIFICATE-----"
}
```

响应示例

状态码： 200

请求已成功

```
{
  "keystore" : {
```

```
    "keystore_id" : "bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e",
    "domain_id" : "b168fe00ff56492495a7d22974df2d0b"
  }
}
```

SDK 代码示例

SDK代码示例如下。

Java

创建别名为"keystore_alia1"，集群ID为"hsm_cluster_id"的租户专属密钥库。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;


public class CreateKeyStoreSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        CreateKeyStoreRequest request = new CreateKeyStoreRequest();
        CreateKeyStoreRequestBody body = new CreateKeyStoreRequestBody();
        body.withHsmCaCert("-----BEGIN CERTIFICATE---******----END CERTIFICATE-----");
        body.withHsmClusterId("hsm_cluster_id");
        body.withKeystoreAlias("keystore_alia1");
        request.withBody(body);
        try {
            CreateKeyStoreResponse response = client.createKeyStore(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

创建别名为"keystore_alia1"，集群ID为"hsm_cluster_id"的租户专属密钥库。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = CreateKeyStoreRequest()
        request.body = CreateKeyStoreRequestBody(
            hsm_ca_cert="-----BEGIN CERTIFICATE---******----END CERTIFICATE-----",
            hsm_cluster_id="hsm_cluster_id",
            keystore_alias="keystore_alia1"
        )
        response = client.create_key_store(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

创建别名为"keystore_alia1"，集群ID为"hsm_cluster_id"的租户专属密钥库。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"
```

```
    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.CreateKeyStoreRequest{}
    request.Body = &model.CreateKeyStoreRequestBody{
        HsmCaCert: "-----BEGIN CERTIFICATE---******----END CERTIFICATE-----",
        HsmClusterId: "hsm_cluster_id",
        KeystoreAlias: "keystore_alia1",
    }
    response, err := client.CreateKeyStore(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.11.2 查询专属密钥库列表

功能介绍

查询租户专属密钥库列表

调用方法

请参见**如何调用API**。

URI

GET /v1.0/{project_id}/keystores

表 4-746 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

表 4-747 Query 参数

参数	是否必选	参数类型	描述
limit	否	Integer	指定查询返回记录条数，默认值10。
offset	否	Integer	索引位置，从offset指定的下一条数据开始查询。

请求参数

表 4-748 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码： 200

表 4-749 响应 Body 参数

参数	参数类型	描述
total	Integer	密钥库总数
keystores	Array of **KeystoreDetails** objects	密钥详情列表。详情参见KeystoreDetails

表 4-750 KeystoreDetails

参数	参数类型	描述
keystore_id	String	密钥库ID
domain_id	String	用户域ID
keystore_alias	String	密钥库别名
keystore_type	String	密钥库类型
hsm_cluster_id	String	DHSM集群id，要求集群当前未创建专属密钥库
create_time	String	密钥库创建时间，UTC时间戳。

状态码： 400

表 4-751 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-752 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-753 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-754 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-755 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-756 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-757 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-758 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-759 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-760 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-761 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-762 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-763 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-764 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码

参数	参数类型	描述
error_msg	String	错误请求返回的错误信息

请求示例

无

响应示例

状态码： 200

请求已成功

```
{
  "total" : 1,
  "keystores" : [ {
    "keystore_id" : "bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e",
    "domain_id" : "b168fe00ff56492495a7d22974df2d0b",
    "keystore_alias" : "test",
    "keystore_type" : "typetest",
    "hsm_cluster_id" : "cluster_id",
    "create_time" : 1581507580000
  } ]
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;


public class ListKeyStoresSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
```

```
            .withCredential(auth)
            .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
            .build();
        ListKeyStoresRequest request = new ListKeyStoresRequest();
        try {
            ListKeyStoresResponse response = client.listKeyStores(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ListKeyStoresRequest()
        response = client.list_key_stores(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)
```

```
func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ListKeyStoresRequest{}
    response, err := client.ListKeyStores(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.11.3 获取专属密钥库

功能介绍

获取租户专属密钥库

调用方法

请参见**如何调用API**。

URI

GET /v1.0/{project_id}/keystores/{keystore_id}

表 4-765 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID
keystore_id	是	String	密钥库ID

请求参数

表 4-766 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码： 200

表 4-767 响应 Body 参数

参数	参数类型	描述
keystore	**KeystoreDetails** object	密钥库详情

表 4-768 KeystoreDetails

参数	参数类型	描述
keystore_id	String	密钥库ID
domain_id	String	用户域ID
keystore_alias	String	密钥库别名
keystore_type	String	密钥库类型
hsm_cluster_id	String	DHSM集群id，要求集群当前未创建专属密钥库
create_time	String	密钥库创建时间，UTC时间戳。

状态码： 400

表 4-769 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-770 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-771 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-772 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-773 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-774 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-775 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-776 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-777 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-778 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码

参数	参数类型	描述
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-779 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-780 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-781 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-782 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

无

响应示例

状态码： 200

请求成功

```
{
  "keystore" : {
    "keystore_id" : "bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e",
    "domain_id" : "b168fe00ff56492495a7d22974df2d0b",
    "keystore_alias" : "test",
    "keystore_type" : "typetest",
    "hsm_cluster_id" : "cluster_id",
    "create_time" : 1581507580000
  }
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;


public class ShowKeyStoreSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        ShowKeyStoreRequest request = new ShowKeyStoreRequest();
        request.withKeystoreId("{keystore_id}");
        try {
            ShowKeyStoreResponse response = client.showKeyStore(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ShowKeyStoreRequest()
        request.keystore_id = "{keystore_id}"
        response = client.show_key_store(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
```

```
        WithCredential(auth).
        Build())

    request := &model.ShowKeyStoreRequest{}
    request.KeystoreId = "{keystore_id}"
    response, err := client.ShowKeyStore(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	被请求的页面需要用户名和密码
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.11.4 删除专属密钥库

功能介绍

删除租户专属密钥库

调用方法

请参见**如何调用API**。

URI

DELETE /v1.0/{project_id}/keystores/{keystore_id}

表 4-783 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID
keystore_id	是	String	密钥库ID

请求参数

表 4-784 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码： 400

表 4-785 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-786 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-787 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-788 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-789 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-790 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-791 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-792 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-793 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-794 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-795 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-796 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-797 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-798 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

无

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;


public class DeleteKeyStoreSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        DeleteKeyStoreRequest request = new DeleteKeyStoreRequest();
        request.withKeystoreId("{keystore_id}");
        try {
            DeleteKeyStoreResponse response = client.deleteKeyStore(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = DeleteKeyStoreRequest()
        request.keystore_id = "{keystore_id}"
        response = client.delete_key_store(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
```

```
        WithCredential(auth).
        Build())

    request := &model.DeleteKeyStoreRequest{}
    request.KeystoreId = "{keystore_id}"
    response, err := client.DeleteKeyStore(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.11.5 启用专属密钥库

功能介绍

启用租户专属密钥库

调用方法

请参见**如何调用API**。

URI

POST /v1.0/{project_id}/keystores/{keystore_id}/enable

表 4-799 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID
keystore_id	是	String	密钥库ID

请求参数

表 4-800 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码： 200

表 4-801 响应 Body 参数

参数	参数类型	描述
keystore	**KeyStoreStateInfo** object	密钥库状态信息

表 4-802 KeyStoreStateInfo

参数	参数类型	描述
keystore_id	String	密钥库ID
keystore_state	String	密钥库状态

状态码： 400

表 4-803 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-804 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-805 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-806 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-807 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-808 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-809 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-810 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-811 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-812 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-813 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-814 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-815 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-816 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

无

响应示例

状态码： 200

请求成功

```
{
  "keystore" : {
    "keystore_id" : "bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e",
    "keystore_state" : "2"
  }
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;


public class EnableKeyStoreSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
```

```
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        EnableKeyStoreRequest request = new EnableKeyStoreRequest();
        request.withKeystoreId("{keystore_id}");
        try {
            EnableKeyStoreResponse response = client.enableKeyStore(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = EnableKeyStoreRequest()
        request.keystore_id = "{keystore_id}"
        response = client.enable_key_store(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
```

```
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.EnableKeyStoreRequest{}
    request.KeystoreId = "{keystore_id}"
    response, err := client.EnableKeyStore(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败

状态码	描述
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.11.6 禁用专属密钥库

功能介绍

禁用租户专属密钥库

调用方法

请参见**如何调用API**。

URI

POST /v1.0/{project_id}/keystores/{keystore_id}/disable

表 4-817 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID
keystore_id	是	String	密钥库ID

请求参数

表 4-818 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码： 200

表 4-819 响应 Body 参数

参数	参数类型	描述
keystore	**KeyStoreStateInfo** object	密钥库状态信息

表 4-820 KeyStoreStateInfo

参数	参数类型	描述
keystore_id	String	密钥库ID
keystore_state	String	密钥库状态

状态码： 400

表 4-821 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-822 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-823 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-824 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-825 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-826 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-827 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-828 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-829 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-830 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-831 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-832 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-833 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-834 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

无

响应示例

状态码： 200

请求成功

```
{
  "keystore" : {
    "keystore_id" : "bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e",
    "keystore_state" : "3"
  }
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;


public class DisableKeyStoreSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        DisableKeyStoreRequest request = new DisableKeyStoreRequest();
        request.withKeystoreId("{keystore_id}");
        try {
            DisableKeyStoreResponse response = client.disableKeyStore(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
```

```
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = DisableKeyStoreRequest()
        request.keystore_id = "{keystore_id}"
        response = client.disable_key_store(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"
```

```
    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.DisableKeyStoreRequest{}
    request.KeystoreId = "{keystore_id}"
    response, err := client.DisableKeyStore(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.12 多区域密钥

4.1.12.1 修改密钥所属的主区域

功能介绍

修改密钥所属的主区域。修改后当前区域会变为副本区域。

调用方法

请参见**如何调用API**。

URI

PUT /v2/{project_id}/kms/keys/{key_id}/update-primary-region

表 4-835 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID
key_id	是	String	待更新的密钥ID，36字节，满足正则匹配“^[0-9a-z]{8}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{12}$”。 例如：0d0466b0-e727-4d9c-b35d-f84bb474a37f。

请求参数

表 4-836 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-837 请求 Body 参数

参数	是否必选	参数类型	描述
key_id	是	String	待更新的密钥ID，36字节，满足正则匹配“^[0-9a-z]{8}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{12}$”。 例如：0d0466b0-e727-4d9c-b35d-f84bb474a37f。
primary_region	是	String	指定密钥所属新的主区域的区域编码。如cn-north-4。

响应参数

状态码： 200

表 4-838 响应 Body 参数

参数	参数类型	描述
key_id	String	密钥ID，36字节，满足正则匹配“^[0-9a-z]{8}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{12}$”。 例如：0d0466b0-e727-4d9c-b35d-f84bb474a37f。
primary_region	String	密钥所在主区域编码。如cn-north-4。

状态码： 400

表 4-839 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-840 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-841 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-842 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码

参数	参数类型	描述
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-843 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-844 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-845 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-846 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-847 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-848 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-849 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-850 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-851 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-852 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

```
{
  "key_id" : "826314dd-1b5b-4037-b976-5f9b7a17df46",
  "primary_region" : "cn-north-4"
}
```

响应示例

状态码： 200

请求已成功

```
{
  "key_id" : "bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e",
  "primary_region" : "cn-north-4"
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;


public class UpdatePrimaryRegionSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        UpdatePrimaryRegionRequest request = new UpdatePrimaryRegionRequest();
        request.withKeyId("{key_id}");
        UpdatePrimaryRegionRequestBody body = new UpdatePrimaryRegionRequestBody();
        body.withPrimaryRegion("cn-north-4");
        body.withKeyId("826314dd-1b5b-4037-b976-5f9b7a17df46");
        request.withBody(body);
        try {
            UpdatePrimaryRegionResponse response = client.updatePrimaryRegion(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
```

```
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = UpdatePrimaryRegionRequest()
        request.key_id = "{key_id}"
        request.body = UpdatePrimaryRegionRequestBody(
            primary_region="cn-north-4",
            key_id="826314dd-1b5b-4037-b976-5f9b7a17df46"
        )
        response = client.update_primary_region(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"
```

```
    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.UpdatePrimaryRegionRequest{}
    request.KeyId = "{key_id}"
    request.Body = &model.UpdatePrimaryRegionRequestBody{
        PrimaryRegion: "cn-north-4",
        KeyId: "826314dd-1b5b-4037-b976-5f9b7a17df46",
    }
    response, err := client.UpdatePrimaryRegion(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.12.2 复制密钥到指定区域

功能介绍

将本区域的密钥复制到指定区域。

调用方法

请参见**如何调用API**。

URI

POST /v2/{project_id}/kms/keys/{key_id}/replicate

表 4-853 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID
key_id	是	String	待复制的密钥ID，36字节，满足正则匹配“^[0-9a-z]{8}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{12}$”。 例如：0d0466b0-e727-4d9c-b35d-f84bb474a37f。

请求参数

表 4-854 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-855 请求 Body 参数

参数	是否必选	参数类型	描述
key_id	是	String	待复制的密钥ID，36字节，满足正则匹配“^[0-9a-z]{8}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{12}$”。 例如：0d0466b0-e727-4d9c-b35d-f84bb474a37f。
replica_region	是	String	复制密钥的目的区域编码。如cn-north-4。
key_alias	是	String	指定复制出的新密钥的别名。
key_description	否	String	指定复制出的新密钥的描述信息。
enterprise_project_id	否	String	指定复制出的新密钥的企业多项目ID。 • 用户未开通企业多项目时，不需要输入该字段。 • 用户开通企业多项目时，创建资源可以输入该字段。若用户户不输入该字段，默认创建属于默认企业多项目ID（ID为“0”）的资源。 注意：若用户没有默认企业多项目ID（ID为“0”）下的创建权限，则接口报错。
replica_project_id	是	String	指定复制出的新密钥的项目ID。
tags	否	Array of **TagItem** objects	标签列表，key和value键值对的集合。

表 4-856 TagItem

参数	是否必选	参数类型	描述
key	是	String	键。 最大长度36个unicode字符。key不能为空。不能包含非打印字符“ASCII(0-31)”、“*”、“<”、“>”、“\”、“=”。

参数	是否必选	参数类型	描述
value	否	String	值。 每个值最大长度43个unicode字符，可以为空字符串。 不能包含非打印字符“ASCII(0-31)”、“*”、“<”、“>”、“\”、“=”。

响应参数

状态码： 200

表 4-857 响应 Body 参数

参数	参数类型	描述
key_id	String	复制出的密钥ID，36字节，满足正则匹配“^[0-9a-z]{8}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{4}-[0-9a-z]{12}$”。 例如：0d0466b0-e727-4d9c-b35d-f84bb474a37f。
domain_id	String	用户域ID。
region	String	复制出的密钥所在区域编码。如cn-north-4。

状态码： 400

表 4-858 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-859 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-860 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-861 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-862 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-863 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-864 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-865 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-866 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-867 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-868 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-869 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-870 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-871 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码

参数	参数类型	描述
error_msg	String	错误请求返回的错误信息

请求示例

```
{
  "key_id" : "826314dd-1b5b-4037-b976-5f9b7a17df46",
  "replica_region" : "cn-north-4",
  "key_alias" : "replica_key_alias",
  "replica_project_id" : "2b31ed520xxxxxxebedb6e57xxxxxxxx"
}
```

响应示例

状态码： 200

请求已成功

```
{
  "key_id" : "8e85c00f-f7d0-47c9-ac96-548f977d16ba",
  "domain_id" : "3bab8e245e854f68af5967c00dd43127",
  "region" : "cn-north-4"
}
```

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.12.3 查询跨区域密钥所支持的区域

功能介绍

- 功能介绍：查询跨区域密钥所支持的区域。

调用方法

请参见**如何调用API**。

URI

GET /v2/{project_id}/kms/regions

表 4-872 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

表 4-873 Query 参数

参数	是否必选	参数类型	描述
limit	否	Integer	指定查询返回记录条数，默认值10。
offset	否	Integer	索引位置，从offset指定的下一条数据开始查询。

请求参数

表 4-874 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码： 200

表 4-875 响应 Body 参数

参数	参数类型	描述
regions	Array of strings	区域信息。

状态码： 400

表 4-876 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-877 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-878 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-879 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-880 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-881 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-882 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-883 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-884 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-885 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-886 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-887 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码

参数	参数类型	描述
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-888 响应 Body 参数

参数	参数类型	描述
error	Object	错误信息返回体。

表 4-889 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

无

响应示例

状态码： 200

请求已成功

```
{
  "regions" : [ "cn-north-2", "cn-north-4", "cn-north-1" ]
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;


public class ListSupportRegionsSolution {
```

```
    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        ListSupportRegionsRequest request = new ListSupportRegionsRequest();
        try {
            ListSupportRegionsResponse response = client.listSupportRegions(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ListSupportRegionsRequest()
        response = client.list_support_regions(request)
        print(response)
```

```
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ListSupportRegionsRequest{}
    response, err := client.ListSupportRegions(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败

状态码	描述
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.1.13 消息验证码

4.1.13.1 生成消息验证码

功能介绍

功能介绍：生成消息验证码

接口约束

- 仅支持key_usage为GENERATE_VERIFY_MAC的密钥进行操作。

调用方法

请参见**如何调用API**。

URI

POST /v1.0/{project_id}/kms/generate-mac

表 4-890 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-891 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-892 请求 Body 参数

参数	是否必选	参数类型	描述
key_id	是	String	密钥ID
mac_algorithm	是	String	Mac算法，HMAC_SM3只有中国区支持。枚举如下： • HMAC_SHA_256 • HMAC_SHA_384 • HMAC_SHA_512 • HMAC_SM3
message	是	String	待处理消息。原消息最小长度1、最大长度4096。请将原消息转为Base64格式后传入

响应参数

状态码： 200

表 4-893 响应 Body 参数

参数	参数类型	描述
key_id	String	密钥ID
mac_algorithm	String	Mac算法
mac	String	生成的消息验证码

请求示例

```
{
  "key_id" : "826314dd-1b5b-4037-b976-5f9b7a17df46",
  "mac_algorithm" : "HMAC_SHA_256",
```

```
  "message" : "ZmRzYQ=="
}
```

响应示例

状态码： 200

请求已成功

```
{
  "mac_algorithm" : "HMAC_SHA_256",
  "key_id" : "826314dd-1b5b-4037-b976-5f9b7a17df46",
  "mac" : "9d266415acf82985bb44daf4990604f1931384c88fd21ef32b202396755dcfd7"
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;


public class GenerateMacSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        GenerateMacRequest request = new GenerateMacRequest();
        GenerateMacRequestBody body = new GenerateMacRequestBody();
        body.withMessage("ZmRzYQ==");

body.withMacAlgorithm(GenerateMacRequestBody.MacAlgorithmEnum.fromValue("HMAC_SHA_256"));
        body.withKeyId("826314dd-1b5b-4037-b976-5f9b7a17df46");
        request.withBody(body);
        try {
            GenerateMacResponse response = client.generateMac(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
```

```
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = GenerateMacRequest()
        request.body = GenerateMacRequestBody(
            message="ZmRzYQ==",
            mac_algorithm="HMAC_SHA_256",
            key_id="826314dd-1b5b-4037-b976-5f9b7a17df46"
        )
        response = client.generate_mac(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
```

```
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.GenerateMacRequest{}
    request.Body = &model.GenerateMacRequestBody{
        Message: "ZmRzYQ==",
        MacAlgorithm: model.GetGenerateMacRequestBodyMacAlgorithmEnum().HMAC_SHA_256,
        KeyId: "826314dd-1b5b-4037-b976-5f9b7a17df46",
    }
    response, err := client.GenerateMac(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功

错误码

请参见**错误码**。

4.1.13.2 校验消息验证码

功能介绍

功能介绍：校验消息验证码

接口约束

- 仅支持key_usage为GENERATE_VERIFY_MAC的密钥进行操作。

调用方法

请参见**如何调用API**。

URI

POST /v1.0/{project_id}/kms/verify-mac

表 4-894 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-895 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-896 请求 Body 参数

参数	是否必选	参数类型	描述
key_id	是	String	密钥ID
mac_algorithm	是	String	Mac算法，HMAC_SM3只有中国区支持。枚举如下： • HMAC_SHA_256 • HMAC_SHA_384 • HMAC_SHA_512 • HMAC_SM3
message	是	String	待处理消息。原消息最小长度1、最大长度4096。请将原消息转为Base64格式后传入
mac	是	String	待校验的消息验证码

响应参数

状态码： 200

表 4-897 响应 Body 参数

参数	参数类型	描述
key_id	String	密钥ID
mac_algorithm	String	MAC算法
mac_valid	Boolean	消息验证码校验结果

请求示例

```
{
  "key_id" : "826314dd-1b5b-4037-b976-5f9b7a17df46",
  "mac_algorithm" : "HMAC_SHA_256",
  "message" : "ZmRzYQ==",
  "mac" : "8549f9f5ef335184e23e6d477776f0fd338d02c59e48e52e8d81d158e2fc9262"
}
```

响应示例

状态码： 200

请求已成功

```
{
  "mac_algorithm" : "HMAC_SHA_256",
  "key_id" : "826314dd-1b5b-4037-b976-5f9b7a17df46",
  "mac_valid" : false
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;


public class VerifyMacSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";
```

```
        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        VerifyMacRequest request = new VerifyMacRequest();
        VerifyMacRequestBody body = new VerifyMacRequestBody();
        body.withMac("8549f9f5ef335184e23e6d477776f0fd338d02c59e48e52e8d81d158e2fc9262");
        body.withMessage("ZmRzYQ==");
        body.withMacAlgorithm(VerifyMacRequestBody.MacAlgorithmEnum.fromValue("HMAC_SHA_256"));
        body.withKeyId("826314dd-1b5b-4037-b976-5f9b7a17df46");
        request.withBody(body);
        try {
            VerifyMacResponse response = client.verifyMac(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = VerifyMacRequest()
        request.body = VerifyMacRequestBody(
            mac="8549f9f5ef335184e23e6d477776f0fd338d02c59e48e52e8d81d158e2fc9262",
            message="ZmRzYQ==",
            mac_algorithm="HMAC_SHA_256",
            key_id="826314dd-1b5b-4037-b976-5f9b7a17df46"
        )
        response = client.verify_mac(request)
```

```
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.VerifyMacRequest{}
    request.Body = &model.VerifyMacRequestBody{
        Mac: "8549f9f5ef335184e23e6d477776f0fd338d02c59e48e52e8d81d158e2fc9262",
        Message: "ZmRzYQ==",
        MacAlgorithm: model.GetVerifyMacRequestBodyMacAlgorithmEnum().HMAC_SHA_256,
        KeyId: "826314dd-1b5b-4037-b976-5f9b7a17df46",
    }
    response, err := client.VerifyMac(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功

错误码

请参见**错误码**。

4.1.14 别名管理

4.1.14.1 关联密钥别名

功能介绍

关联别名。

您可以将别名从原密钥关联到另一个新的密钥

调用方法

请参见**如何调用API**。

URI

POST /v1.0/{project_id}/kms/alias/associate

表 4-898 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-899 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-900 请求 Body 参数

参数	是否必选	参数类型	描述
alias	是	String	待关联别名
target_key_id	是	String	待关联的密钥ID

响应参数

状态码： 200

表 4-901 响应 Body 参数

参数	参数类型	描述
domain_id	String	账号ID
key_id	String	密钥ID
alias	String	别名
alias_urn	String	别名资源定位符
create_time	String	创建时间
update_time	String	更新时间

请求示例

将 kms-1234 关联到密钥 bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e

```
{
  "target_key_id" : "bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e",
  "alias" : "kms-1234"
}
```

响应示例

状态码： 200

关联别名响应消息体

```
{
  "domain_id" : "3bab8e245e854f68af5967c00dd43127",
  "key_id" : "bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e",
  "alias" : "kms-1234",
  "alias_urn" : "kms:cn-north-7:3bab8e245e854f68af5967c00dd43127:alias:kms-1234",
  "create_time" : "2024-04-01T00:00:00Z",
  "update_time" : "2024-04-01T00:00:00Z"
}
```

SDK 代码示例

SDK代码示例如下。

Java

将 kms-1234 关联到密钥 bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;


public class AssociateAliasSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        AssociateAliasRequest request = new AssociateAliasRequest();
        AssociateAliasRequestBody body = new AssociateAliasRequestBody();
        body.withTargetKeyId("bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e");
        body.withAlias("kms-1234");
        request.withBody(body);
        try {
            AssociateAliasResponse response = client.associateAlias(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

将 kms-1234 关联到密钥 bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
```

```
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = AssociateAliasRequest()
        request.body = AssociateAliasRequestBody(
            target_key_id="bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e",
            alias="kms-1234"
        )
        response = client.associate_alias(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

将 kms-1234 关联到密钥 bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())
```

```
    request := &model.AssociateAliasRequest{}
    request.Body = &model.AssociateAliasRequestBody{
        TargetKeyId: "bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e",
        Alias: "kms-1234",
    }
    response, err := client.AssociateAlias(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	关联别名响应消息体

错误码

请参见**错误码**。

4.1.14.2 查询密钥关联的别名

功能介绍

查询一个密钥关联的所有别名

调用方法

请参见**如何调用API**。

URI

GET /v1.0/{project_id}/kms/aliases

表 4-902 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

表 4-903 Query 参数

参数	是否必选	参数类型	描述
key_id	否	String	密钥ID
limit	否	String	指定查询返回记录条数
marker	否	String	分页查询起始位置标识

请求参数

表 4-904 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码： 200

表 4-905 响应 Body 参数

参数	参数类型	描述
[数组元素]	Array<Array<**ListAliasResponseBody**>>	查询别名响应消息体

表 4-906 ListAliasResponseBody

参数	参数类型	描述
aliases	Array of **AliasEntity** objects	密钥关联的所有别名
page_info	**PageInfo** object	分页信息

表 4-907 AliasEntity

参数	参数类型	描述
domain_id	String	账号ID
key_id	String	密钥ID
alias	String	别名
alias_urn	String	别名资源定位符
create_time	String	创建时间
update_time	String	更新时间

表 4-908 PageInfo

参数	参数类型	描述
next_marker	String	下一页的查询标志
current_count	Integer	本页返回条目数量

请求示例

无

响应示例

状态码： 200

查询别名响应消息体

```
{
  "aliases" : [ {
    "domain_id" : "3bab8e245e854f68af5967c00dd43127",
    "key_id" : "bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e",
    "alias" : "kms-1234",
    "alias_urn" : "kms:cn-north-7:3bab8e245e854f68af5967c00dd43127:alias:kms-1234",
    "create_time" : "2024-04-01T00:00:00Z",
    "update_time" : "2024-04-01T00:00:00Z"
  } ],
  "page_info" : {
    "next_marker" : "",
    "current_count" : 1
  }
}
```

状态码

状态码	描述
200	查询别名响应消息体

错误码

请参见**错误码**。

4.1.14.3 关联密钥别名

功能介绍

为指定密钥关联一个新的别名

调用方法

请参见**如何调用API**。

URI

POST /v1.0/{project_id}/kms/aliases

表 4-909 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	密钥ID

请求参数

表 4-910 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-911 请求 Body 参数

参数	是否必选	参数类型	描述
key_id	是	String	密钥ID
alias	是	String	别名。一个账号在同一个区域别名不能重复

响应参数

状态码： 201

表 4-912 响应 Body 参数

参数	参数类型	描述
domain_id	String	账号ID
key_id	String	密钥ID
alias	String	别名
alias_urn	String	别名资源定位符
create_time	String	创建时间
update_time	String	更新时间

请求示例

为密钥bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e关联一个新的别名kms-1234

```
{
  "key_id" : "bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e",
  "alias" : "kms-1234"
}
```

响应示例

状态码： 201

创建成功

```
{
  "domain_id" : "3bab8e245e854f68af5967c00dd43127",
  "key_id" : "bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e",
  "alias" : "kms-1234",
  "alias_urn" : "kms:cn-north-7:3bab8e245e854f68af5967c00dd43127:alias:kms-1234",
  "create_time" : "2024-04-01T00:00:00Z",
  "update_time" : "2024-04-01T00:00:00Z"
}
```

SDK 代码示例

SDK代码示例如下。

Java

为密钥bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e关联一个新的别名kms-1234

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;


public class CreateAliasSolution {
```

```
    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        CreateAliasRequest request = new CreateAliasRequest();
        CreateAliasRequestBody body = new CreateAliasRequestBody();
        body.withAlias("kms-1234");
        body.withKeyId("bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e");
        request.withBody(body);
        try {
            CreateAliasResponse response = client.createAlias(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

为密钥bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e关联一个新的别名kms-1234

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
```

```
        .build()

    try:
        request = CreateAliasRequest()
        request.body = CreateAliasRequestBody(
            alias="kms-1234",
            key_id="bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e"
        )
        response = client.create_alias(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

为密钥bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e关联一个新的别名kms-1234

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.CreateAliasRequest{}
    request.Body = &model.CreateAliasRequestBody{
        Alias: "kms-1234",
        KeyId: "bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e",
    }
    response, err := client.CreateAlias(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
201	创建成功

错误码

请参见**错误码**。

4.1.14.4 删除密钥别名

功能介绍

删除别名

调用方法

请参见**如何调用API**。

URI

DELETE /v1.0/{project_id}/kms/aliases

表 4-913 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-914 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-915 请求 Body 参数

参数	是否必选	参数类型	描述
aliases	是	Array of strings	待删除的别名
key_id	是	String	别名关联的密钥ID

响应参数

无

请求示例

删除别名为kms-1234和kms-4567， 关联的密钥为bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e

```
{
  "key_id" : "bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e",
  "aliases" : [ "kms-1234", "kms-4567" ]
}
```

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

删除别名为kms-1234和kms-4567， 关联的密钥为bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kms.v2.region.KmsRegion;
import com.huaweicloud.sdk.kms.v2.*;
import com.huaweicloud.sdk.kms.v2.model.*;

import java.util.List;
import java.util.ArrayList;

public class DeleteAliasSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
```

```
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KmsClient client = KmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KmsRegion.valueOf("<YOUR REGION>"))
                .build();
        DeleteAliasRequest request = new DeleteAliasRequest();
        DeleteAliasRequestBody body = new DeleteAliasRequestBody();
        List<String> listbodyAliases = new ArrayList<>();
        listbodyAliases.add("kms-1234");
        listbodyAliases.add("kms-4567");
        body.withKeyId("bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e");
        body.withAliases(listbodyAliases);
        request.withBody(body);
        try {
            DeleteAliasResponse response = client.deleteAlias(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

删除别名为kms-1234和kms-4567， 关联的密钥为bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkms.v2.region.kms_region import KmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkms.v2 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = DeleteAliasRequest()
```

```
        listAliasesbody = [
            "kms-1234",
            "kms-4567"
        ]
        request.body = DeleteAliasRequestBody(
            key_id="bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e",
            aliases=listAliasesbody
        )
        response = client.delete_alias(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

删除别名为kms-1234和kms-4567， 关联的密钥为bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kms/v2/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kms.NewKmsClient(
        kms.KmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.DeleteAliasRequest{}
    var listAliasesbody = []string{
        "kms-1234",
        "kms-4567",
    }
    request.Body = &model.DeleteAliasRequestBody{
        KeyId: "bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e",
        Aliases: listAliasesbody,
    }
    response, err := client.DeleteAlias(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
204	删除成功

错误码

请参见**错误码**。

4.2 管理 SSH 密钥对

4.2.1 密钥对管理

4.2.1.1 创建和导入 SSH 密钥对

功能介绍

创建和导入SSH密钥对

调用方法

请参见**如何调用API**。

URI

POST /v3/{project_id}/keypairs

表 4-916 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-917 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-918 请求 Body 参数

参数	是否必选	参数类型	描述
keypair	是	**CreateKeypairAction** object	创建密钥对请求体请求参数

表 4-919 CreateKeypairAction

参数	是否必选	参数类型	描述
name	是	String	SSH密钥对的名称。 • 新创建的密钥对名称不能和已有密钥对的名称相同。 • SSH密钥对名称由英文字母、数字、下划线、中划线组成，长度不能超过255个字节
type	否	String	SSH密钥对的类型。ssh或x509。
public_key	否	String	导入公钥的字符串信息。
scope	否	String	租户级或者用户级。domain或user。
user_id	否	String	SSH密钥对所属的用户信息
key_protection	否	**KeyProtection** object	SSH密钥对私钥托管与保护。

表 4-920 KeyProtection

参数	是否必选	参数类型	描述
private_key	否	String	导入SSH密钥对的私钥。
encryption	是	**Encryption** object	对私钥进行加密存储的方式。

表 4-921 Encryption

参数	是否必选	参数类型	描述
type	是	String	取值范围：“kms”或“default”。 • “default”为默认加密方式，适用于没有kms服务的局点。 • “kms”为采用kms服务加密方式。 若局点没有kms服务，请填“default”。
kms_key_name	否	String	kms密钥的名称。 • 若“type”为“kms”，则必须填入"kms_key_name"或"kms_key_id"。
kms_key_id	否	String	kms密钥的ID。 • 若“type”为“kms”，则必须填入"kms_key_name"或"kms_key_id"。

响应参数

状态码： 200

表 4-922 响应 Body 参数

参数	参数类型	描述
keypair	**CreateKeypairResp** object	SSH密钥对信息详情

表 4-923 CreateKeypairResp

参数	参数类型	描述
name	String	SSH密钥对的名称
type	String	SSH密钥对的类型。ssh或x509。
public_key	String	SSH密钥对对应的publicKey信息
private_key	String	SSH密钥对对应的privateKey信息 ● 创建SSH密钥对时，响应中包括private_key的信息。 ● 导入SSH密钥对时，响应中不包括private_key的信息。
fingerprint	String	SSH密钥对应指纹信息
user_id	String	SSH密钥对所属的用户信息

状态码： 400

表 4-924 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

请求示例

```
{
  "keypair" : {
    "name" : "demo2"
  }
}
```

响应示例

状态码： 200

请求已成功

```
{
  "keypair" : {
    "name" : "demo",
    "type" : "ssh",
    "public_key" : "ssh-rsa AAAAB3NzaC1yc2EAAAADAQAB...",
    "private_key" : "-----BEGIN RSA PRIVATE KEY-----...",
    "fingerprint" : "49:ef:73:2b:9b:7f:2e:0c:58:d3:e3:42:8e:28:04:3b",
    "user_id" : "e4f380899b1248918f3d37098dc63746"
  }
}
```

状态码： 400

Error response

```
{
  "error_code" : "KPS.XXX",
  "error_msg" : "XXX"
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kps.v3.region.KpsRegion;
import com.huaweicloud.sdk.kps.v3.*;
import com.huaweicloud.sdk.kps.v3.model.*;


public class CreateKeypairSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KpsClient client = KpsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KpsRegion.valueOf("<YOUR REGION>"))
                .build();
        CreateKeypairRequest request = new CreateKeypairRequest();
        CreateKeypairRequestBody body = new CreateKeypairRequestBody();
        CreateKeypairAction keypairbody = new CreateKeypairAction();
        keypairbody.withName("demo2");
        body.withKeypair(keypairbody);
        request.withBody(body);
        try {
            CreateKeypairResponse response = client.createKeypair(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkps.v3.region.kps_region import KpsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkps.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KpsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KpsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = CreateKeypairRequest()
        keypairbody = CreateKeypairAction(
            name="demo2"
        )
        request.body = CreateKeypairRequestBody(
            keypair=keypairbody
        )
        response = client.create_keypair(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kps "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
```

```
        Build()

    client := kps.NewKpsClient(
        kps.KpsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.CreateKeypairRequest{}
    keypairbody := &model.CreateKeypairAction{
        Name: "demo2",
    }
    request.Body = &model.CreateKeypairRequestBody{
        Keypair: keypairbody,
    }
    response, err := client.CreateKeypair(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	Error response

错误码

请参见**错误码**。

4.2.1.2 清除私钥

功能介绍

清除SSH密钥对私钥。

调用方法

请参见**如何调用API**。

URI

DELETE /v3/{project_id}/keypairs/{keypair_name}/private-key

表 4-925 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。
keypair_name	是	String	密钥对名称。

请求参数

表 4-926 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码： 404

表 4-927 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

请求示例

无

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
```

```
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kps.v3.region.KpsRegion;
import com.huaweicloud.sdk.kps.v3.*;
import com.huaweicloud.sdk.kps.v3.model.*;


public class ClearPrivateKeySolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KpsClient client = KpsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KpsRegion.valueOf("<YOUR REGION>"))
                .build();
        ClearPrivateKeyRequest request = new ClearPrivateKeyRequest();
        request.withKeypairName("{keypair_name}");
        try {
            ClearPrivateKeyResponse response = client.clearPrivateKey(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkps.v3.region.kps_region import KpsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkps.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)
```

```
    client = KpsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KpsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ClearPrivateKeyRequest()
        request.keypair_name = "{keypair_name}"
        response = client.clear_private_key(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kps "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kps.NewKpsClient(
        kps.KpsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ClearPrivateKeyRequest{}
    request.KeypairName = "{keypair_name}"
    response, err := client.ClearPrivateKey(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功。
404	资源不存在，资源未找到

错误码

请参见**错误码**。

4.2.1.3 查询 SSH 密钥对列表

功能介绍

查询SSH密钥对列表

调用方法

请参见**如何调用API**。

URI

GET /v3/{project_id}/keypairs

表 4-928 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

表 4-929 Query 参数

参数	是否必选	参数类型	描述
limit	否	String	每页返回的个数。 默认值：50。
marker	否	String	分页查询起始的资源id，为空时为查询第一页

请求参数

表 4-930 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码： 200

表 4-931 响应 Body 参数

参数	参数类型	描述
keypairs	Array of **Keypairs** objects	SSH密钥对信息列表
page_info	**PageInfo** object	分页信息

表 4-932 Keypairs

参数	参数类型	描述
keypair	**Keypair** object	密钥对信息

表 4-933 Keypair

参数	参数类型	描述
name	String	SSH密钥对的名称
type	String	SSH密钥对的类型，值为“ssh”或“x509”
scope	String	租户级或者用户级。domain或user。
public_key	String	SSH密钥对对应的publicKey信息
fingerprint	String	SSH密钥对应指纹信息
is_key_protection	Boolean	是否托管密钥

参数	参数类型	描述
frozen_state	String	冻结状态 ● 0：正常状态 ● 1：普通冻结 ● 2：公安冻结 ● 3：普通冻结及公安冻结 ● 4：违规冻结 ● 5：普通冻结及违规冻结 ● 6：公安冻结及违规冻结 ● 7：普通冻结、公安冻结及违规冻结 ● 8：未实名认证冻结 ● 9：普通冻结及未实名认证冻结 ● 10：公安冻结及未实名认证冻结

表 4-934 PageInfo

参数	参数类型	描述
next_marker	String	返回下一页的查询地址
previous_marker	String	返回上一页的查询地址
current_count	Integer	返回条目数量

状态码： 400

表 4-935 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

请求示例

无

响应示例

状态码： 200

请求已成功

```
{
  "keypairs" : [ {
    "keypair" : {
      "name" : "1hprr3TI",
      "type" : "ssh",
      "scope" : "user",
      "public_key" : "ssh-rsa AAAAB3NzaC1yc2EAAAADAQABjV8GvwpSs.....",
      "fingerprint" : "65:ca:87:0a:16:86:59:ea:57:ea:18:37:58:e2:04:b0",
      "is_key_protection" : false,
      "frozen_state" : 0
    }
  } ],
  "page_info" : {
    "next_marker" : "KeyPair-dxxx",
    "previous_marker" : "KeyPair-xxxx",
    "current_count" : 49
  }
}
```

状态码： 400

Error response

```
{
  "error_code" : "KPS.XXX",
  "error_msg" : "XXX"
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kps.v3.region.KpsRegion;
import com.huaweicloud.sdk.kps.v3.*;
import com.huaweicloud.sdk.kps.v3.model.*;


public class ListKeypairsSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KpsClient client = KpsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KpsRegion.valueOf("<YOUR REGION>"))
                .build();
        ListKeypairsRequest request = new ListKeypairsRequest();
```

```
        try {
            ListKeypairsResponse response = client.listKeypairs(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkps.v3.region.kps_region import KpsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkps.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KpsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KpsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ListKeypairsRequest()
        response = client.list_keypairs(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kps "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
```

```
variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this
example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kps.NewKpsClient(
        kps.KpsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ListKeypairsRequest{}
    response, err := client.ListKeypairs(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	Error response

错误码

请参见**错误码**。

4.2.1.4 查询 SSH 密钥对详细信息

功能介绍

查询SSH密钥对详细信息

调用方法

请参见**如何调用API**。

URI

GET /v3/{project_id}/keypairs/{keypair_name}

表 4-936 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID
keypair_name	是	String	密钥对名称

请求参数

表 4-937 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码： 200

表 4-938 响应 Body 参数

参数	参数类型	描述
keypair	**KeypairDetail** object	密钥对详细信息

表 4-939 KeypairDetail

参数	参数类型	描述
name	String	SSH密钥对的名称
id	Long	SSH密钥对的ID
type	String	SSH密钥对的类型。ssh或x509。
scope	String	租户级或者用户级。domain或user。
public_key	String	SSH密钥对对应的publicKey信息
fingerprint	String	SSH密钥对应指纹信息
is_key_protection	Boolean	是否托管密钥
deleted	Boolean	SSH密钥对删除的标记

参数	参数类型	描述
description	String	SSH密钥对的描述信息
user_id	String	SSH密钥对所属的用户信息
create_time	Long	SSH密钥对创建的时间，时间戳，即从1970年1月1日至该时间的总秒数
delete_time	Long	SSH密钥对删除的时间，时间戳，即从1970年1月1日至该时间的总秒数
update_time	Long	SSH密钥对的更新时间，时间戳，即从1970年1月1日至该时间的总秒数
frozen_state	Integer	冻结状态 ● 0：正常状态 ● 1：普通冻结 ● 2：公安冻结 ● 3：普通冻结及公安冻结 ● 4：违规冻结 ● 5：普通冻结及违规冻结 ● 6：公安冻结及违规冻结 ● 7：普通冻结、公安冻结及违规冻结 ● 8：未实名认证冻结 ● 9：普通冻结及未实名认证冻结 ● 10：公安冻结及未实名认证冻结
key_id	String	密钥ID。
algorithm	String	生成算法。

状态码： 400

表 4-940 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

请求示例

无

响应示例

状态码： 200

请求已成功

```
{
  "keypair" : {
    "name" : "1hprr3TI",
    "id" : 116248,
    "type" : "ssh",
    "scope" : "user",
    "public_key" : "ssh-rsa AAAGenerated-by-Nova",
    "fingerprint" : "65:ca:87:0a:16:86:59:ea:57:ea:18:37:58:e2:04:b0",
    "is_key_protection" : false,
    "deleted" : false,
    "description" : "12345",
    "user_id" : "6c2a33b1b8474d0dbac0a24297127525",
    "create_time" : 1581507580000,
    "delete_time" : null,
    "update_time" : null,
    "frozen_state" : 0
  }
}
```

状态码： 400

Error response

```
{
  "error_code" : "KPS.XXX",
  "error_msg" : "XXX"
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kps.v3.region.KpsRegion;
import com.huaweicloud.sdk.kps.v3.*;
import com.huaweicloud.sdk.kps.v3.model.*;


public class ListKeypairDetailSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
```

```
            .withSk(sk);

    KpsClient client = KpsClient.newBuilder()
            .withCredential(auth)
            .withRegion(KpsRegion.valueOf("<YOUR REGION>"))
            .build();
    ListKeypairDetailRequest request = new ListKeypairDetailRequest();
    request.withKeypairName("{keypair_name}");
    try {
        ListKeypairDetailResponse response = client.listKeypairDetail(request);
        System.out.println(response.toString());
    } catch (ConnectionException e) {
        e.printStackTrace();
    } catch (RequestTimeoutException e) {
        e.printStackTrace();
    } catch (ServiceResponseException e) {
        e.printStackTrace();
        System.out.println(e.getHttpStatusCode());
        System.out.println(e.getRequestId());
        System.out.println(e.getErrorCode());
        System.out.println(e.getErrorMsg());
    }
  }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkps.v3.region.kps_region import KpsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkps.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KpsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KpsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ListKeypairDetailRequest()
        request.keypair_name = "{keypair_name}"
        response = client.list_keypair_detail(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
```

```
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kps "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kps.NewKpsClient(
        kps.KpsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ListKeypairDetailRequest{}
    request.KeypairName = "{keypair_name}"
    response, err := client.ListKeypairDetail(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	Error response

错误码

请参见**错误码**。

4.2.1.5 删除 SSH 密钥对

功能介绍

删除SSH密钥对。

调用方法

请参见**如何调用API**。

URI

DELETE /v3/{project_id}/keypairs/{keypair_name}

表 4-941 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID
keypair_name	是	String	密钥对名称

请求参数

表 4-942 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码： 400

表 4-943 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

请求示例

无

响应示例

状态码： 400

Error response

```
{
  "error_code" : "KPS.XXX",
  "error_msg" : "XXX"
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kps.v3.region.KpsRegion;
import com.huaweicloud.sdk.kps.v3.*;
import com.huaweicloud.sdk.kps.v3.model.*;


public class DeleteKeypairSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KpsClient client = KpsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KpsRegion.valueOf("<YOUR REGION>"))
                .build();
        DeleteKeypairRequest request = new DeleteKeypairRequest();
        request.withKeypairName("{keypair_name}");
        try {
            DeleteKeypairResponse response = client.deleteKeypair(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8
```

```
import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkps.v3.region.kps_region import KpsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkps.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KpsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KpsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = DeleteKeypairRequest()
        request.keypair_name = "{keypair_name}"
        response = client.delete_keypair(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kps "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kps.NewKpsClient(
        kps.KpsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.DeleteKeypairRequest{}
```

```
    request.KeypairName = "{keypair_name}"
    response, err := client.DeleteKeypair(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
204	OK
400	Error response

错误码

请参见**错误码**。

4.2.1.6 更新 SSH 密钥对描述

功能介绍

更新SSH密钥对描述。

调用方法

请参见**如何调用API**。

URI

PUT /v3/{project_id}/keypairs/{keypair_name}

表 4-944 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID
keypair_name	是	String	密钥对名称

请求参数

表 4-945 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-946 请求 Body 参数

参数	是否必选	参数类型	描述
keypair	是	**UpdateKeypairDescriptionReq** object	更新SSH密钥对描述消息体

表 4-947 UpdateKeypairDescriptionReq

参数	是否必选	参数类型	描述
description	是	String	描述信息

响应参数

状态码： 400

表 4-948 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

请求示例

```
{
  "keypair" : {
    "description" : "description"
  }
}
```

响应示例

状态码： 400

Error response

```
{
  "error_code" : "KPS.XXX",
  "error_msg" : "XXX"
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kps.v3.region.KpsRegion;
import com.huaweicloud.sdk.kps.v3.*;
import com.huaweicloud.sdk.kps.v3.model.*;


public class UpdateKeypairDescriptionSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KpsClient client = KpsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KpsRegion.valueOf("<YOUR REGION>"))
                .build();
        UpdateKeypairDescriptionRequest request = new UpdateKeypairDescriptionRequest();
        request.withKeypairName("{keypair_name}");
        UpdateKeypairDescriptionRequestBody body = new UpdateKeypairDescriptionRequestBody();
        UpdateKeypairDescriptionReq keypairbody = new UpdateKeypairDescriptionReq();
        keypairbody.withDescription("description");
        body.withKeypair(keypairbody);
        request.withBody(body);
        try {
            UpdateKeypairDescriptionResponse response = client.updateKeypairDescription(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
```

```
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkps.v3.region.kps_region import KpsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkps.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KpsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KpsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = UpdateKeypairDescriptionRequest()
        request.keypair_name = "{keypair_name}"
        keypairbody = UpdateKeypairDescriptionReq(
            description="description"
        )
        request.body = UpdateKeypairDescriptionRequestBody(
            keypair=keypairbody
        )
        response = client.update_keypair_description(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kps "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"
```

```
    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kps.NewKpsClient(
        kps.KpsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.UpdateKeypairDescriptionRequest{}
    request.KeypairName = "{keypair_name}"
    keypairbody := &model.UpdateKeypairDescriptionReq{
        Description: "description",
    }
    request.Body = &model.UpdateKeypairDescriptionRequestBody{
        Keypair: keypairbody,
    }
    response, err := client.UpdateKeypairDescription(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK
400	Error response

错误码

请参见**错误码**。

4.2.1.7 导入私钥

功能介绍

导入私钥到指定密钥对。

调用方法

请参见**如何调用API**。

URI

POST /v3/{project_id}/keypairs/private-key/import

表 4-949 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-950 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-951 请求 Body 参数

参数	是否必选	参数类型	描述
keypair	是	**ImportPrivateKeyKeypairBean** object	待导入的密钥对信息。

表 4-952 ImportPrivateKeyKeypairBean

参数	是否必选	参数类型	描述
name	是	String	SSH密钥对的名称。 - 新创建的密钥对名称不能和已有密钥对的名称相同。 - SSH密钥对名称由英文字母、数字、下划线、中划线组成，长度不能超过64个字节
user_id	否	String	SSH密钥对所属的用户信息
key_protection	是	**ImportPrivateKeyProtection** object	SSH密钥对私钥托管与保护。

表 4-953 ImportPrivateKeyProtection

参数	是否必选	参数类型	描述
private_key	是	String	导入SSH密钥对的私钥。
encryption	是	**Encryption** object	对私钥进行加密存储的方式。

表 4-954 Encryption

参数	是否必选	参数类型	描述
type	是	String	取值范围：“kms”或“default”。 • “default”为默认加密方式，适用于没有kms服务的局点。 • “kms”为采用kms服务加密方式。 若局点没有kms服务，请填“default”。
kms_key_name	否	String	kms密钥的名称。 • 若“type”为“kms”，则必须填入"kms_key_name"或"kms_key_id"。
kms_key_id	否	String	kms密钥的ID。 • 若“type”为“kms”，则必须填入"kms_key_name"或"kms_key_id"。

响应参数

状态码： 200

表 4-955 响应 Body 参数

参数	参数类型	描述
keypair	**ImportPrivateKeyKeypairBean** object	—

表 4-956 ImportPrivateKeyKeypairBean

参数	参数类型	描述
name	String	SSH密钥对的名称。 - 新创建的密钥对名称不能和已有密钥对的名称相同。 - SSH密钥对名称由英文字母、数字、下划线、中划线组成，长度不能超过64个字节
user_id	String	SSH密钥对所属的用户信息
key_protection	**ImportPrivateKeyProtection** object	SSH密钥对私钥托管与保护。

表 4-957 ImportPrivateKeyProtection

参数	参数类型	描述
private_key	String	导入SSH密钥对的私钥。
encryption	**Encryption** object	对私钥进行加密存储的方式。

表 4-958 Encryption

参数	参数类型	描述
type	String	取值范围：“kms”或“default”。 ● “default”为默认加密方式，适用于没有kms服务的局点。 ● “kms”为采用kms服务加密方式。 若局点没有kms服务，请填“default”。
kms_key_name	String	kms密钥的名称。 ● 若“type”为“kms”，则必须填入"kms_key_name"或"kms_key_id"。
kms_key_id	String	kms密钥的ID。 ● 若“type”为“kms”，则必须填入"kms_key_name"或"kms_key_id"。

状态码： 400

表 4-959 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

请求示例

```
{
  "keypair" : {
    "name" : "demo2",
    "key_protection" : {
      "private_key" : "-----BEGIN RSA PRIVATE KEY-----...",
      "encryption" : {
        "type" : "kms",
        "kms_key_name" : "kps/default"
      }
    }
  }
}
```

响应示例

状态码： 200

请求已成功

```
{
  "keypair" : {
    "name" : "demo2"
  }
}
```

状态码： 400

Error response

```
{
  "error_code" : "KPS.XXX",
  "error_msg" : "XXX"
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kps.v3.region.KpsRegion;
import com.huaweicloud.sdk.kps.v3.*;
import com.huaweicloud.sdk.kps.v3.model.*;
```

```
public class ImportPrivateKeySolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KpsClient client = KpsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KpsRegion.valueOf("<YOUR REGION>"))
                .build();
        ImportPrivateKeyRequest request = new ImportPrivateKeyRequest();
        ImportPrivateKeyRequestBody body = new ImportPrivateKeyRequestBody();
        Encryption encryptionKeyProtection = new Encryption();
        encryptionKeyProtection.withType(Encryption.TypeEnum.fromValue("kms"))
            .withKmsKeyName("kps/default");
        ImportPrivateKeyProtection keyProtectionKeypair = new ImportPrivateKeyProtection();
        keyProtectionKeypair.withPrivateKey("-----BEGIN RSA PRIVATE KEY-----...")
            .withEncryption(encryptionKeyProtection);
        ImportPrivateKeyKeypairBean keypairbody = new ImportPrivateKeyKeypairBean();
        keypairbody.withName("demo2")
            .withKeyProtection(keyProtectionKeypair);
        body.withKeypair(keypairbody);
        request.withBody(body);
        try {
            ImportPrivateKeyResponse response = client.importPrivateKey(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkps.v3.region.kps_region import KpsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkps.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
```

```
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KpsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KpsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ImportPrivateKeyRequest()
        encryptionKeyProtection = Encryption(
            type="kms",
            kms_key_name="kps/default"
        )
        keyProtectionKeypair = ImportPrivateKeyProtection(
            private_key="-----BEGIN RSA PRIVATE KEY-----...",
            encryption=encryptionKeyProtection
        )
        keypairbody = ImportPrivateKeyKeypairBean(
            name="demo2",
            key_protection=keyProtectionKeypair
        )
        request.body = ImportPrivateKeyRequestBody(
            keypair=keypairbody
        )
        response = client.import_private_key(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kps "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kps.NewKpsClient(
        kps.KpsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ImportPrivateKeyRequest{}
```

```
    kmsKeyNameEncryption:= "kps/default"
    encryptionKeyProtection := &model.Encryption{
        Type: model.GetEncryptionTypeEnum().KMS,
        KmsKeyName: &kmsKeyNameEncryption,
    }
    keyProtectionKeypair := &model.ImportPrivateKeyProtection{
        PrivateKey: "-----BEGIN RSA PRIVATE KEY-----...",
        Encryption: encryptionKeyProtection,
    }
    keypairbody := &model.ImportPrivateKeyKeypairBean{
        Name: "demo2",
        KeyProtection: keyProtectionKeypair,
    }
    request.Body = &model.ImportPrivateKeyRequestBody{
        Keypair: keypairbody,
    }
    response, err := client.ImportPrivateKey(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	Error response

错误码

请参见**错误码**。

4.2.1.8 导出私钥

功能介绍

导出指定密钥对的私钥。

调用方法

请参见**如何调用API**。

URI

POST /v3/{project_id}/keypairs/private-key/export

表 4-960 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-961 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-962 请求 Body 参数

参数	是否必选	参数类型	描述
keypair	是	**KeypairBean** object	待导出私钥的的密钥对信息。

表 4-963 KeypairBean

参数	是否必选	参数类型	描述
name	是	String	SSH密钥对名称。

响应参数

状态码： 200

表 4-964 响应 Body 参数

参数	参数类型	描述
keypair	**ExportPrivateKeyKeypairBean** object	导出的私钥信息。

表 4-965 ExportPrivateKeyKeypairBean

参数	参数类型	描述
name	String	SSH密钥对的名称。
private_key	String	SSH密钥对的私钥

状态码： 400

表 4-966 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

请求示例

```
{
  "keypair" : {
    "name" : "demo2"
  }
}
```

响应示例

状态码： 200

请求已成功

```
{
  "keypair" : {
    "name" : "demo2",
    "private_key" : "-----BEGIN RSA PRIVATE KEY-----..."
  }
}
```

状态码： 400

Error response

```
{
  "error_code" : "KPS.XXX",
  "error_msg" : "XXX"
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
```

```
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kps.v3.region.KpsRegion;
import com.huaweicloud.sdk.kps.v3.*;
import com.huaweicloud.sdk.kps.v3.model.*;


public class ExportPrivateKeySolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KpsClient client = KpsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KpsRegion.valueOf("<YOUR REGION>"))
                .build();
        ExportPrivateKeyRequest request = new ExportPrivateKeyRequest();
        ExportPrivateKeyRequestBody body = new ExportPrivateKeyRequestBody();
        KeypairBean keypairbody = new KeypairBean();
        keypairbody.withName("demo2");
        body.withKeypair(keypairbody);
        request.withBody(body);
        try {
            ExportPrivateKeyResponse response = client.exportPrivateKey(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkps.v3.region.kps_region import KpsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkps.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
```

```
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KpsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KpsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ExportPrivateKeyRequest()
        keypairbody = KeypairBean(
            name="demo2"
        )
        request.body = ExportPrivateKeyRequestBody(
            keypair=keypairbody
        )
        response = client.export_private_key(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kps "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kps.NewKpsClient(
        kps.KpsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ExportPrivateKeyRequest{}
    keypairbody := &model.KeypairBean{
        Name: "demo2",
    }
    request.Body = &model.ExportPrivateKeyRequestBody{
        Keypair: keypairbody,
    }
    response, err := client.ExportPrivateKey(request)
```

```
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	Error response

错误码

请参见**错误码**。

4.2.1.9 批量导入 SSH 密钥对

功能介绍

批量导入SSH密钥对,单次批量导入不得超过10条记录。

调用方法

请参见**如何调用API**。

URI

POST /v3/{project_id}/keypairs/batch-import

表 4-967 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。

请求参数

表 4-968 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-969 请求 Body 参数

参数	是否必选	参数类型	描述
[数组元素]	是	Array of **CreateKeypairRequestBody** objects	待批量导入的密钥对列表

表 4-970 CreateKeypairRequestBody

参数	是否必选	参数类型	描述
keypair	是	**CreateKeypairAction** object	创建密钥对请求体请求参数

表 4-971 CreateKeypairAction

参数	是否必选	参数类型	描述
name	是	String	SSH密钥对的名称。 • 新创建的密钥对名称不能和已有密钥对的名称相同。 • SSH密钥对名称由英文字母、数字、下划线、中划线组成，长度不能超过255个字节
type	否	String	SSH密钥对的类型。ssh或x509。
public_key	否	String	导入公钥的字符串信息。
scope	否	String	租户级或者用户级。domain或user。

参数	是否必选	参数类型	描述
user_id	否	String	SSH密钥对所属的用户信息
key_protection	否	KeyProtection object	SSH密钥对私钥托管与保护。

表 4-972 KeyProtection

参数	是否必选	参数类型	描述
private_key	否	String	导入SSH密钥对的私钥。
encryption	是	Encryption object	对私钥进行加密存储的方式。

表 4-973 Encryption

参数	是否必选	参数类型	描述
type	是	String	取值范围：“kms”或“default”。 • “default”为默认加密方式，适用于没有kms服务的局点。 • “kms”为采用kms服务加密方式。 若局点没有kms服务，请填“default”。
kms_key_name	否	String	kms密钥的名称。 • 若“type”为“kms”，则必须填入"kms_key_name"或"kms_key_id"。
kms_key_id	否	String	kms密钥的ID。 • 若“type”为“kms”，则必须填入"kms_key_name"或"kms_key_id"。

响应参数

状态码： 200

表 4-974 响应 Body 参数

参数	参数类型	描述
failed_keypairs	Array of **FailedKeypair** objects	导入失败的SSH密钥对信息及导入失败的原因
succeeded_keypairs	Array of **CreateKeypairResponseBody** objects	成功导入的SSH密钥对信息

表 4-975 FailedKeypair

参数	参数类型	描述
keypair_name	String	SSH密钥对名称
failed_reason	String	导入失败的原因

表 4-976 CreateKeypairResponseBody

参数	参数类型	描述
keypair	**CreateKeypairResp** object	SSH密钥对信息详情

表 4-977 CreateKeypairResp

参数	参数类型	描述
name	String	SSH密钥对的名称
type	String	SSH密钥对的类型。ssh或x509。
public_key	String	SSH密钥对对应的publicKey信息
private_key	String	SSH密钥对对应的privateKey信息 • 创建SSH密钥对时，响应中包括private_key的信息。 • 导入SSH密钥对时，响应中不包括private_key的信息。
fingerprint	String	SSH密钥对应指纹信息
user_id	String	SSH密钥对所属的用户信息

请求示例

批量导入名称为”demo2，demo3"的SSH密钥对。

```
[ {
  "keypair" : {
    "name" : "demo2",
    "public_key" : "ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAAA... Generated-by-Nova"
  }
}, {
  "keypair" : {
    "name" : "demo3",
    "public_key" : "ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAAA... Generated-by-Nova"
  }
} ]
```

响应示例

状态码： 200

批量导入SSH密钥对响应信息

```
{
  "failed_keypairs" : [ {
    "keypair_name" : "demo3",
    "failed_reason" : "{\"error_code\":\"KPS.7004\", \"error_msg\":\"Key pair 'demo3' already exists.\"}"
  } ],
  "succeeded_keypairs" : [ {
    "keypair" : {
      "name" : "demo2",
      "fingerprint" : "SHA256:Wm91+h496cPk5JleSp4RdD2n4Z/5KdlDeD51lmiZ11M",
      "type" : "ssh",
      "public_key" : "ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAAA... Generated-by-Nova",
      "user_id" : "6c2a33b1b8474d0dbac0a24297127525"
    }
  } ]
}
```

状态码

状态码	描述
200	批量导入SSH密钥对响应信息

错误码

请参见**错误码**。

4.2.1.10 批量导出密钥对私钥

功能介绍

批量导出密钥对私钥，单次最多导出10条数据

调用方法

请参见**如何调用API**。

URI

POST /v3/{project_id}/keypairs/private-key/batch-export

表 4-978 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。

请求参数

表 4-979 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-980 请求 Body 参数

参数	是否必选	参数类型	描述
[数组元素]	是	Array of **Keypairs** objects	待导出的密钥列表

表 4-981 Keypairs

参数	是否必选	参数类型	描述
keypair	是	**Keypair** object	密钥对信息

表 4-982 Keypair

参数	是否必选	参数类型	描述
name	否	String	SSH密钥对的名称
type	否	String	SSH密钥对的类型，值为“ssh”或“x509”

参数	是否必选	参数类型	描述
scope	否	String	租户级或者用户级。domain或user。
public_key	否	String	SSH密钥对对应的publicKey信息
fingerprint	否	String	SSH密钥对应指纹信息
is_key_protection	否	Boolean	是否托管密钥
frozen_state	否	String	冻结状态 ● 0：正常状态 ● 1：普通冻结 ● 2：公安冻结 ● 3：普通冻结及公安冻结 ● 4：违规冻结 ● 5：普通冻结及违规冻结 ● 6：公安冻结及违规冻结 ● 7：普通冻结、公安冻结及违规冻结 ● 8：未实名认证冻结 ● 9：普通冻结及未实名认证冻结 ● 10：公安冻结及未实名认证冻结

响应参数

无

请求示例

批量导出名称为”demo2，demo3, demo4"的SSH密钥对。

```
[ {
  "keypair" : {
    "name" : "demo2"
  }
}, {
  "keypair" : {
    "name" : "demo3"
  }
}, {
  "keypair" : {
    "name" : "demo4"
  }
} ]
```

响应示例

状态码： 200

请求已成功

```
"keypairs.zip"
```

状态码

状态码	描述
200	请求已成功

错误码

请参见**错误码**。

4.2.2 密钥对任务管理

4.2.2.1 绑定 SSH 密钥对

功能介绍

给指定的虚拟机绑定（替换或重置，替换需提供虚拟机已配置的SSH密钥对私钥；重置不需要提供虚拟机的SSH密钥对私钥）新的SSH密钥对。

调用方法

请参见**如何调用API**。

URI

POST /v3/{project_id}/keypairs/associate

表 4-983 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-984 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-985 请求 Body 参数

参数	是否必选	参数类型	描述
keypair_name	是	String	SSH密钥对的名称
server	是	**EcsServerInfo** object	需要绑定密钥对的虚拟机信息。

表 4-986 EcsServerInfo

参数	是否必选	参数类型	描述
id	是	String	需要绑定(替换或重置)SSH密钥对的虚拟机id
auth	否	**Auth** object	可选字段，鉴权认证类型。替换时需要该参数，重置时不需要该参数。
disable_password	否	Boolean	• true：禁用虚拟机的ssh登录。 • false：不禁用虚拟机的ssh登录。
port	否	Long	SSH监听端口。

表 4-987 Auth

参数	是否必选	参数类型	描述
type	否	String	取值为枚举类型。password或keypair。

参数	是否必选	参数类型	描述
key	否	String	• type为枚举值password时，key表示密码； • type为枚举值keypair时，key表示私钥；

响应参数

状态码： 202

表 4-988 响应 Body 参数

参数	参数类型	描述
task_id	String	任务下发成功返回的ID。
server_id	String	绑定的虚拟机id。
status	String	任务下发的状态。SUCCESS或FAILED。
error_code	String	任务下发失败返回的错误码。
error_msg	String	任务下发失败返回的错误信息。

状态码： 400

表 4-989 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

请求示例

- ```
  {
    "keypair_name" : "newkeypair1",
    "server" : {
      "id" : "d76baba7-ef09-40a2-87ff-3eafec0696e7",
      "auth" : {
        "type" : "keypair",
        "key" : "-----BEGINRSAPRIVATEKEY-----M..."
      }
    }
  }
  ```
- ```
  {
    "keypair_name" : "newkeypair2",
    "server" : {
      "id" : "d76baba7-ef09-40a2-87ff-3eafec0696e7"
  ```

```
  }
}
```

响应示例

状态码： 202

OK

```
{
  "task_id" : "aee8d2fe-5484-4753-9177-5a38dc15546c"
}
```

状态码： 400

Error response

```
{
  "error_code" : "KPS.XXX",
  "error_msg" : "XXX"
}
```

SDK 代码示例

SDK代码示例如下。

Java

- ```
  package com.huaweicloud.sdk.test;

  import com.huaweicloud.sdk.core.auth.ICredential;
  import com.huaweicloud.sdk.core.auth.BasicCredentials;
  import com.huaweicloud.sdk.core.exception.ConnectionException;
  import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
  import com.huaweicloud.sdk.core.exception.ServiceResponseException;
  import com.huaweicloud.sdk.kps.v3.region.KpsRegion;
  import com.huaweicloud.sdk.kps.v3.*;
  import com.huaweicloud.sdk.kps.v3.model.*;


  public class AssociateKeypairSolution {

      public static void main(String[] args) {
          // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
          // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
          String ak = System.getenv("CLOUD_SDK_AK");
          String sk = System.getenv("CLOUD_SDK_SK");
          String projectId = "{project_id}";

          ICredential auth = new BasicCredentials()
                  .withProjectId(projectId)
                  .withAk(ak)
                  .withSk(sk);

          KpsClient client = KpsClient.newBuilder()
                  .withCredential(auth)
                  .withRegion(KpsRegion.valueOf("<YOUR REGION>"))
                  .build();
          AssociateKeypairRequest request = new AssociateKeypairRequest();
          AssociateKeypairRequestBody body = new AssociateKeypairRequestBody();
          Auth authServer = new Auth();
          authServer.withType(Auth.TypeEnum.fromValue("keypair"))
              .withKey("-----BEGINRSAPRIVATEKEY-----M...");
  ```
  ```

```
        EcsServerInfo serverbody = new EcsServerInfo();
        serverbody.withId("d76baba7-ef09-40a2-87ff-3eafec0696e7")
            .withAuth(authServer);
        body.withServer(serverbody);
        body.withKeypairName("newkeypair1");
        request.withBody(body);
        try {
            AssociateKeypairResponse response = client.associateKeypair(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

- ```
  package com.huaweicloud.sdk.test;

  import com.huaweicloud.sdk.core.auth.ICredential;
  import com.huaweicloud.sdk.core.auth.BasicCredentials;
  import com.huaweicloud.sdk.core.exception.ConnectionException;
  import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
  import com.huaweicloud.sdk.core.exception.ServiceResponseException;
  import com.huaweicloud.sdk.kps.v3.region.KpsRegion;
  import com.huaweicloud.sdk.kps.v3.*;
  import com.huaweicloud.sdk.kps.v3.model.*;


  public class AssociateKeypairSolution {

      public static void main(String[] args) {
          // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
  security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
  environment variables and decrypted during use to ensure security.
          // In this example, AK and SK are stored in environment variables for authentication. Before
  running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local
  environment
          String ak = System.getenv("CLOUD_SDK_AK");
          String sk = System.getenv("CLOUD_SDK_SK");
          String projectId = "{project_id}";

          ICredential auth = new BasicCredentials()
                  .withProjectId(projectId)
                  .withAk(ak)
                  .withSk(sk);

          KpsClient client = KpsClient.newBuilder()
                  .withCredential(auth)
                  .withRegion(KpsRegion.valueOf("<YOUR REGION>"))
                  .build();
          AssociateKeypairRequest request = new AssociateKeypairRequest();
          AssociateKeypairRequestBody body = new AssociateKeypairRequestBody();
          EcsServerInfo serverbody = new EcsServerInfo();
          serverbody.withId("d76baba7-ef09-40a2-87ff-3eafec0696e7");
          body.withServer(serverbody);
          body.withKeypairName("newkeypair2");
          request.withBody(body);
          try {
              AssociateKeypairResponse response = client.associateKeypair(request);
              System.out.println(response.toString());
          } catch (ConnectionException e) {
              e.printStackTrace();
          } catch (RequestTimeoutException e) {
  ```
  ```

```
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

- ```
  # coding: utf-8

  import os
  from huaweicloudsdkcore.auth.credentials import BasicCredentials
  from huaweicloudsdkkps.v3.region.kps_region import KpsRegion
  from huaweicloudsdkcore.exceptions import exceptions
  from huaweicloudsdkkps.v3 import *

  if __name__ == "__main__":
      # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
      # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
      ak = os.environ["CLOUD_SDK_AK"]
      sk = os.environ["CLOUD_SDK_SK"]
      projectId = "{project_id}"

      credentials = BasicCredentials(ak, sk, projectId)

      client = KpsClient.new_builder() \
          .with_credentials(credentials) \
          .with_region(KpsRegion.value_of("<YOUR REGION>")) \
          .build()

      try:
          request = AssociateKeypairRequest()
          authServer = Auth(
              type="keypair",
              key="-----BEGINRSAPRIVATEKEY-----M..."
          )
          serverbody = EcsServerInfo(
              id="d76baba7-ef09-40a2-87ff-3eafec0696e7",
              auth=authServer
          )
          request.body = AssociateKeypairRequestBody(
              server=serverbody,
              keypair_name="newkeypair1"
          )
          response = client.associate_keypair(request)
          print(response)
      except exceptions.ClientRequestException as e:
          print(e.status_code)
          print(e.request_id)
          print(e.error_code)
          print(e.error_msg)
  ```
- ```
  # coding: utf-8

  import os
  from huaweicloudsdkcore.auth.credentials import BasicCredentials
  from huaweicloudsdkkps.v3.region.kps_region import KpsRegion
  from huaweicloudsdkcore.exceptions import exceptions
  from huaweicloudsdkkps.v3 import *

  if __name__ == "__main__":
  ```

```
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KpsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KpsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = AssociateKeypairRequest()
        serverbody = EcsServerInfo(
            id="d76baba7-ef09-40a2-87ff-3eafec0696e7"
        )
        request.body = AssociateKeypairRequestBody(
            server=serverbody,
            keypair_name="newkeypair2"
        )
        response = client.associate_keypair(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

- ```
  package main

  import (
      "fmt"
      "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
      kps "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3"
      "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3/model"
      region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3/region"
  )

  func main() {
      // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
      // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
      ak := os.Getenv("CLOUD_SDK_AK")
      sk := os.Getenv("CLOUD_SDK_SK")
      projectId := "{project_id}"

      auth := basic.NewCredentialsBuilder().
          WithAk(ak).
          WithSk(sk).
          WithProjectId(projectId).
          Build()

      client := kps.NewKpsClient(
          kps.KpsClientBuilder().
              WithRegion(region.ValueOf("<YOUR REGION>")).
              WithCredential(auth).
              Build())
  ```
  ```

```
    request := &model.AssociateKeypairRequest{}
    typeAuth:= model.GetAuthTypeEnum().KEYPAIR
    keyAuth:= "-----BEGINRSAPRIVATEKEY-----M..."
    authServer := &model.Auth{
        Type: &typeAuth,
        Key: &keyAuth,
    }
    serverbody := &model.EcsServerInfo{
        Id: "d76baba7-ef09-40a2-87ff-3eafec0696e7",
        Auth: authServer,
    }
    request.Body = &model.AssociateKeypairRequestBody{
        Server: serverbody,
        KeypairName: "newkeypair1",
    }
    response, err := client.AssociateKeypair(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

- ```
  package main

  import (
      "fmt"
      "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
      kps "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3"
      "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3/model"
      region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3/region"
  )

  func main() {
      // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
      // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
      ak := os.Getenv("CLOUD_SDK_AK")
      sk := os.Getenv("CLOUD_SDK_SK")
      projectId := "{project_id}"

      auth := basic.NewCredentialsBuilder().
          WithAk(ak).
          WithSk(sk).
          WithProjectId(projectId).
          Build()

      client := kps.NewKpsClient(
          kps.KpsClientBuilder().
              WithRegion(region.ValueOf("<YOUR REGION>")).
              WithCredential(auth).
              Build())

      request := &model.AssociateKeypairRequest{}
      serverbody := &model.EcsServerInfo{
          Id: "d76baba7-ef09-40a2-87ff-3eafec0696e7",
      }
      request.Body = &model.AssociateKeypairRequestBody{
          Server: serverbody,
          KeypairName: "newkeypair2",
      }
      response, err := client.AssociateKeypair(request)
      if err == nil {
          fmt.Printf("%+v\n", response)
      } else {
          fmt.Println(err)
  ```
  ```

```
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
202	OK
400	Error response

错误码

请参见**错误码**。

4.2.2.2 解绑 SSH 密钥对

功能介绍

给指定的虚拟机解除绑定SSH密钥对并恢复SSH密码登录。

调用方法

请参见**如何调用API**。

URI

POST /v3/{project_id}/keypairs/disassociate

表 4-990 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-991 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-992 请求 Body 参数

参数	是否必选	参数类型	描述
server	是	DisassociateEcsServerInfo object	需要绑定密钥对的虚拟机信息。

表 4-993 DisassociateEcsServerInfo

参数	是否必选	参数类型	描述
id	是	String	需要绑定(替换或重置)SSH密钥对的虚拟机id
auth	否	Auth object	可选字段，鉴权认证类型。替换时需要该参数，重置时不需要该参数。

表 4-994 Auth

参数	是否必选	参数类型	描述
type	否	String	取值为枚举类型。password或keypair。
key	否	String	• type为枚举值password时，key表示密码； • type为枚举值keypair时，key表示私钥；

响应参数

状态码： 202

表 4-995 响应 Body 参数

参数	参数类型	描述
task_id	String	任务下发成功返回的ID。
server_id	String	绑定的虚拟机id。
status	String	任务下发的状态。SUCCESS或FAILED。
error_code	String	任务下发失败返回的错误码。
error_msg	String	任务下发失败返回的错误信息。

状态码： 400

表 4-996 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

请求示例

- ```
  {
    "server" : {
      "id" : "d76baba7-ef09-40a2-87ff-3eafec0696e7",
      "auth" : {
        "type" : "keypair",
        "key" : "-----BEGINRSAPRIVATEKEY-----\nM..."
      }
    }
  }
  ```
- ```
  {
    "server" : {
      "id" : "x76baba7-ef09-40a2-87ff-3eafec0696e7"
    }
  }
  ```

响应示例

状态码： 202

OK

```
{
  "task_id" : "aee8d2fe-5484-4753-9177-5a38dc15546c"
}
```

状态码： 400

Error response

```
{
  "error_code" : "KPS.XXX",
```

```
  "error_msg" : "XXX"
}
```

SDK 代码示例

SDK代码示例如下。

Java

- ```
  package com.huaweicloud.sdk.test;

  import com.huaweicloud.sdk.core.auth.ICredential;
  import com.huaweicloud.sdk.core.auth.BasicCredentials;
  import com.huaweicloud.sdk.core.exception.ConnectionException;
  import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
  import com.huaweicloud.sdk.core.exception.ServiceResponseException;
  import com.huaweicloud.sdk.kps.v3.region.KpsRegion;
  import com.huaweicloud.sdk.kps.v3.*;
  import com.huaweicloud.sdk.kps.v3.model.*;


  public class DisassociateKeypairSolution {

      public static void main(String[] args) {
          // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
          // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
          String ak = System.getenv("CLOUD_SDK_AK");
          String sk = System.getenv("CLOUD_SDK_SK");
          String projectId = "{project_id}";

          ICredential auth = new BasicCredentials()
                  .withProjectId(projectId)
                  .withAk(ak)
                  .withSk(sk);

          KpsClient client = KpsClient.newBuilder()
                  .withCredential(auth)
                  .withRegion(KpsRegion.valueOf("<YOUR REGION>"))
                  .build();
          DisassociateKeypairRequest request = new DisassociateKeypairRequest();
          DisassociateKeypairRequestBody body = new DisassociateKeypairRequestBody();
          Auth authServer = new Auth();
          authServer.withType(Auth.TypeEnum.fromValue("keypair"))
              .withKey("-----BEGINRSAPRIVATEKEY-----
  M...");
          DisassociateEcsServerInfo serverbody = new DisassociateEcsServerInfo();
          serverbody.withId("d76baba7-ef09-40a2-87ff-3eafec0696e7")
              .withAuth(authServer);
          body.withServer(serverbody);
          request.withBody(body);
          try {
              DisassociateKeypairResponse response = client.disassociateKeypair(request);
              System.out.println(response.toString());
          } catch (ConnectionException e) {
              e.printStackTrace();
          } catch (RequestTimeoutException e) {
              e.printStackTrace();
          } catch (ServiceResponseException e) {
              e.printStackTrace();
              System.out.println(e.getHttpStatusCode());
              System.out.println(e.getRequestId());
              System.out.println(e.getErrorCode());
              System.out.println(e.getErrorMsg());
          }
  ```
  ```

```
    }
}
```

- ```
  package com.huaweicloud.sdk.test;

  import com.huaweicloud.sdk.core.auth.ICredential;
  import com.huaweicloud.sdk.core.auth.BasicCredentials;
  import com.huaweicloud.sdk.core.exception.ConnectionException;
  import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
  import com.huaweicloud.sdk.core.exception.ServiceResponseException;
  import com.huaweicloud.sdk.kps.v3.region.KpsRegion;
  import com.huaweicloud.sdk.kps.v3.*;
  import com.huaweicloud.sdk.kps.v3.model.*;


  public class DisassociateKeypairSolution {

      public static void main(String[] args) {
          // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
          // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
          String ak = System.getenv("CLOUD_SDK_AK");
          String sk = System.getenv("CLOUD_SDK_SK");
          String projectId = "{project_id}";

          ICredential auth = new BasicCredentials()
                  .withProjectId(projectId)
                  .withAk(ak)
                  .withSk(sk);

          KpsClient client = KpsClient.newBuilder()
                  .withCredential(auth)
                  .withRegion(KpsRegion.valueOf("<YOUR REGION>"))
                  .build();
          DisassociateKeypairRequest request = new DisassociateKeypairRequest();
          DisassociateKeypairRequestBody body = new DisassociateKeypairRequestBody();
          DisassociateEcsServerInfo serverbody = new DisassociateEcsServerInfo();
          serverbody.withId("x76baba7-ef09-40a2-87ff-3eafec0696e7");
          body.withServer(serverbody);
          request.withBody(body);
          try {
              DisassociateKeypairResponse response = client.disassociateKeypair(request);
              System.out.println(response.toString());
          } catch (ConnectionException e) {
              e.printStackTrace();
          } catch (RequestTimeoutException e) {
              e.printStackTrace();
          } catch (ServiceResponseException e) {
              e.printStackTrace();
              System.out.println(e.getHttpStatusCode());
              System.out.println(e.getRequestId());
              System.out.println(e.getErrorCode());
              System.out.println(e.getErrorMsg());
          }
      }
  }
  ```

Python

- ```
  # coding: utf-8

  import os
  from huaweicloudsdkcore.auth.credentials import BasicCredentials
  from huaweicloudsdkkps.v3.region.kps_region import KpsRegion
  from huaweicloudsdkcore.exceptions import exceptions
  from huaweicloudsdkkps.v3 import *
  ```

```
if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KpsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KpsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = DisassociateKeypairRequest()
        authServer = Auth(
            type="keypair",
            key="-----BEGINRSAPRIVATEKEY-----
        M..."
        )
        serverbody = DisassociateEcsServerInfo(
            id="d76baba7-ef09-40a2-87ff-3eafec0696e7",
            auth=authServer
        )
        request.body = DisassociateKeypairRequestBody(
            server=serverbody
        )
        response = client.disassociate_keypair(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

- ```
  # coding: utf-8

  import os
  from huaweicloudsdkcore.auth.credentials import BasicCredentials
  from huaweicloudsdkkps.v3.region.kps_region import KpsRegion
  from huaweicloudsdkcore.exceptions import exceptions
  from huaweicloudsdkkps.v3 import *

  if __name__ == "__main__":
      # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
      # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
      ak = os.environ["CLOUD_SDK_AK"]
      sk = os.environ["CLOUD_SDK_SK"]
      projectId = "{project_id}"

      credentials = BasicCredentials(ak, sk, projectId)

      client = KpsClient.new_builder() \
          .with_credentials(credentials) \
          .with_region(KpsRegion.value_of("<YOUR REGION>")) \
          .build()

      try:
          request = DisassociateKeypairRequest()
          serverbody = DisassociateEcsServerInfo(
              id="x76baba7-ef09-40a2-87ff-3eafec0696e7"
  ```
  ```

```
        )
        request.body = DisassociateKeypairRequestBody(
            server=serverbody
        )
        response = client.disassociate_keypair(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

- ```
  package main

  import (
      "fmt"
      "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
      kps "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3"
      "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3/model"
      region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3/region"
  )

  func main() {
      // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
      // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
      ak := os.Getenv("CLOUD_SDK_AK")
      sk := os.Getenv("CLOUD_SDK_SK")
      projectId := "{project_id}"

      auth := basic.NewCredentialsBuilder().
          WithAk(ak).
          WithSk(sk).
          WithProjectId(projectId).
          Build()

      client := kps.NewKpsClient(
          kps.KpsClientBuilder().
              WithRegion(region.ValueOf("<YOUR REGION>")).
              WithCredential(auth).
              Build())

      request := &model.DisassociateKeypairRequest{}
      typeAuth:= model.GetAuthTypeEnum().KEYPAIR
      keyAuth:= "-----BEGINRSAPRIVATEKEY-----
  M..."
      authServer := &model.Auth{
          Type: &typeAuth,
          Key: &keyAuth,
      }
      serverbody := &model.DisassociateEcsServerInfo{
          Id: "d76baba7-ef09-40a2-87ff-3eafec0696e7",
          Auth: authServer,
      }
      request.Body = &model.DisassociateKeypairRequestBody{
          Server: serverbody,
      }
      response, err := client.DisassociateKeypair(request)
      if err == nil {
          fmt.Printf("%+v\n", response)
      } else {
          fmt.Println(err)
      }
  }
  ```
  ```

- ```
  package main

  import (
      "fmt"
      "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
      kps "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3"
      "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3/model"
      region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3/region"
  )

  func main() {
      // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
      // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
      ak := os.Getenv("CLOUD_SDK_AK")
      sk := os.Getenv("CLOUD_SDK_SK")
      projectId := "{project_id}"

      auth := basic.NewCredentialsBuilder().
          WithAk(ak).
          WithSk(sk).
          WithProjectId(projectId).
          Build()

      client := kps.NewKpsClient(
          kps.KpsClientBuilder().
              WithRegion(region.ValueOf("<YOUR REGION>")).
              WithCredential(auth).
              Build())

      request := &model.DisassociateKeypairRequest{}
      serverbody := &model.DisassociateEcsServerInfo{
          Id: "x76baba7-ef09-40a2-87ff-3eafec0696e7",
      }
      request.Body = &model.DisassociateKeypairRequestBody{
          Server: serverbody,
      }
      response, err := client.DisassociateKeypair(request)
      if err == nil {
          fmt.Printf("%+v\n", response)
      } else {
          fmt.Println(err)
      }
  }
  ```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
202	OK
400	Error response
  ```

错误码

请参见**错误码**。

4.2.2.3 批量绑定 SSH 密钥对

功能介绍

给指定的虚拟机批量绑定新的SSH密钥对。

调用方法

请参见**如何调用API**。

URI

POST /v3/{project_id}/keypairs/batch-associate

表 4-997 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-998 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-999 请求 Body 参数

参数	是否必选	参数类型	描述
batch_keypairs	是	Array of **AssociateKeypairRequestBody** objects	最多可同时选择10个弹性云服务器绑定密钥对。 约束：只支持选择相同的密钥对，弹性云服务器处于“运行中”状态，并未绑定密钥对。

表 4-1000 AssociateKeypairRequestBody

参数	是否必选	参数类型	描述
keypair_name	是	String	SSH密钥对的名称
server	是	**EcsServerInfo** object	需要绑定密钥对的虚拟机信息。

表 4-1001 EcsServerInfo

参数	是否必选	参数类型	描述
id	是	String	需要绑定(替换或重置)SSH密钥对的虚拟机id
auth	否	**Auth** object	可选字段，鉴权认证类型。替换时需要该参数，重置时不需要该参数。
disable_password	否	Boolean	• true：禁用虚拟机的ssh登录。 • false：不禁用虚拟机的ssh登录。
port	否	Long	SSH监听端口。

表 4-1002 Auth

参数	是否必选	参数类型	描述
type	否	String	取值为枚举类型。password或keypair。
key	否	String	• type为枚举值password时，key表示密码； • type为枚举值keypair时，key表示私钥；

响应参数

状态码： 202

表 4-1003 响应 Body 参数

参数	参数类型	描述
tasks	Array of TaskResponseBody objects	批量绑定密钥对任务。

表 4-1004 TaskResponseBody

参数	参数类型	描述
task_id	String	任务下发成功返回的ID。
server_id	String	绑定的虚拟机id。
status	String	任务下发的状态。SUCCESS或FAILED。
error_code	String	任务下发失败返回的错误码。
error_msg	String	任务下发失败返回的错误信息。

状态码： 400

表 4-1005 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

请求示例

```
{
 "batch_keypairs" : [ {
  "keypair_name" : "1",
  "server" : {
   "id" : "fxxx16e3-74b8-4025-9852-1f451932c20c",
   "disable_password" : false,
   "auth" : {
    "type" : "password",
    "key" : "password"
   }
  }
 }, {
  "keypair_name" : "1",
  "server" : {
   "id" : "4xxxxfc4-b4bf-49c2-b983-a1811c9760c1",
   "disable_password" : false,
   "auth" : {
    "type" : "password",
    "key" : "password"
   }
  }
```

```
  } ]
}
```

响应示例

状态码： 202

请求已成功。

```
{
  "tasks" : [ {
    "server_id" : "xxx",
    "task_id" : "xxx",
    "status" : "SUCCESS"
  }, {
    "server_id" : "xxx",
    "status" : "Failed",
    "error_code" : "xxxx",
    "error_msg" : "xxxx"
  } ]
}
```

状态码： 400

Error response

```
{
  "error_code" : "KPS.XXX",
  "error_msg" : "XXX"
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kps.v3.region.KpsRegion;
import com.huaweicloud.sdk.kps.v3.*;
import com.huaweicloud.sdk.kps.v3.model.*;

import java.util.List;
import java.util.ArrayList;

public class BatchAssociateKeypairSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);
```

```
        KpsClient client = KpsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KpsRegion.valueOf("<YOUR REGION>"))
                .build();
        BatchAssociateKeypairRequest request = new BatchAssociateKeypairRequest();
        BatchAssociateKeypairRequestBody body = new BatchAssociateKeypairRequestBody();
        Auth authServer = new Auth();
        authServer.withType(Auth.TypeEnum.fromValue("password"))
            .withKey("password");
        EcsServerInfo serverBatchKeypairs = new EcsServerInfo();
        serverBatchKeypairs.withId("4xxxxfc4-b4bf-49c2-b983-a1811c9760c1")
            .withAuth(authServer)
            .withDisablePassword(false);
        Auth authServer1 = new Auth();
        authServer1.withType(Auth.TypeEnum.fromValue("password"))
            .withKey("password");
        EcsServerInfo serverBatchKeypairs1 = new EcsServerInfo();
        serverBatchKeypairs1.withId("fxxx16e3-74b8-4025-9852-1f451932c20c")
            .withAuth(authServer1)
            .withDisablePassword(false);
        List<AssociateKeypairRequestBody> listbodyBatchKeypairs = new ArrayList<>();
        listbodyBatchKeypairs.add(
            new AssociateKeypairRequestBody()
                .withKeypairName("1")
                .withServer(serverBatchKeypairs1)
        );
        listbodyBatchKeypairs.add(
            new AssociateKeypairRequestBody()
                .withKeypairName("1")
                .withServer(serverBatchKeypairs)
        );
        body.withBatchKeypairs(listbodyBatchKeypairs);
        request.withBody(body);
        try {
            BatchAssociateKeypairResponse response = client.batchAssociateKeypair(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkps.v3.region.kps_region import KpsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkps.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
```

```
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KpsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KpsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = BatchAssociateKeypairRequest()
        authServer = Auth(
            type="password",
            key="password"
        )
        serverBatchKeypairs = EcsServerInfo(
            id="4xxxxfc4-b4bf-49c2-b983-a1811c9760c1",
            auth=authServer,
            disable_password=False
        )
        authServer1 = Auth(
            type="password",
            key="password"
        )
        serverBatchKeypairs1 = EcsServerInfo(
            id="fxxx16e3-74b8-4025-9852-1f451932c20c",
            auth=authServer1,
            disable_password=False
        )
        listBatchKeypairsbody = [
            AssociateKeypairRequestBody(
                keypair_name="1",
                server=serverBatchKeypairs1
            ),
            AssociateKeypairRequestBody(
                keypair_name="1",
                server=serverBatchKeypairs
            )
        ]
        request.body = BatchAssociateKeypairRequestBody(
            batch_keypairs=listBatchKeypairsbody
        )
        response = client.batch_associate_keypair(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kps "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
```

```
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kps.NewKpsClient(
        kps.KpsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.BatchAssociateKeypairRequest{}
    typeAuth:= model.GetAuthTypeEnum().PASSWORD
    keyAuth:= "password"
    authServer := &model.Auth{
        Type: &typeAuth,
        Key: &keyAuth,
    }
    disablePasswordServer:= false
    serverBatchKeypairs := &model.EcsServerInfo{
        Id: "4xxxxfc4-b4bf-49c2-b983-a1811c9760c1",
        Auth: authServer,
        DisablePassword: &disablePasswordServer,
    }
    typeAuth1:= model.GetAuthTypeEnum().PASSWORD
    keyAuth1:= "password"
    authServer1 := &model.Auth{
        Type: &typeAuth1,
        Key: &keyAuth1,
    }
    disablePasswordServer1:= false
    serverBatchKeypairs1 := &model.EcsServerInfo{
        Id: "fxxx16e3-74b8-4025-9852-1f451932c20c",
        Auth: authServer1,
        DisablePassword: &disablePasswordServer1,
    }
    var listBatchKeypairsbody = []model.AssociateKeypairRequestBody{
        {
            KeypairName: "1",
            Server: serverBatchKeypairs1,
        },
        {
            KeypairName: "1",
            Server: serverBatchKeypairs,
        },
    }
    request.Body = &model.BatchAssociateKeypairRequestBody{
        BatchKeypairs: listBatchKeypairsbody,
    }
    response, err := client.BatchAssociateKeypair(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
202	请求已成功。
400	Error response

错误码

请参见**错误码**。

4.2.2.4 查询任务信息

功能介绍

根据SSH密钥对接口返回的task_id，查询SSH密钥对当前任务的执行状态。

调用方法

请参见**如何调用API**。

URI

GET /v3/{project_id}/tasks/{task_id}

表 4-1006 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID
task_id	是	String	任务ID

请求参数

表 4-1007 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码： 200

表 4-1008 响应 Body 参数

参数	参数类型	描述
server_id	String	租户虚拟机ID
task_id	String	任务下发成功返回的ID
task_status	String	密钥对正在处理的状态。 ● READY_RESET 准备重置 ● RUNNING_RESET 正在重置 ● FAILED_RESET 重置失败 ● SUCCESS_RESET 重置成功 ● READY_REPLACE 准备替换 ● RUNNING_REPLACE 正在替换 ● FAILED_RESET 替换失败 ● SUCCESS_RESET 替换成功 ● READY_UNBIND 准备解绑 ● RUNNING_UNBIND 正在解绑 ● FAILED_UNBIND 解绑失败 ● SUCCESS_UNBIND 解绑成功

状态码： 400

表 4-1009 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

请求示例

无

响应示例

状态码： 200

OK

```
{
  "task_id" : "aee8d2fe-5484-4753-9177-5a38dc15546c",
  "task_status" : "RUNNING_RESET",
  "server_id" : "c9aa197b-a6b6-4c33-b3a6-fa0b4ec50006"
}
```

状态码： 400

Error response

```
{
  "error_code" : "KPS.XXX",
  "error_msg" : "XXX"
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kps.v3.region.KpsRegion;
import com.huaweicloud.sdk.kps.v3.*;
import com.huaweicloud.sdk.kps.v3.model.*;


public class ListKeypairTaskSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KpsClient client = KpsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KpsRegion.valueOf("<YOUR REGION>"))
                .build();
        ListKeypairTaskRequest request = new ListKeypairTaskRequest();
        request.withTaskId("{task_id}");
        try {
            ListKeypairTaskResponse response = client.listKeypairTask(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkps.v3.region.kps_region import KpsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkps.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KpsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KpsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ListKeypairTaskRequest()
        request.task_id = "{task_id}"
        response = client.list_keypair_task(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kps "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kps.NewKpsClient(
        kps.KpsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
```

```
        WithCredential(auth).
        Build())

    request := &model.ListKeypairTaskRequest{}
    request.TaskId = "{task_id}"
    response, err := client.ListKeypairTask(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK
400	Error response

错误码

请参见**错误码**。

4.2.2.5 查询正在处理的任务信息

功能介绍

查询正在处理的任务信息。

调用方法

请参见**如何调用API**。

URI

GET /v3/{project_id}/running-tasks

表 4-1010 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

表 4-1011 Query 参数

参数	是否必选	参数类型	描述
limit	否	Integer	每页显示的条目数量。默认值1000。
offset	否	Integer	首个展示的正在处理任务信息的偏移量

请求参数

表 4-1012 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码： 200

表 4-1013 响应 Body 参数

参数	参数类型	描述
total	Integer	正在处理的任务总数。
tasks	Array of **RunningTasks** objects	正在处理的任务列表。

表 4-1014 RunningTasks

参数	参数类型	描述
task_id	String	虚拟机ID
operate_type	String	操作类型。 • FAILED_RESET 重置 • FAILED_REPLACE 替换 • FAILED_UNBIND 解绑
task_time	String	任务时间

参数	参数类型	描述
server_name	String	虚拟机名称
server_id	String	虚拟机ID
keypair_name	String	密钥对名称

状态码： 400

表 4-1015 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

请求示例

无

响应示例

状态码： 200

OK

```
{
  "total" : 1,
  "tasks" : [ {
    "task_id" : "aee8d2fe-5484-4753-9177-5a38dc15546c",
    "operate_type" : "RUNNING_RESET",
    "task_time" : "1523342130000",
    "server_name" : "Test",
    "server_id" : "c9aa197b-a6b6-4c33-b3a6-fa0b4ec50006",
    "keypair_name" : "KeyPair-xt"
  } ]
}
```

状态码： 400

Error response

```
{
  "error_code" : "KPS.XXX",
  "error_msg" : "XXX"
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;
```

```
import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kps.v3.region.KpsRegion;
import com.huaweicloud.sdk.kps.v3.*;
import com.huaweicloud.sdk.kps.v3.model.*;


public class ListRunningTaskSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KpsClient client = KpsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KpsRegion.valueOf("<YOUR REGION>"))
                .build();
        ListRunningTaskRequest request = new ListRunningTaskRequest();
        try {
            ListRunningTaskResponse response = client.listRunningTask(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkps.v3.region.kps_region import KpsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkps.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"
```

```
    credentials = BasicCredentials(ak, sk, projectId)

    client = KpsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KpsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ListRunningTaskRequest()
        response = client.list_running_task(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kps "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kps.NewKpsClient(
        kps.KpsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ListRunningTaskRequest{}
    response, err := client.ListRunningTask(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK
400	Error response

错误码

请参见**错误码**。

4.2.2.6 查询失败的任务信息

功能介绍

查询绑定、解绑等操作失败的任务信息。

调用方法

请参见**如何调用API**。

URI

GET /v3/{project_id}/failed-tasks

表 4-1016 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

表 4-1017 Query 参数

参数	是否必选	参数类型	描述
limit	否	Integer	每页显示的条目数量。默认值1000。
offset	否	Integer	失败的任务信息列表的偏移量

请求参数

表 4-1018 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码： 200

表 4-1019 响应 Body 参数

参数	参数类型	描述
total	Integer	失败任务总数。
tasks	Array of **FailedTasks** objects	失败的任务列表

表 4-1020 FailedTasks

参数	参数类型	描述
task_id	String	虚拟机ID
operate_type	String	任务的操作类型。 • FAILED_RESET 重置 • FAILED_REPLACE 替换 • FAILED_UNBIND 解绑
task_time	String	任务时间
task_error_code	String	任务失败错误码
task_error_msg	String	任务失败错误码
server_name	String	虚拟机名称
server_id	String	虚拟机ID
keypair_name	String	密钥对名称

状态码： 400

表 4-1021 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

请求示例

无

响应示例

状态码： 200

OK

```
{
  "total" : 1,
  "tasks" : [ {
    "task_id" : "aee8d2fe-5484-4753-9177-5a38dc15546c",
    "operate_type" : "RUNNING_RESET",
    "task_time" : "1523342130000",
    "task_error_code" : null,
    "task_error_msg" : "Update public key error",
    "server_name" : "Test",
    "server_id" : "c9aa197b-a6b6-4c33-b3a6-fa0b4ec50006",
    "keypair_name" : "KeyPair-xt"
  } ]
}
```

状态码： 400

Error response

```
{
  "error_code" : "KPS.XXX",
  "error_msg" : "XXX"
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kps.v3.region.KpsRegion;
import com.huaweicloud.sdk.kps.v3.*;
import com.huaweicloud.sdk.kps.v3.model.*;
```

```
public class ListFailedTaskSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KpsClient client = KpsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KpsRegion.valueOf("<YOUR REGION>"))
                .build();
        ListFailedTaskRequest request = new ListFailedTaskRequest();
        try {
            ListFailedTaskResponse response = client.listFailedTask(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkps.v3.region.kps_region import KpsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkps.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KpsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KpsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ListFailedTaskRequest()
        response = client.list_failed_task(request)
```

```
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kps "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kps.NewKpsClient(
        kps.KpsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ListFailedTaskRequest{}
    response, err := client.ListFailedTask(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK
400	Error response

错误码

请参见**错误码**。

4.2.2.7 删除所有失败的任务

功能介绍

删除操作失败的任务信息。

调用方法

请参见**如何调用API**。

URI

DELETE /v3/{project_id}/failed-tasks

表 4-1022 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-1023 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码： 400

表 4-1024 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

请求示例

无

响应示例

状态码： 400

Error response

```
{
  "error_code" : "KPS.XXX",
  "error_msg" : "XXX"
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.kps.v3.region.KpsRegion;
import com.huaweicloud.sdk.kps.v3.*;
import com.huaweicloud.sdk.kps.v3.model.*;


public class DeleteAllFailedTaskSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KpsClient client = KpsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KpsRegion.valueOf("<YOUR REGION>"))
                .build();
        DeleteAllFailedTaskRequest request = new DeleteAllFailedTaskRequest();
        try {
            DeleteAllFailedTaskResponse response = client.deleteAllFailedTask(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
```

```
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkps.v3.region.kps_region import KpsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkps.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KpsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(KpsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = DeleteAllFailedTaskRequest()
        response = client.delete_all_failed_task(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kps "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
```

```
        Build()

    client := kps.NewKpsClient(
        kps.KpsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.DeleteAllFailedTaskRequest{}
    response, err := client.DeleteAllFailedTask(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK
400	Error response

错误码

请参见**错误码**。

4.2.2.8 删除失败的任务

功能介绍

删除失败的任务。

调用方法

请参见**如何调用API**。

URI

DELETE /v3/{project_id}/failed-tasks/{task_id}

表 4-1025 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID
task_id	是	String	任务ID

请求参数

表 4-1026 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码： 400

表 4-1027 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

请求示例

无

响应示例

状态码： 400

Error response

```
{
  "error_code" : "KPS.XXX",
  "error_msg" : "XXX"
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
```

```
import com.huaweicloud.sdk.kps.v3.region.KpsRegion;
import com.huaweicloud.sdk.kps.v3.*;
import com.huaweicloud.sdk.kps.v3.model.*;


public class DeleteFailedTaskSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        KpsClient client = KpsClient.newBuilder()
                .withCredential(auth)
                .withRegion(KpsRegion.valueOf("<YOUR REGION>"))
                .build();
        DeleteFailedTaskRequest request = new DeleteFailedTaskRequest();
        request.withTaskId("{task_id}");
        try {
            DeleteFailedTaskResponse response = client.deleteFailedTask(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkkps.v3.region.kps_region import KpsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkkps.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = KpsClient.new_builder() \
        .with_credentials(credentials) \
```

```
        .with_region(KpsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = DeleteFailedTaskRequest()
        request.task_id = "{task_id}"
        response = client.delete_failed_task(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    kps "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/kps/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := kps.NewKpsClient(
        kps.KpsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.DeleteFailedTaskRequest{}
    request.TaskId = "{task_id}"
    response, err := client.DeleteFailedTask(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK
400	Error response

错误码

请参见**错误码**。

4.3 凭据管理服务

4.3.1 生命周期管理

4.3.1.1 创建凭据

功能介绍

创建新的凭据，并将凭据值存入凭据的初始版本。

凭据管理服务将凭据值加密后，存储在凭据对象下的版本中。每个版本可与多个凭据版本状态相关联，凭据版本状态用于标识凭据版本处于的阶段，没有版本状态标记的版本视为已弃用，可用凭据管理服务自动删除。

初始版本的状态被标记为SYSCURRENT。

接口约束

您可以指定一个对称密钥类型的用户主密钥作为保护凭据的加密密钥。当不指定kms_key_id参数时，凭据管理服务将默认使用名为csms/default的默认主密钥，用于加密您账号在本项目中创建的凭据。如果用户账号下不存在该名称的主密钥，则凭据管理服务会自动为您创建该名称的密钥。

如果您指定主密钥，则需要同时具备相应主密钥的kms:dek:create权限，用于凭据值进行加密。

调用方法

请参见**如何调用API**。

URI

POST /v1/{project_id}/secrets

表 4-1028 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。

请求参数

表 4-1029 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-1030 请求 Body 参数

参数	是否必选	参数类型	描述
name	是	String	待创建凭据的名称。 约束：取值范围为1到64个字符，满足正则匹配“^[a-zA-Z0-9_-]{1,64}$”。
kms_key_id	否	String	用于加密保护凭据值的KMS主密钥ID，如果您未指定此参数，凭据管理服务将默认使用名为csms/default的默认主密钥，用于加密您账号在本项目中创建的凭据值。如果用户账号下不存在该名称的主密钥，则凭据管理服务自动为您创建该名称的密钥。
description	否	String	凭据的描述信息。 约束：2048字节。
secret_binary	否	String	二进制类型凭据在base64编码后的明文，凭据管理服务将其加密后，存入凭据的初始版本中。 类型：base64编码的二进制数据对象。 约束：secret_binary和secret_string必须且只能设置一个，最大32K。

参数	是否必选	参数类型	描述
secret_string	否	String	文本类型凭据的明文，凭据管理服务将其加密后，存入凭据的初始版本中。 约束：secret_binary和secret_string必须且只能设置一个，最大32K。
secret_type	否	String	凭据类型 ● COMMON：通用凭据(默认)。用于应用系统中的各种敏感信息储存。 ● RDS：RDS凭据 。专门针对RDS的凭据，用于存储RDS的账号信息。（已不支持，使用RDS-FG替代） ● RDS-FG：RDS凭据 。专门针对RDS的凭据，用于存储RDS的账号信息。 ● GaussDB-FG：TaurusDB凭据。专门针对TaurusDB的凭据，用于存储TaurusDB的账号信息。
auto_rotation	否	Boolean	自动轮转 取值：true 开启 ,false 关闭 (默认)
rotation_period	否	String	轮转周期 约束：6小时-8,760小时 （365天） 类型：Integer[unit] ， Integer表示时间长度 。unit表示时间单位，d（天）、h（小时）、m（分钟）、s（秒）。例如 1d表示一天，24h也表示一天 说明：当开启自动轮转时，必须填写该值

参数	是否必选	参数类型	描述
rotation_config	否	String	轮转配置 约束：范围不超过1024个字符。 当secret_type为RDS-FG、GaussDB-FG时，必须填写本参数，参数格式为{"InstanceId":"","SecretSubType":""} 参数说明：InstanceId为实例ID,SecretSubType为轮转子类型，取值为：SingleUser，MultiUser。 SingleUser：指定轮转类型为单用户模式轮转，每次轮转将指定账号重置为新的口令。 MultiUser：指定轮转类型为双用户模式轮转，SYSCURRENT和SYSPREVIOUS分别引用其中一个账号。凭据轮转时，SYSPREVIOUS引用的账号口令会被重置为新的随机口令，随后凭据交换SYSCURRENT和SYSPREVIOUS对账号的引用。
event_subscriptions	否	Array of strings	凭据订阅的事件列表，当前最大可订阅一个事件。当事件包含的基础事件触发时，通知消息将发送到事件对应的通知主题。
enterprise_project_id	否	String	该参数针对企业用户使用。如果您是企业用户，且已创建企业项目，则请从下拉列表中为密钥选择需要绑定的企业项目，默认项目为“default”。 未开通企业管理的用户页面则没有“企业项目”参数项，无需进行配置。
rotation_func_urn	否	String	FunctionGraph函数的urn。

响应参数

状态码： 200

表 4-1031 响应 Body 参数

参数	参数类型	描述
secret	**Secret** object	凭据对象。

表 4-1032 Secret

参数	参数类型	描述
id	String	凭据的资源标识符。
name	String	凭据名称。
state	String	凭据状态，取值如下： ENABLED：表示启用状态 DISABLED：表示禁用状态 PENDING_DELETE：表示待删除状态 FROZEN：表示冻结状态
kms_key_id	String	用于加密凭据值的KMS主密钥的ID值。
description	String	凭据的描述信息。
create_time	Long	凭据创建时间，时间戳，即从1970年1月1日至该时间的总秒数。
update_time	Long	凭据上次更新时间，时间戳，即从1970年1月1日至该时间的总秒数。
scheduled_delete_time	Long	凭据计划删除时间，时间戳，即从1970年1月1日至该时间的总秒数。 凭据不在删除计划中时，本项值为null。
secret_type	String	凭据类型 • COMMON：通用凭据(默认)。用于应用系统中的各种敏感信息储存。 • RDS：RDS凭据 。专门针对RDS的凭据，用于存储RDS的账号信息。（已不支持，使用RDS-FG替代） • RDS-FG：RDS凭据 。专门针对RDS的凭据，用于存储RDS的账号信息。 • GaussDB-FG：TaurusDB凭据。专门针对TaurusDB的凭据，用于存储TaurusDB的账号信息。
auto_rotation	Boolean	自动轮转 取值：true 开启, false 关闭(默认)

参数	参数类型	描述
rotation_period	String	轮转周期 约束：6小时-8,760小时 （ 365天 ） 类型：Integer[unit] ，Integer表示时间长度 。unit表示时间单位，d（天）、h（小时）、m（分钟）、s（秒）。例如 1d 表示一天，24h也表示一天 说明：当开启自动轮转时，必须填写该值
rotation_config	String	轮转配置 约束：范围不超过1024个字符。 当secret_type为RDS-FG、GaussDB-FG时，配置为{"InstanceId":"","SecretSubType":""} 说明：当secret_type为RDS-FG、GaussDB-FG时，必须填写该值 InstanceId为实例ID,SecretSubType为轮转子类型，取值为：SingleUser，MultiUser。 SingleUser：指定轮转类型为单用户模式轮转，每次轮转将指定账号重置为新的口令。 MultiUser：指定轮转类型为双用户模式轮转，SYSCURRENT和SYSPREVIOUS分别引用其中一个账号。凭据轮转时，SYSPREVIOUS引用的账号口令会被重置为新的随机口令，随后凭据交换SYSCURRENT和SYSPREVIOUS对账号的引用。
rotation_time	Long	轮转时间戳
next_rotation_time	Long	下一次轮转时间戳
event_subscriptions	Array of strings	凭据订阅的事件列表，当前最大可订阅一个事件。当事件包含的基础事件触发时，通知消息将发送到事件对应的通知主题。
enterprise_project_id	String	企业项目ID
rotation_func_urn	String	FunctionGraph函数的urn。

状态码： 400

表 4-1033 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 401

表 4-1034 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 403

表 4-1035 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 404

表 4-1036 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 500

表 4-1037 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 502

表 4-1038 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码

参数	参数类型	描述
error_msg	String	错误描述

状态码： 504

表 4-1039 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

请求示例

创建一个名字为demo的凭据，使用kms密钥ID为0d0466b0-e727-4d9c-b35d-f84bb474a37f对“this is a demo secret string”凭据值加密。

```
{
  "name" : "demo",
  "kms_key_id" : "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
  "secret_string" : "this is a demo secret string"
}
```

响应示例

状态码： 200

请求已成功

```
{
  "secret" : {
    "id" : "bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e",
    "name" : "test",
    "state" : "ENABLED",
    "kms_key_id" : "b168fe00ff56492495a7d22974df2d0b",
    "description" : "description",
    "create_time" : 1581507580000,
    "update_time" : 1581507580000,
    "scheduled_delete_time" : 1581507580000,
    "secret_type" : "RDS-FG",
    "auto_rotation" : true,
    "rotation_config" : "{'InstanceId':'63616bceef2c45409575d762a498318bin01','SecretSubType':'MultiUser'}",
    "rotation_period" : "1d",
    "rotation_time" : 1668567940000,
    "next_rotation_time" : 1668629140000,
    "event_subscriptions" : [ "pocEvent" ],
    "rotation_func_urn" : "urn:fss:{region}:46b6f338fc3445b8846c71dfb1fbxxxx:function:default:test2-0:latest"
  }
}
```

SDK 代码示例

SDK代码示例如下。

Java

创建一个名字为demo的凭据，使用kms密钥ID为0d0466b0-e727-4d9c-b35d-f84bb474a37f对“this is a demo secret string”凭据值加密。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.csms.v1.region.CsmsRegion;
import com.huaweicloud.sdk.csms.v1.*;
import com.huaweicloud.sdk.csms.v1.model.*;


public class CreateSecretSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CsmsClient client = CsmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(CsmsRegion.valueOf("<YOUR REGION>"))
                .build();
        CreateSecretRequest request = new CreateSecretRequest();
        CreateSecretRequestBody body = new CreateSecretRequestBody();
        body.withSecretString("this is a demo secret string");
        body.withKmsKeyId("0d0466b0-e727-4d9c-b35d-f84bb474a37f");
        body.withName("demo");
        request.withBody(body);
        try {
            CreateSecretResponse response = client.createSecret(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

创建一个名字为demo的凭据，使用kms密钥ID为0d0466b0-e727-4d9c-b35d-f84bb474a37f对“this is a demo secret string”凭据值加密。

```
# coding: utf-8
```

```
import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcsms.v1.region.csms_region import CsmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcsms.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CsmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CsmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = CreateSecretRequest()
        request.body = CreateSecretRequestBody(
            secret_string="this is a demo secret string",
            kms_key_id="0d0466b0-e727-4d9c-b35d-f84bb474a37f",
            name="demo"
        )
        response = client.create_secret(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

创建一个名字为demo的凭据，使用kms密钥ID为0d0466b0-e727-4d9c-b35d-f84bb474a37f对“this is a demo secret string”凭据值加密。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    csms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()
```

```
    client := csms.NewCsmsClient(
        csms.CsmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.CreateSecretRequest{}
    secretStringCreateSecretRequestBody:= "this is a demo secret string"
    kmsKeyIdCreateSecretRequestBody:= "0d0466b0-e727-4d9c-b35d-f84bb474a37f"
    request.Body = &model.CreateSecretRequestBody{
        SecretString: &secretStringCreateSecretRequestBody,
        KmsKeyId: &kmsKeyIdCreateSecretRequestBody,
        Name: "demo",
    }
    response, err := client.CreateSecret(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.3.1.2 查询凭据列表

功能介绍

查询当前用户在本项目下创建的所有凭据。

接口约束

此接口返回的信息为凭据的元数据信息，不包含凭据值。

调用方法

请参见**如何调用API**。

URI

GET /v1/{project_id}/secrets

表 4-1040 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。

表 4-1041 Query 参数

参数	是否必选	参数类型	描述
limit	否	String	每页返回的个数。 默认值：50。
marker	否	String	分页查询起始的凭据名称，为空时为查询第一页
event_name	否	String	指定事件名称时，仅返回关联该事件的凭据

请求参数

表 4-1042 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码： 200

表 4-1043 响应 Body 参数

参数	参数类型	描述
secrets	Array of **Secret** objects	凭据详情列表。
page_info	**PageInfo** object	分页信息。

表 4-1044 Secret

参数	参数类型	描述
id	String	凭据的资源标识符。
name	String	凭据名称。
state	String	凭据状态，取值如下： ENABLED：表示启用状态 DISABLED：表示禁用状态 PENDING_DELETE：表示待删除状态 FROZEN：表示冻结状态
kms_key_id	String	用于加密凭据值的KMS主密钥的ID值。
description	String	凭据的描述信息。
create_time	Long	凭据创建时间，时间戳，即从1970年1月1日至该时间的总秒数。
update_time	Long	凭据上次更新时间，时间戳，即从1970年1月1日至该时间的总秒数。
scheduled_delete_time	Long	凭据计划删除时间，时间戳，即从1970年1月1日至该时间的总秒数。 凭据不在删除计划中时，本项值为null。
secret_type	String	凭据类型 • COMMON：通用凭据(默认)。用于应用系统中的各种敏感信息储存。 • RDS：RDS凭据 。专门针对RDS的凭据，用于存储RDS的账号信息。（已不支持，使用RDS-FG替代） • RDS-FG：RDS凭据 。专门针对RDS的凭据，用于存储RDS的账号信息。 • GaussDB-FG：TaurusDB凭据。专门针对TaurusDB的凭据，用于存储TaurusDB的账号信息。

参数	参数类型	描述
auto_rotation	Boolean	自动轮转 取值：true 开启, false 关闭(默认)
rotation_period	String	轮转周期 约束：6小时-8,760小时 （ 365天 ） 类型：Integer[unit] ， Integer表示时间长度 。unit表示时间单位，d（天）、h（小时）、m（分钟）、s（秒）。例如 1d 表示一天，24h也表示一天 说明：当开启自动轮转时，必须填写该值
rotation_config	String	轮转配置 约束：范围不超过1024个字符。 当secret_type为RDS-FG、GaussDB-FG时，配置为{"InstanceId":"","SecretSubType":""} 说明：当secret_type为RDS-FG、GaussDB-FG时，必须填写该值 InstanceId为实例ID,SecretSubType为轮转子类型，取值为：SingleUser，MultiUser。 SingleUser：指定轮转类型为单用户模式轮转，每次轮转将指定账号重置为新的口令。 MultiUser：指定轮转类型为双用户模式轮转，SYSCURRENT和SYSPREVIOUS分别引用其中一个账号。凭据轮转时，SYSPREVIOUS引用的账号口令会被重置为新的随机口令，随后凭据交换SYSCURRENT和SYSPREVIOUS对账号的引用。
rotation_time	Long	轮转时间戳
next_rotation_time	Long	下一次轮转时间戳
event_subscriptions	Array of strings	凭据订阅的事件列表，当前最大可订阅一个事件。当事件包含的基础事件触发时，通知消息将发送到事件对应的通知主题。
enterprise_project_id	String	企业项目ID
rotation_func_urn	String	FunctionGraph函数的urn。

表 4-1045 PageInfo

参数	参数类型	描述
next_marker	String	下一页查询地址（本页的末尾凭据名称，下一页起始凭据名称）。

参数	参数类型	描述
previous_marker	String	本页的起始凭据名称，上一页末尾凭据名称。
current_count	Integer	本页返回条目数量。

状态码： 400

表 4-1046 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 401

表 4-1047 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 403

表 4-1048 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 404

表 4-1049 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 500

表 4-1050 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 502

表 4-1051 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 504

表 4-1052 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

请求示例

无

响应示例

状态码： 200

请求已成功

```
{
  "secrets" : [ {
    "id" : "bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e",
    "name" : "secret-name-test",
    "state" : "ENABLED",
    "kms_key_id" : "b168fe00ff56492495a7d22974df2d0b",
    "description" : "description",
    "create_time" : 1581507580000,
    "update_time" : 1581507580000,
    "scheduled_delete_time" : 1581507580000,
    "secret_type" : "RDS-FG",
    "auto_rotation" : true,
    "rotation_config" : "{'InstanceId':'indstance id','SecretSubType':'MultiUser'}",
```

```
    "rotation_period" : "1d",
    "rotation_time" : 1668567940000,
    "next_rotation_time" : 1668629140000,
    "event_subscriptions" : [ "pocEvent" ]
  } ],
  "page_info" : {
    "next_marker" : "secret-name-test",
    "previous_marker" : "secret-name-test",
    "current_count" : 1
  }
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.csms.v1.region.CsmsRegion;
import com.huaweicloud.sdk.csms.v1.*;
import com.huaweicloud.sdk.csms.v1.model.*;


public class ListSecretsSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CsmsClient client = CsmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(CsmsRegion.valueOf("<YOUR REGION>"))
                .build();
        ListSecretsRequest request = new ListSecretsRequest();
        try {
            ListSecretsResponse response = client.listSecrets(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcsms.v1.region.csms_region import CsmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcsms.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CsmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CsmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ListSecretsRequest()
        response = client.list_secrets(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    csms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := csms.NewCsmsClient(
        csms.CsmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
```

```
        Build())

    request := &model.ListSecretsRequest{}
    response, err := client.ListSecrets(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.3.1.3 查询凭据

功能介绍

查询指定凭据的信息。

接口约束

此接口返回的信息为凭据的元数据信息，不包含凭据值。

调用方法

请参见**如何调用API**。

URI

GET /v1/{project_id}/secrets/{secret_name}

表 4-1053 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。
secret_name	是	String	凭据的名称。

请求参数

表 4-1054 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码： 200

表 4-1055 响应 Body 参数

参数	参数类型	描述
secret	**Secret** object	凭据对象。

表 4-1056 Secret

参数	参数类型	描述
id	String	凭据的资源标识符。
name	String	凭据名称。
state	String	凭据状态，取值如下： ENABLED：表示启用状态 DISABLED：表示禁用状态 PENDING_DELETE：表示待删除状态 FROZEN：表示冻结状态

参数	参数类型	描述
kms_key_id	String	用于加密凭据值的KMS主密钥的ID值。
description	String	凭据的描述信息。
create_time	Long	凭据创建时间，时间戳，即从1970年1月1日至该时间的总秒数。
update_time	Long	凭据上次更新时间，时间戳，即从1970年1月1日至该时间的总秒数。
scheduled_delete_time	Long	凭据计划删除时间，时间戳，即从1970年1月1日至该时间的总秒数。 凭据不在删除计划中时，本项值为null。
secret_type	String	凭据类型 ● COMMON：通用凭据(默认)。用于应用系统中的各种敏感信息储存。 ● RDS：RDS凭据 。专门针对RDS的凭据，用于存储RDS的账号信息。（ 已不支持，使用RDS-FG替代 ） ● RDS-FG：RDS凭据 。专门针对RDS的凭据，用于存储RDS的账号信息。 ● GaussDB-FG：TaurusDB凭据。专门针对TaurusDB的凭据，用于存储TaurusDB的账号信息。
auto_rotation	Boolean	自动轮转 取值：true 开启, false 关闭(默认)
rotation_period	String	轮转周期 约束：6小时-8,760小时 （ 365天 ） 类型：Integer[unit] ， Integer表示时间长度 。unit表示时间单位，d（ 天 ）、h（ 小时 ）、m（ 分钟 ）、s（ 秒 ）。例如 1d 表示一天，24h也表示一天 说明：当开启自动轮转时，必须填写该值

参数	参数类型	描述
rotation_config	String	轮转配置 约束：范围不超过1024个字符。 当secret_type为RDS-FG、GaussDB-FG时，配置为{"InstanceId":"","SecretSubType":""} 说明：当secret_type为RDS-FG、GaussDB-FG时，必须填写该值 InstanceId为实例ID,SecretSubType为轮转子类型，取值为：SingleUser，MultiUser。 SingleUser：指定轮转类型为单用户模式轮转，每次轮转将指定账号重置为新的口令。 MultiUser：指定轮转类型为双用户模式轮转，SYSCURRENT和SYSPREVIOUS分别引用其中一个账号。凭据轮转时，SYSPREVIOUS引用的账号口令会被重置为新的随机口令，随后凭据交换SYSCURRENT和SYSPREVIOUS对账号的引用。
rotation_time	Long	轮转时间戳
next_rotation_time	Long	下一次轮转时间戳
event_subscriptions	Array of strings	凭据订阅的事件列表，当前最大可订阅一个事件。当事件包含的基础事件触发时，通知消息将发送到事件对应的通知主题。
enterprise_project_id	String	企业项目ID
rotation_func_urn	String	FunctionGraph函数的urn。

状态码： 400

表 4-1057 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 401

表 4-1058 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 403

表 4-1059 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 404

表 4-1060 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 500

表 4-1061 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 502

表 4-1062 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 504

表 4-1063 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

请求示例

无

响应示例

状态码： 200

请求已成功

```
{
  "secret" : {
    "id" : "bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e",
    "name" : "test",
    "state" : "ENABLED",
    "kms_key_id" : "b168fe00ff56492495a7d22974df2d0b",
    "description" : "description",
    "create_time" : 1581507580000,
    "update_time" : 1581507580000,
    "scheduled_delete_time" : 1581507580000,
    "secret_type" : "RDS-FG",
    "auto_rotation" : true,
    "rotation_config" : "{'InstanceId':'63616bceef2c45409575d762a498318bin01','SecretSubType':'MultiUser'}",
    "rotation_period" : "1d",
    "rotation_time" : 1668567940000,
    "next_rotation_time" : 1668629140000,
    "event_subscriptions" : [ "pocEvent" ],
    "rotation_func_urn" : "urn:fss:{region}:46b6f338fc3445b8846c71dfb1fbxxxx:function:default:test2-0:latest"
  }
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.csms.v1.region.CsmsRegion;
import com.huaweicloud.sdk.csms.v1.*;
import com.huaweicloud.sdk.csms.v1.model.*;


public class ShowSecretSolution {
```

```
    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CsmsClient client = CsmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(CsmsRegion.valueOf("<YOUR REGION>"))
                .build();
        ShowSecretRequest request = new ShowSecretRequest();
        request.withSecretName("{secret_name}");
        try {
            ShowSecretResponse response = client.showSecret(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcsms.v1.region.csms_region import CsmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcsms.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CsmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CsmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ShowSecretRequest()
        request.secret_name = "{secret_name}"
        response = client.show_secret(request)
```

```
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    csms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := csms.NewCsmsClient(
        csms.CsmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ShowSecretRequest{}
    request.SecretName = "{secret_name}"
    response, err := client.ShowSecret(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码

状态码	描述
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.3.1.4 更新凭据

功能介绍

更新指定凭据的元数据信息。

接口约束

此接口仅能修改凭据的元数据信息，无法修改凭据值。

调用方法

请参见**如何调用API**。

URI

PUT /v1/{project_id}/secrets/{secret_name}

表 4-1064 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。
secret_name	是	String	凭据名称。

请求参数

表 4-1065 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-1066 请求 Body 参数

参数	是否必选	参数类型	描述
kms_key_id	否	String	用于加密保护凭据值的KMS主密钥ID。更新凭据的主密钥后，仅新创建的凭据版本使用更新后的主密钥ID加密，之前的凭据版本依旧使用之前的主密钥ID解密。
description	否	String	凭据的描述信息。 约束：2048字节。
auto_rotation	否	Boolean	自动轮转 取值：true 开启 false 关
rotation_period	否	String	轮转周期 约束：6小时-8,760小时（365天） 类型：Integer[unit]，Integer表示时间长度 。unit表示时间单位，d（天）、h（小时）、m（分钟）、s（秒）。例如 1d表示一天，24h也表示一天 说明：当开启自动轮转时，必须填写该值
event_subscriptions	否	Array of strings	凭据订阅的事件列表，当前最大可订阅一个事件。当事件包含的基础事件触发时，通知消息将发送到事件对应的通知主题。
rotation_func_urn	否	String	FunctionGraph函数的urn。

响应参数

状态码： 200

表 4-1067 响应 Body 参数

参数	参数类型	描述
secret	**Secret** object	凭据对象。

表 4-1068 Secret

参数	参数类型	描述
id	String	凭据的资源标识符。
name	String	凭据名称。
state	String	凭据状态，取值如下： ENABLED：表示启用状态 DISABLED：表示禁用状态 PENDING_DELETE：表示待删除状态 FROZEN：表示冻结状态
kms_key_id	String	用于加密凭据值的KMS主密钥的ID值。
description	String	凭据的描述信息。
create_time	Long	凭据创建时间，时间戳，即从1970年1月1日至该时间的总秒数。
update_time	Long	凭据上次更新时间，时间戳，即从1970年1月1日至该时间的总秒数。
scheduled_delete_time	Long	凭据计划删除时间，时间戳，即从1970年1月1日至该时间的总秒数。 凭据不在删除计划中时，本项值为null。
secret_type	String	凭据类型 • COMMON：通用凭据(默认)。用于应用系统中的各种敏感信息储存。 • RDS：RDS凭据 。专门针对RDS的凭据，用于存储RDS的账号信息。（已不支持，使用RDS-FG替代） • RDS-FG：RDS凭据 。专门针对RDS的凭据，用于存储RDS的账号信息。 • GaussDB-FG：TaurusDB凭据。专门针对TaurusDB的凭据，用于存储TaurusDB的账号信息。
auto_rotation	Boolean	自动轮转 取值：true 开启, false 关闭(默认)

参数	参数类型	描述
rotation_period	String	轮转周期 约束：6小时-8,760小时 （ 365天 ） 类型：Integer[unit] ，Integer表示时间长度 。unit表示时间单位，d（天）、h（小时）、m（分钟）、s（秒）。例如 1d 表示一天，24h也表示一天 说明：当开启自动轮转时，必须填写该值
rotation_config	String	轮转配置 约束：范围不超过1024个字符。 当secret_type为RDS-FG、GaussDB-FG时，配置为{"InstanceId":"","SecretSubType":""} 说明：当secret_type为RDS-FG、GaussDB-FG时，必须填写该值 InstanceId为实例ID,SecretSubType为轮转子类型，取值为：SingleUser，MultiUser。 SingleUser：指定轮转类型为单用户模式轮转，每次轮转将指定账号重置为新的口令。 MultiUser：指定轮转类型为双用户模式轮转，SYSCURRENT和SYSPREVIOUS分别引用其中一个账号。凭据轮转时，SYSPREVIOUS引用的账号口令会被重置为新的随机口令，随后凭据交换SYSCURRENT和SYSPREVIOUS对账号的引用。
rotation_time	Long	轮转时间戳
next_rotation_time	Long	下一次轮转时间戳
event_subscriptions	Array of strings	凭据订阅的事件列表，当前最大可订阅一个事件。当事件包含的基础事件触发时，通知消息将发送到事件对应的通知主题。
enterprise_project_id	String	企业项目ID
rotation_func_urn	String	FunctionGraph函数的urn。

状态码： 400

表 4-1069 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 401

表 4-1070 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 403

表 4-1071 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 404

表 4-1072 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 500

表 4-1073 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 502

表 4-1074 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码

参数	参数类型	描述
error_msg	String	错误描述

状态码： 504

表 4-1075 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

请求示例

更新凭据KMS密钥ID为test，描述为"update description"。

```
{
  "kms_key_id" : "test",
  "description" : "update description",
  "event_subscriptions" : [ "pocEvent2" ]
}
```

响应示例

状态码： 200

请求已成功

```
{
  "secret" : {
    "id" : "bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e",
    "name" : "test",
    "state" : "ENABLED",
    "kms_key_id" : "b168fe00ff56492495a7d22974df2d0b",
    "description" : "description",
    "create_time" : 1581507580000,
    "update_time" : 1581507580000,
    "scheduled_delete_time" : 1581507580000,
    "secret_type" : "RDS-FG",
    "auto_rotation" : true,
    "rotation_config" : "{'InstanceId':'indstance id','SecretSubType':'MultiUser'}",
    "rotation_period" : "1d",
    "rotation_time" : 1668567940000,
    "next_rotation_time" : 1668629140000,
    "event_subscriptions" : [ "pocEvent" ],
    "rotation_func_urn" : "urn:fss:{region}:46b6f338fc3445b8846c71dfb1fbxxxx:function:default:test2-0:latest"
  }
}
```

SDK 代码示例

SDK代码示例如下。

Java

更新凭据KMS密钥ID为test，描述为"update description"。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.csms.v1.region.CsmsRegion;
import com.huaweicloud.sdk.csms.v1.*;
import com.huaweicloud.sdk.csms.v1.model.*;

import java.util.List;
import java.util.ArrayList;

public class UpdateSecretSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CsmsClient client = CsmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(CsmsRegion.valueOf("<YOUR REGION>"))
                .build();
        UpdateSecretRequest request = new UpdateSecretRequest();
        request.withSecretName("{secret_name}");
        UpdateSecretRequestBody body = new UpdateSecretRequestBody();
        List<String> listbodyEventSubscriptions = new ArrayList<>();
        listbodyEventSubscriptions.add("pocEvent2");
        body.withEventSubscriptions(listbodyEventSubscriptions);
        body.withDescription("update description");
        body.withKmsKeyId("test");
        request.withBody(body);
        try {
            UpdateSecretResponse response = client.updateSecret(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

更新凭据KMS密钥ID为test，描述为"update description"。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcsms.v1.region.csms_region import CsmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcsms.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CsmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CsmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = UpdateSecretRequest()
        request.secret_name = "{secret_name}"
        listEventSubscriptionsbody = [
            "pocEvent2"
        ]
        request.body = UpdateSecretRequestBody(
            event_subscriptions=listEventSubscriptionsbody,
            description="update description",
            kms_key_id="test"
        )
        response = client.update_secret(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

更新凭据KMS密钥ID为test，描述为"update description"。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    csms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
```

```
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := csms.NewCsmsClient(
        csms.CsmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.UpdateSecretRequest{}
    request.SecretName = "{secret_name}"
    var listEventSubscriptionsbody = []string{
        "pocEvent2",
    }
    descriptionUpdateSecretRequestBody:= "update description"
    kmsKeyIdUpdateSecretRequestBody:= "test"
    request.Body = &model.UpdateSecretRequestBody{
        EventSubscriptions: &listEventSubscriptionsbody,
        Description: &descriptionUpdateSecretRequestBody,
        KmsKeyId: &kmsKeyIdUpdateSecretRequestBody,
    }
    response, err := client.UpdateSecret(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.3.1.5 立即删除凭据

功能介绍

立即删除指定的凭据，且无法恢复。

接口约束

调用此接口删除指定凭据后，不可恢复。

调用方法

请参见**如何调用API**。

URI

DELETE /v1/{project_id}/secrets/{secret_name}

表 4-1076 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。
secret_name	是	String	凭据名称。

请求参数

表 4-1077 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码： 400

表 4-1078 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 401

表 4-1079 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 403

表 4-1080 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 404

表 4-1081 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 500

表 4-1082 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 502

表 4-1083 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码

参数	参数类型	描述
error_msg	String	错误描述

状态码： 504

表 4-1084 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

请求示例

无

响应示例

无

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.3.1.6 恢复凭据对象

功能介绍

通过上传凭据备份文件，恢复凭据对象

接口约束

此接口返回的信息为凭据的元数据信息，不包含凭据值。

调用方法

请参见**如何调用API**。

URI

POST /v1/{project_id}/secrets/restore

表 4-1085 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。

请求参数

表 4-1086 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-1087 请求 Body 参数

参数	是否必选	参数类型	描述
secret_blob	是	String	将指定凭据对象进行备份后得到的凭据备份文件，备份文件包含有凭据当前所有的凭据版本信息，备份文件经过加密与编码，内容不可直接读。

响应参数

状态码： 200

表 4-1088 响应 Body 参数

参数	参数类型	描述
secret	**Secret** object	凭据对象。

表 4-1089 Secret

参数	参数类型	描述
id	String	凭据的资源标识符。
name	String	凭据名称。
state	String	凭据状态，取值如下： ENABLED：表示启用状态 DISABLED：表示禁用状态 PENDING_DELETE：表示待删除状态 FROZEN：表示冻结状态
kms_key_id	String	用于加密凭据值的KMS主密钥的ID值。
description	String	凭据的描述信息。
create_time	Long	凭据创建时间，时间戳，即从1970年1月1日至该时间的总秒数。
update_time	Long	凭据上次更新时间，时间戳，即从1970年1月1日至该时间的总秒数。
scheduled_delete_time	Long	凭据计划删除时间，时间戳，即从1970年1月1日至该时间的总秒数。 凭据不在删除计划中时，本项值为null。
secret_type	String	凭据类型 • COMMON：通用凭据(默认)。用于应用系统中的各种敏感信息储存。 • RDS：RDS凭据 。专门针对RDS的凭据，用于存储RDS的账号信息。（已不支持，使用RDS-FG替代） • RDS-FG：RDS凭据 。专门针对RDS的凭据，用于存储RDS的账号信息。 • GaussDB-FG：TaurusDB凭据。专门针对TaurusDB的凭据，用于存储TaurusDB的账号信息。
auto_rotation	Boolean	自动轮转 取值：true 开启, false 关闭(默认)

参数	参数类型	描述
rotation_period	String	轮转周期 约束：6小时-8,760小时 （ 365天 ） 类型：Integer[unit] ，Integer表示时间长度 。unit表示时间单位，d（天）、h（小时）、m（分钟）、s（秒）。例如 1d 表示一天，24h也表示一天 说明：当开启自动轮转时，必须填写该值
rotation_config	String	轮转配置 约束：范围不超过1024个字符。 当secret_type为RDS-FG、GaussDB-FG时，配置为{"InstanceId":"","SecretSubType":""} 说明：当secret_type为RDS-FG、GaussDB-FG时，必须填写该值 InstanceId为实例ID,SecretSubType为轮转子类型，取值为：SingleUser，MultiUser。 SingleUser：指定轮转类型为单用户模式轮转，每次轮转将指定账号重置为新的口令。 MultiUser：指定轮转类型为双用户模式轮转，SYSCURRENT和SYSPREVIOUS分别引用其中一个账号。凭据轮转时，SYSPREVIOUS引用的账号口令会被重置为新的随机口令，随后凭据交换SYSCURRENT和SYSPREVIOUS对账号的引用。
rotation_time	Long	轮转时间戳
next_rotation_time	Long	下一次轮转时间戳
event_subscriptions	Array of strings	凭据订阅的事件列表，当前最大可订阅一个事件。当事件包含的基础事件触发时，通知消息将发送到事件对应的通知主题。
enterprise_project_id	String	企业项目ID
rotation_func_urn	String	FunctionGraph函数的urn。

状态码： 400

表 4-1090 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 401

表 4-1091 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 403

表 4-1092 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 404

表 4-1093 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 500

表 4-1094 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 502

表 4-1095 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码

参数	参数类型	描述
error_msg	String	错误描述

状态码： 504

表 4-1096 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

请求示例

上传凭据备份文件。

```
{
  "secret_blob" :
")CloudSecretManagementBackupV1.comeyJraWQiOiI5ZjNlZmRjNS0zZjVlLTRiZWQtYThkMS05NjE2ZTUwNDQzYWIiLCJlbmMiOiJBMTI4Q0JDLUhTMjU2IiwiYWxnIjoiUlNBLU9BRVAtMjU2In0.CtrOcFMSeW_qMdQjgKzNaWtC6hkSTdjOSMSr2IOKNa8OpbJH8rOaCt9l4LYLHKw8CF70YLWOODgaYrLiWuHgdR-O9hlALkT6CbXxJ-Cbmf6qpJF61kXKHX4TBe6-oV8t4PaPaSDDR_oeyt4Xl2EOOlHxs9PnU1st9Fkd7wOHNa4ueM16Ze5ICEdQK3cN1hnelid0zlb1qq58KhsSroNeI8B5RnoYDB-0eiFWD0XWJLppgkLnewXpuPLmLN_c558yUQ0u0VoUyBGB6EFePPbbT-Z1_LUCSRyiP9Y2S0Vz5jzzeabWZ4vZkW8JX57Wc-onHplUpsUUpIqcdHLjp40NEQ.VtA6Sg--jeA1QavYxY9z7Q.Mr6dLyontoJCaDaRFMAYg_qUdEPzd-aIIrCHWH7wvYayNpSFUjR5QJd3XPpGGy93y22jN-DoHZHclgMeureQwKq39QQF0xIdRqhOR2Lxy69PkgRaNtpz7ikLOlsbjh1wd7mbSmyolsK_0t1X9OlvOSmUMjxUXpXLzqLXxPY0R_MUxEanHb3V_vsLArF9sN1X7Km-fdUKXTV1EzVUq1eC5aSYqg3rGkLHPHG6lPXOetPWNsVCE1bX0Voh0XnlyFLSSoYzX45l04hR8JXgcP42FXfD7GugcNi7jTKuvxu4l2Q2v7wnk"
}
```

响应示例

状态码： 200

请求已成功

```
{
  "secret" : {
    "id" : "bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e",
    "name" : "test",
    "state" : "ENABLED",
    "kms_key_id" : "b168fe00ff56492495a7d22974df2d0b",
    "description" : "description",
    "create_time" : 1581507580000,
    "update_time" : 1581507580000,
    "scheduled_delete_time" : 1581507580000
  }
}
```

SDK 代码示例

SDK代码示例如下。

Java

上传凭据备份文件。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.csms.v1.region.CsmsRegion;
import com.huaweicloud.sdk.csms.v1.*;
import com.huaweicloud.sdk.csms.v1.model.*;


public class UploadSecretBlobSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CsmsClient client = CsmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(CsmsRegion.valueOf("<YOUR REGION>"))
                .build();
        UploadSecretBlobRequest request = new UploadSecretBlobRequest();
        UploadSecretBlobRequestBody body = new UploadSecretBlobRequestBody();

body.withSecretBlob("")CloudSecretManagementBackupV1.comeyJraWQiOiI5ZjNlZmRjNS0zZjVlLTRiZWQtYThkMS05NjE2ZTUwNDQzYWIiLCJlbmMiOiJBMTI4Q0JDLUhTMjU2IiwiYWxnIjoiUlNBLU9BRVAtMjU2In0.CtrOcFMSeW_qMdQjgKzNaWtC6hkSTdjOSMSr2IOKNa8OpbJH8rOaCt9l4LYLHKw8CF70YLWOODgaYrLiWuHgdR-O9hlALkT6CbXxJ-Cbmf6qpJF61kXKHX4TBe6-oV8t4PaPaSDDR_oeyt4Xl2EOOlHxs9PnU1st9Fkd7wOHNa4ueM16Ze5ICEdQK3cN1hnelid0zlb1qq58KhsSroNeI8B5RnoYDB-0eiFWD0XWJLppgkLnewXpuPLmLN_c558yUQ0u0VoUyBGB6EFePPbbT-Z1_LUCSRyiP9Y2S0Vz5jzzeabWZ4vZkW8JX57Wc-onHplUpsUUpIqcdHLjp40NEQ.VtA6Sg--jeA1QavYxY9z7Q.Mr6dLyontoJCaDaRFMAYg_qUdEPzd-aIIrCHWH7wvYayNpSFUjR5QJd3XPpGGy93y22jN-DoHZHclgMeureQwKq39QQF0xIdRqhOR2Lxy69PkgRaNtpz7ikLOlsbjh1wd7mbSmyolsK_0t1X9OlvOSmUMjxUXpXLzqLXxPY0R_MUxEanHb3V_vsLArF9sN1X7Km-fdUKXTV1EzVUq1eC5aSYqg3rGkLHPHG6lPXOetPWNsVCE1bX0Voh0XnlyFLSSoYzX45l04hR8JXgcP42FXfD7GugcNi7jTKuvxu4l2Q2v7wnk");
        request.withBody(body);
        try {
            UploadSecretBlobResponse response = client.uploadSecretBlob(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

上传凭据备份文件。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcsms.v1.region.csms_region import CsmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcsms.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CsmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CsmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = UploadSecretBlobRequest()
        request.body = UploadSecretBlobRequestBody(

secret_blob="")CloudSecretManagementBackupV1.comeyJraWQiOiI5ZjNlZmRjNS0zZjVlLTRiZWQtYThkMS05NjE2ZTUwNDQzYWIiLCJlbmMiOiJBMTI4Q0JDLUhTMjU2IiwiYWxnIjoiUlNBLU9BRVAtMjU2In0.CtrOcFMSeW_qMdQjgKzNaWtC6hkSTdjOSMSr2IOKNa8OpbJH8rOaCt9l4LYLHKw8CF70YLWOODgaYrLiWuHgdR-O9hlALkT6CbXxJ-Cbmf6qpJF61kXKHX4TBe6-oV8t4PaPaSDDR_oeyt4Xl2EOOlHxs9PnU1st9Fkd7wOHNa4ueM16Ze5ICEdQK3cN1hnelid0zlb1qq58KhsSroNeI8B5RnoYDB-0eiFWD0XWJLppgkLnewXpuPLmLN_c558yUQ0u0VoUyBGB6EFePPbbT-Z1_LUCSRyiP9Y2S0Vz5jzzeabWZ4vZkW8JX57Wc-onHplUpsUUpIqcdHLjp40NEQ.VtA6Sg--jeA1QavYxY9z7Q.Mr6dLyontoJCaDaRFMAYg_qUdEPzd-aIIrCHWH7wvYayNpSFUjR5QJd3XPpGGy93y22jN-DoHZHclgMeureQwKq39QQF0xIdRqhOR2Lxy69PkgRaNtpz7ikLOlsbjh1wd7mbSmyolsK_0t1X9OlvOSmUMjxUXpXLzqLXxPY0R_MUxEanHb3V_vsLArF9sN1X7Km-fdUKXTV1EzVUq1eC5aSYqg3rGkLHPHG6lPXOetPWNsVCE1bX0Voh0XnlyFLSSoYzX45l04hR8JXgcP42FXfD7GugcNi7jTKuvxu4l2Q2v7wnk"
        )
        response = client.upload_secret_blob(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

上传凭据备份文件。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    csms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/region"
)
```

```
func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := csms.NewCsmsClient(
        csms.CsmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.UploadSecretBlobRequest{}
    request.Body = &model.UploadSecretBlobRequestBody{
        SecretBlob:
")CloudSecretManagementBackupV1.comeyJraWQiOiI5ZjNlZmRjNS0zZjVlLTRiZWQtYThkMS05NjE2ZTUwNDQzYWIiLCJlbmMiOiJBMTI4Q0JDLUhTMjU2IiwiYWxnIjoiUlNBLU9BRVAtMjU2In0.CtrOcFMSeW_qMdQjgKzNaWtC6hkSTdjOSMSr2IOKNa8OpbJH8rOaCt9l4LYLHKw8CF70YLWOODgaYrLiWuHgdR-O9hlALkT6CbXxJ-Cbmf6qpJF61kXKHX4TBe6-oV8t4PaPaSDDR_oeyt4Xl2EOOlHxs9PnU1st9Fkd7wOHNa4ueM16Ze5ICEdQK3cN1hnelid0zlb1qq58KhsSroNeI8B5RnoYDB-0eiFWD0XWJLppgkLnewXpuPLmLN_c558yUQ0u0VoUyBGB6EFePPbbT-Z1_LUCSRyiP9Y2S0Vz5jzzeabWZ4vZkW8JX57Wc-onHplUpsUUpIqcdHLjp40NEQ.VtA6Sg--jeA1QavYxY9z7Q.Mr6dLyontoJCaDaRFMAYg_qUdEPzd-aIIrCHWH7wvYayNpSFUjR5QJd3XPpGGy93y22jN-DoHZHclgMeureQwKq39QQF0xIdRqhOR2Lxy69PkgRaNtpz7ikLOlsbjh1wd7mbSmyolsK_0t1X9OlvOSmUMjxUXpXLzqLXxPY0R_MUxEanHb3V_vsLArF9sN1X7Km-fdUKXTV1EzVUq1eC5aSYqg3rGkLHPHG6lPXOetPWNsVCE1bX0Voh0XnlyFLSSoYzX45l04hR8JXgcP42FXfD7GugcNi7jTKuvxu4l2Q2v7wnk",
    }
    response, err := client.UploadSecretBlob(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到

状态码	描述
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.3.1.7 下载凭据备份

功能介绍

下载指定凭据的备份文件

接口约束

此接口返回的信息为表示凭据备份文件的字符串，内容加密格式，不可读。

调用方法

请参见**如何调用API**。

URI

POST /v1/{project_id}/secrets/{secret_name}/backup

表 4-1097 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。
secret_name	是	String	凭据的名称。

请求参数

表 4-1098 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码： 200

表 4-1099 响应 Body 参数

参数	参数类型	描述
secret_blob	String	将指定凭据对象进行备份后得到的凭据备份文件，备份文件包含有凭据当前所有的凭据版本信息，备份文件经过加密与编码，内容不可直接读。

状态码： 400

表 4-1100 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 401

表 4-1101 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 403

表 4-1102 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 404

表 4-1103 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 500

表 4-1104 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 502

表 4-1105 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 504

表 4-1106 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

请求示例

无

响应示例

状态码： 200

请求已成功

```
{
  "secret_blob" :
```

```
")CloudSecretManagementBackupV1.comeyJraWQiOiI5ZjNlZmRjNS0zZjVlLTRiZWQtYThkMS05NjE2ZTUwNDQzYWIiLCJlbmMiOiJBMTI4Q0JDLUhTMjU2IiwiYWxnIjoiUlNBLU9BRVAtMjU2In0.CtrOcFMSeW_qMdQjgKzNaWtC6hkSTdjOSMSr2IOKNa8OpbJH8rOaCt9l4LYLHKw8CF70YLWOODgaYrLiWuHgdR-O9hlALkT6CbXxJ-Cbmf6qpJF61kXKHX4TBe6-oV8t4PaPaSDDR_oeyt4Xl2EOOlHxs9PnU1st9Fkd7wOHNa4ueM16Ze5ICEdQK3cN1hnelid0zlb1qq58KhsSroNeI8B5RnoYDB-0eiFWD0XWJLppgkLnewXpuPLmLN_c558yUQ0u0VoUyBGB6EFePPbbT-Z1_LUCSRyiP9Y2S0Vz5jzzeabWZ4vZkW8JX57Wc-onHplUpsUUpIqcdHLjp40NEQ.VtA6Sg--jeA1QavYxY9z7Q.Mr6dLyontoJCaDaRFMAYg_qUdEPzd-aIIrCHWH7wvYayNpSFUjR5QJd3XPpGGy93y22jN-DoHZHclgMeureQwKq39QQF0xIdRqhOR2Lxy69PkgRaNtpz7ikLOlsbjh1wd7mbSmyolsK_0t1X9OlvOSmUMjxUXpXLzqLXxPY0R_MUxEanHb3V_vsLArF9sN1X7Km-fdUKXTV1EzVUq1eC5aSYqg3rGkLHPHG6lPXOetPWNsVCE1bX0Voh0XnlyFLSSoYzX45l04hR8JXgcP42FXfD7GugcNi7jTKuvxu4l2Q2v7wnk"
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.csms.v1.region.CsmsRegion;
import com.huaweicloud.sdk.csms.v1.*;
import com.huaweicloud.sdk.csms.v1.model.*;


public class DownloadSecretBlobSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CsmsClient client = CsmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(CsmsRegion.valueOf("<YOUR REGION>"))
                .build();
        DownloadSecretBlobRequest request = new DownloadSecretBlobRequest();
        request.withSecretName("{secret_name}");
        try {
            DownloadSecretBlobResponse response = client.downloadSecretBlob(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
```

```
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcsms.v1.region.csms_region import CsmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcsms.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CsmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CsmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = DownloadSecretBlobRequest()
        request.secret_name = "{secret_name}"
        response = client.download_secret_blob(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    csms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()
```

```
    client := csms.NewCsmsClient(
        csms.CsmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.DownloadSecretBlobRequest{}
    request.SecretName = "{secret_name}"
    response, err := client.DownloadSecretBlob(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.3.1.8 创建凭据的定时删除任务

功能介绍

指定延迟删除时间，创建删除凭据的定时任务，可设置7~30天的的延迟删除时间。

接口约束

处于“定时删除”状态的凭据对象无法更新元数据信息，也无法查看凭据值。

调用方法

请参见**如何调用API**。

URI

POST /v1/{project_id}/secrets/{secret_name}/scheduled-deleted-tasks/create

表 4-1107 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。
secret_name	是	String	凭据名称。

请求参数

表 4-1108 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-1109 请求 Body 参数

参数	是否必选	参数类型	描述
recovery_window_in_days	是	Integer	创建定时删除凭据的任务，且指定可恢复的天数。 约束：7~30。 默认值：30。

响应参数

状态码： 200

表 4-1110 响应 Body 参数

参数	参数类型	描述
secret	**Secret** object	凭据对象。

表 4-1111 Secret

参数	参数类型	描述
id	String	凭据的资源标识符。
name	String	凭据名称。
state	String	凭据状态，取值如下： ENABLED：表示启用状态 DISABLED：表示禁用状态 PENDING_DELETE：表示待删除状态 FROZEN：表示冻结状态
kms_key_id	String	用于加密凭据值的KMS主密钥的ID值。
description	String	凭据的描述信息。
create_time	Long	凭据创建时间，时间戳，即从1970年1月1日至该时间的总秒数。
update_time	Long	凭据上次更新时间，时间戳，即从1970年1月1日至该时间的总秒数。
scheduled_delete_time	Long	凭据计划删除时间，时间戳，即从1970年1月1日至该时间的总秒数。 凭据不在删除计划中时，本项值为null。
secret_type	String	凭据类型 ● COMMON：通用凭据(默认)。用于应用系统中的各种敏感信息储存。 ● RDS：RDS凭据 。专门针对RDS的凭据，用于存储RDS的账号信息。（ 已不支持，使用RDS-FG替代 ） ● RDS-FG：RDS凭据 。专门针对RDS的凭据，用于存储RDS的账号信息。 ● GaussDB-FG：TaurusDB凭据。专门针对TaurusDB的凭据，用于存储TaurusDB的账号信息。
auto_rotation	Boolean	自动轮转 取值：true 开启, false 关闭(默认)
rotation_period	String	轮转周期 约束：6小时-8,760小时 （ 365天 ） 类型：Integer[unit] ， Integer表示时间长度 。unit表示时间单位，d（天）、h（小时）、m（分钟）、s（秒）。例如 1d 表示一天，24h也表示一天 说明：当开启自动轮转时，必须填写该值

参数	参数类型	描述
rotation_config	String	轮转配置 约束：范围不超过1024个字符。 当secret_type为RDS-FG、GaussDB-FG时，配置为{"InstanceId":"","SecretSubType":""} 说明：当secret_type为RDS-FG、GaussDB-FG时，必须填写该值 InstanceId为实例ID,SecretSubType为轮转子类型，取值为：SingleUser，MultiUser。 SingleUser：指定轮转类型为单用户模式轮转，每次轮转将指定账号重置为新的口令。 MultiUser：指定轮转类型为双用户模式轮转，SYSCURRENT和SYSPREVIOUS分别引用其中一个账号。凭据轮转时，SYSPREVIOUS引用的账号口令会被重置为新的随机口令，随后凭据交换SYSCURRENT和SYSPREVIOUS对账号的引用。
rotation_time	Long	轮转时间戳
next_rotation_time	Long	下一次轮转时间戳
event_subscriptions	Array of strings	凭据订阅的事件列表，当前最大可订阅一个事件。当事件包含的基础事件触发时，通知消息将发送到事件对应的通知主题。
enterprise_project_id	String	企业项目ID
rotation_func_urn	String	FunctionGraph函数的urn。

状态码： 400

表 4-1112 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 401

表 4-1113 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 403

表 4-1114 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 404

表 4-1115 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 500

表 4-1116 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 502

表 4-1117 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 504

表 4-1118 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

请求示例

创建凭据定时删除任务，15天后进行删除。

```
{
  "recovery_window_in_days" : 15
}
```

响应示例

状态码： 200

请求已成功

```
{
  "secret" : {
    "id" : "bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e",
    "name" : "test",
    "state" : "ENABLED",
    "kms_key_id" : "b168fe00ff56492495a7d22974df2d0b",
    "description" : "description",
    "create_time" : 1581507580000,
    "update_time" : 1581507580000,
    "scheduled_delete_time" : 1581507580000
  }
}
```

SDK 代码示例

SDK代码示例如下。

Java

创建凭据定时删除任务，15天后进行删除。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.csms.v1.region.CsmsRegion;
import com.huaweicloud.sdk.csms.v1.*;
import com.huaweicloud.sdk.csms.v1.model.*;


public class DeleteSecretForScheduleSolution {

    public static void main(String[] args) {
```

```
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CsmsClient client = CsmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(CsmsRegion.valueOf("<YOUR REGION>"))
                .build();
        DeleteSecretForScheduleRequest request = new DeleteSecretForScheduleRequest();
        request.withSecretName("{secret_name}");
        DeleteSecretForScheduleRequestBody body = new DeleteSecretForScheduleRequestBody();
        body.withRecoveryWindowInDays(15);
        request.withBody(body);
        try {
            DeleteSecretForScheduleResponse response = client.deleteSecretForSchedule(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

创建凭据定时删除任务，15天后进行删除。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcsms.v1.region.csms_region import CsmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcsms.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CsmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CsmsRegion.value_of("<YOUR REGION>")) \
        .build()
```

```
    try:
        request = DeleteSecretForScheduleRequest()
        request.secret_name = "{secret_name}"
        request.body = DeleteSecretForScheduleRequestBody(
            recovery_window_in_days=15
        )
        response = client.delete_secret_for_schedule(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

创建凭据定时删除任务，15天后进行删除。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    csms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := csms.NewCsmsClient(
        csms.CsmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.DeleteSecretForScheduleRequest{}
    request.SecretName = "{secret_name}"
    request.Body = &model.DeleteSecretForScheduleRequestBody{
        RecoveryWindowInDays: int32(15),
    }
    response, err := client.DeleteSecretForSchedule(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.3.1.9 取消凭据的定时删除任务

功能介绍

取消凭据的定时删除任务，凭据对象恢复可使用状态。

接口约束

处于“定时删除”状态的凭据，才能执行本接口。

调用方法

请参见**如何调用API**。

URI

POST /v1/{project_id}/secrets/{secret_name}/scheduled-deleted-tasks/cancel

表 4-1119 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。
secret_name	是	String	凭据名称。

请求参数

表 4-1120 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码： 200

表 4-1121 响应 Body 参数

参数	参数类型	描述
secret	**Secret** object	凭据对象。

表 4-1122 Secret

参数	参数类型	描述
id	String	凭据的资源标识符。
name	String	凭据名称。
state	String	凭据状态，取值如下： ENABLED：表示启用状态 DISABLED：表示禁用状态 PENDING_DELETE：表示待删除状态 FROZEN：表示冻结状态
kms_key_id	String	用于加密凭据值的KMS主密钥的ID值。
description	String	凭据的描述信息。
create_time	Long	凭据创建时间，时间戳，即从1970年1月1日至该时间的总秒数。
update_time	Long	凭据上次更新时间，时间戳，即从1970年1月1日至该时间的总秒数。
scheduled_delete_time	Long	凭据计划删除时间，时间戳，即从1970年1月1日至该时间的总秒数。 凭据不在删除计划中时，本项值为null。

参数	参数类型	描述
secret_type	String	凭据类型 ● COMMON：通用凭据(默认)。用于应用系统中的各种敏感信息储存。 ● RDS：RDS凭据 。专门针对RDS的凭据，用于存储RDS的账号信息。（ 已不支持，使用RDS-FG替代 ） ● RDS-FG：RDS凭据 。专门针对RDS的凭据，用于存储RDS的账号信息。 ● GaussDB-FG：TaurusDB凭据。专门针对TaurusDB的凭据，用于存储TaurusDB的账号信息。
auto_rotation	Boolean	自动轮转 取值：true 开启, false 关闭(默认)
rotation_period	String	轮转周期 约束：6小时-8,760小时 （ 365天 ） 类型：Integer[unit] ，Integer表示时间长度 。unit表示时间单位，d（ 天 ）、h（ 小时 ）、m（ 分钟 ）、s（ 秒 ）。例如 1d 表示一天，24h也表示一天 说明：当开启自动轮转时，必须填写该值
rotation_config	String	轮转配置 约束：范围不超过1024个字符。 当secret_type为RDS-FG、GaussDB-FG时，配置为{"InstanceId":"","SecretSubType":""} 说明：当secret_type为RDS-FG、GaussDB-FG时，必须填写该值 InstanceId为实例ID,SecretSubType为轮转子类型，取值为：SingleUser，MultiUser。 SingleUser：指定轮转类型为单用户模式轮转，每次轮转将指定账号重置为新的口令。 MultiUser：指定轮转类型为双用户模式轮转，SYSCURRENT和SYSPREVIOUS分别引用其中一个账号。凭据轮转时，SYSPREVIOUS引用的账号口令会被重置为新的随机口令，随后凭据交换SYSCURRENT和SYSPREVIOUS对账号的引用。
rotation_time	Long	轮转时间戳
next_rotation_time	Long	下一次轮转时间戳
event_subscriptions	Array of strings	凭据订阅的事件列表，当前最大可订阅一个事件。当事件包含的基础事件触发时，通知消息将发送到事件对应的通知主题。

参数	参数类型	描述
enterprise_project_id	String	企业项目ID
rotation_func_urn	String	FunctionGraph函数的urn。

状态码： 400

表 4-1123 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 401

表 4-1124 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 403

表 4-1125 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 404

表 4-1126 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码

参数	参数类型	描述
error_msg	String	错误描述

状态码： 500

表 4-1127 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 502

表 4-1128 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 504

表 4-1129 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

请求示例

无

响应示例

状态码： 200

请求已成功

```
{
  "secret" : {
    "id" : "bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e",
    "name" : "test",
```

```
  "state" : "ENABLED",
  "kms_key_id" : "b168fe00ff56492495a7d22974df2d0b",
  "description" : "description",
  "create_time" : 1581507580000,
  "update_time" : 1581507580000,
  "scheduled_delete_time" : 1581507580000
 }
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.csms.v1.region.CsmsRegion;
import com.huaweicloud.sdk.csms.v1.*;
import com.huaweicloud.sdk.csms.v1.model.*;


public class RestoreSecretSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CsmsClient client = CsmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(CsmsRegion.valueOf("<YOUR REGION>"))
                .build();
        RestoreSecretRequest request = new RestoreSecretRequest();
        request.withSecretName("{secret_name}");
        try {
            RestoreSecretResponse response = client.restoreSecret(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcsms.v1.region.csms_region import CsmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcsms.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CsmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CsmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = RestoreSecretRequest()
        request.secret_name = "{secret_name}"
        response = client.restore_secret(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    csms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := csms.NewCsmsClient(
        csms.CsmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
```

```
        WithCredential(auth).
        Build())

    request := &model.RestoreSecretRequest{}
    request.SecretName = "{secret_name}"
    response, err := client.RestoreSecret(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.3.1.10 轮转凭据

功能介绍

立即执行轮转凭据。在指定的凭据中，创建一个新的凭据版本，用于加密存储后台随机产生的凭据值。同时将新创建的凭据版本标记为SYSCURRENT状态。

接口约束

RotateSecret接口不支持轮转通用凭据。

用户账号拥有以下权限：

rds修改数据库密码

查询密钥信息

查询密钥列表

创建数据密钥

解密数据密钥。

调用方法

请参见**如何调用API**。

URI

POST /v1/{project_id}/secrets/{secret_name}/rotate

表 4-1130 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。
secret_name	是	String	凭据名称。

请求参数

表 4-1131 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码： 200

表 4-1132 响应 Body 参数

参数	参数类型	描述
version_id	String	凭据的版本号标识符。
secret_name	String	凭据的名称。
rotation_task_id	String	凭据轮转任务ID。

状态码： 400

表 4-1133 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1134 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-1135 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1136 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-1137 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1138 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-1139 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1140 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-1141 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1142 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-1143 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1144 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-1145 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1146 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

无

响应示例

状态码： 200

请求已成功

```
{
  "rotation_task_id" : "a71a4b47-6cac-4f11-92c1-21a165bb6401"
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.csms.v1.region.CsmsRegion;
import com.huaweicloud.sdk.csms.v1.*;
import com.huaweicloud.sdk.csms.v1.model.*;


public class RotateSecretSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CsmsClient client = CsmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(CsmsRegion.valueOf("<YOUR REGION>"))
                .build();
        RotateSecretRequest request = new RotateSecretRequest();
        request.withSecretName("{secret_name}");
        try {
            RotateSecretResponse response = client.rotateSecret(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcsms.v1.region.csms_region import CsmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcsms.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this
```

```
example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CsmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CsmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = RotateSecretRequest()
        request.secret_name = "{secret_name}"
        response = client.rotate_secret(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    csms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := csms.NewCsmsClient(
        csms.CsmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.RotateSecretRequest{}
    request.SecretName = "{secret_name}"
    response, err := client.RotateSecret(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.3.2 凭据版本管理

4.3.2.1 创建凭据版本

功能介绍

在指定的凭据中，创建一个新的凭据版本，用于加密保管新的凭据值。默认情况下，新创建的凭据版本被标记为SYSCURRENT状态，而SYSCURRENT标记的前一个凭据版本被标记为SYSPREVIOUS状态。您可以通过指定VersionStage参数来覆盖默认行为。

接口约束

- 凭据管理服务console控制台仅使用secret_string字段。如果您想要添加二进制类型凭据到secret_binary字段中保存，必须使用SDK或API的方式。
- 凭据管理服务的每个凭据中最多可支持20个版本。
- 只能对处于ENABLED状态的凭据添加新的版本。
- 每次存入新的凭据值时，凭据版本号按照为v1，v2，v3...的模式自动增加。

调用方法

请参见**如何调用API**。

URI

POST /v1/{project_id}/secrets/{secret_name}/versions

表 4-1147 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。
secret_name	是	String	凭据名称。

请求参数

表 4-1148 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-1149 请求 Body 参数

参数	是否必选	参数类型	描述
secret_binary	否	String	新创建凭据的凭据值，将其加密后，存入初始版本中。 类型：base64编码的二进制数据对象。 约束：secret_binary和secret_string必须且只能设置一个，最大32K。
secret_string	否	String	新创建凭据的凭据值，将其加密后，存入初始版本中。 约束：secret_binary和secret_string必须且只能设置一个，最大32K。
version_stages	否	Array of strings	凭据版本在存入时需要被同时标记的版本状态。如果您不指定此参数，凭据管家默认为新版本标记SYSCURRENT 约束：数组大小：最小1，最大12。stage长度：最小1字节，最大64字节。

参数	是否必选	参数类型	描述
expire_time	否	Long	凭据版本过期时间，时间戳，即从1970年1月1日至该时间的总秒数。默认为空，凭据订阅“版本过期”事件类型时，有效期判断所依据的值。

响应参数

状态码： 200

表 4-1150 响应 Body 参数

参数	参数类型	描述
version_metadata	**VersionMetadata** object	凭据版本被标记的状态。

表 4-1151 VersionMetadata

参数	参数类型	描述
id	String	凭据的版本号标识符，凭据对象下唯一。
create_time	Long	凭据版本创建时间，时间戳，即从1970年1月1日至该时间的总秒数。
expire_time	Long	凭据版本过期时间，时间戳，即从1970年1月1日至该时间的总秒数。默认为空，凭据订阅“版本过期”事件类型时，有效期判断所依据的值。
kms_key_id	String	加密版本凭据值的KMS主密钥ID。
secret_name	String	凭据名称。
version_stages	Array of strings	凭据版本被标记的状态列表。每个版本标签对于凭据对象下版本是唯一存在的，如果创建版本时，指定的是同一凭据对象下的一个已经标记在其他版本上的状态，该标签将自动从其他版本上删除，并附加到此版本上。 如果未指定version_stage的值，则凭据管理服务会自动移动临时标签SYSCURRENT到此新版本。

状态码： 400

表 4-1152 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 401

表 4-1153 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 403

表 4-1154 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 404

表 4-1155 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 500

表 4-1156 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 502

表 4-1157 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 504

表 4-1158 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

请求示例

创建凭据版本，凭据值为secret_string。

```
{
  "secret_string" : "secret_string"
}
```

响应示例

状态码： 200

请求已成功

```
{
  "version_metadata" : {
    "id" : "v1",
    "kms_key_id" : "b168fe00ff56492495a7d22974df2d0b",
    "create_time" : 1581507580000,
    "secret_name" : "secret-name-demo",
    "version_stages" : [ "SYSCURRENT" ]
  }
}
```

SDK 代码示例

SDK代码示例如下。

Java

创建凭据版本，凭据值为secret_string。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
```

```
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.csms.v1.region.CsmsRegion;
import com.huaweicloud.sdk.csms.v1.*;
import com.huaweicloud.sdk.csms.v1.model.*;


public class CreateSecretVersionSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CsmsClient client = CsmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(CsmsRegion.valueOf("<YOUR REGION>"))
                .build();
        CreateSecretVersionRequest request = new CreateSecretVersionRequest();
        request.withSecretName("{secret_name}");
        CreateSecretVersionRequestBody body = new CreateSecretVersionRequestBody();
        body.withSecretString("secret_string");
        request.withBody(body);
        try {
            CreateSecretVersionResponse response = client.createSecretVersion(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

创建凭据版本，凭据值为secret_string。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcsms.v1.region.csms_region import CsmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcsms.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this
```

```
example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CsmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CsmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = CreateSecretVersionRequest()
        request.secret_name = "{secret_name}"
        request.body = CreateSecretVersionRequestBody(
            secret_string="secret_string"
        )
        response = client.create_secret_version(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

创建凭据版本，凭据值为secret_string。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    csms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := csms.NewCsmsClient(
        csms.CsmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.CreateSecretVersionRequest{}
    request.SecretName = "{secret_name}"
    secretStringCreateSecretVersionRequestBody:= "secret_string"
    request.Body = &model.CreateSecretVersionRequestBody{
        SecretString: &secretStringCreateSecretVersionRequestBody,
    }
    response, err := client.CreateSecretVersion(request)
```

```
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.3.2.2 查询凭据的版本列表

功能介绍

查询指定凭据下的版本列表信息。

接口约束

此接口返回的信息为凭据版本的元数据信息，不包含凭据值。

调用方法

请参见**如何调用API**。

URI

GET /v1/{project_id}/secrets/{secret_name}/versions

表 4-1159 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。
secret_name	是	String	凭据名称。

表 4-1160 Query 参数

参数	是否必选	参数类型	描述
marker	否	String	分页参数，取值为上一页数据的最后一条记录的版本号。
limit	否	Integer	每页显示的条目数量。默认值50。

请求参数

表 4-1161 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码： 200

表 4-1162 响应 Body 参数

参数	参数类型	描述
version_metadatas	Array of **VersionMetadata** objects	version_metadata对象。
page_info	**PageInfo** object	分页信息。

表 4-1163 VersionMetadata

参数	参数类型	描述
id	String	凭据的版本号标识符，凭据对象下唯一。
create_time	Long	凭据版本创建时间，时间戳，即从1970年1月1日至该时间的总秒数。
expire_time	Long	凭据版本过期时间，时间戳，即从1970年1月1日至该时间的总秒数。默认为空，凭据订阅“版本过期”事件类型时，有效期判断所依据的值。
kms_key_id	String	加密版本凭据值的KMS主密钥ID。
secret_name	String	凭据名称。
version_stages	Array of strings	凭据版本被标记的状态列表。每个版本标签对于凭据对象下版本是唯一存在的，如果创建版本时，指定的是同一凭据对象下的一个已经标记在其他版本上的状态，该标签将自动从其他版本上删除，并附加到此版本上。 如果未指定version_stage的值，则凭据管理服务会自动移动临时标签SYSCURRENT到此新版本。

表 4-1164 PageInfo

参数	参数类型	描述
next_marker	String	下一页查询地址（本页的末尾凭据名称，下一页起始凭据名称）。
previous_marker	String	本页的起始凭据名称，上一页末尾凭据名称。
current_count	Integer	本页返回条目数量。

状态码： 400

表 4-1165 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 401

表 4-1166 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 403

表 4-1167 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 404

表 4-1168 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 500

表 4-1169 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 502

表 4-1170 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 504

表 4-1171 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

请求示例

无

响应示例

状态码： 200

请求已成功

```
{
  "version_metadatas" : [ {
    "id" : "v1",
    "kms_key_id" : "b168fe00ff56492495a7d22974df2d0b",
    "create_time" : 1581507580000,
    "secret_name" : "secret-name-demo",
    "version_stages" : [ "SYSCURRENT" ]
  } ],
  "page_info" : {
    "next_marker" : "v10",
    "previous_marker" : "v1",
    "current_count" : 10
  }
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.csms.v1.region.CsmsRegion;
import com.huaweicloud.sdk.csms.v1.*;
import com.huaweicloud.sdk.csms.v1.model.*;


public class ListSecretVersionsSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
```

```
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CsmsClient client = CsmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(CsmsRegion.valueOf("<YOUR REGION>"))
                .build();
        ListSecretVersionsRequest request = new ListSecretVersionsRequest();
        request.withSecretName("{secret_name}");
        try {
            ListSecretVersionsResponse response = client.listSecretVersions(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcsms.v1.region.csms_region import CsmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcsms.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CsmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CsmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ListSecretVersionsRequest()
        request.secret_name = "{secret_name}"
        response = client.list_secret_versions(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    csms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := csms.NewCsmsClient(
        csms.CsmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ListSecretVersionsRequest{}
    request.SecretName = "{secret_name}"
    response, err := client.ListSecretVersions(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误

状态码	描述
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.3.2.3 更新凭据版本

功能介绍

当前支持更新指定凭据版本的有效期，只能更新ENABLED状态的凭据。在关联订阅的事件包含“版本过期”基础事件类型时，每次更新版本有效期后仅会触发一次事件通知。

调用方法

请参见**如何调用API**。

URI

PUT /v1/{project_id}/secrets/{secret_name}/versions/{version_id}

表 4-1172 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。
secret_name	是	String	凭据名称。
version_id	是	String	凭据的版本标识符。

请求参数

表 4-1173 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-1174 请求 Body 参数

参数	是否必选	参数类型	描述
expire_time	是	Long	凭据版本过期时间，时间戳，即从1970年1月1日至该时间的总秒数。默认为空，凭据订阅“版本过期”事件类型时，有效期判断所依据的值。

响应参数

状态码： 200

表 4-1175 响应 Body 参数

参数	参数类型	描述
version_metadata	**VersionMetadata** object	凭据版本被标记的状态。

表 4-1176 VersionMetadata

参数	参数类型	描述
id	String	凭据的版本号标识符，凭据对象下唯一。
create_time	Long	凭据版本创建时间，时间戳，即从1970年1月1日至该时间的总秒数。
expire_time	Long	凭据版本过期时间，时间戳，即从1970年1月1日至该时间的总秒数。默认为空，凭据订阅“版本过期”事件类型时，有效期判断所依据的值。
kms_key_id	String	加密版本凭据值的KMS主密钥ID。
secret_name	String	凭据名称。
version_stages	Array of strings	凭据版本被标记的状态列表。每个版本标签对于凭据对象下版本是唯一存在的，如果创建版本时，指定的是同一凭据对象下的一个已经标记在其他版本上的状态，该标签将自动从其他版本上删除，并附加到此版本上。 如果未指定version_stage的值，则凭据管理服务会自动移动临时标签SYSCURRENT到此新版本。

状态码： 400

表 4-1177 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 401

表 4-1178 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 403

表 4-1179 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 404

表 4-1180 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 500

表 4-1181 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 502

表 4-1182 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 504

表 4-1183 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

请求示例

更新凭据版本，过期时间为1696838472000

```
{
  "expire_time" : 1696838472000
}
```

响应示例

状态码： 200

请求已成功

```
{
  "version_metadata" : {
    "id" : "v3",
    "kms_key_id" : "b168fe00ff56492495a7d22974df2d0b",
    "create_time" : 1581507580000,
    "secret_name" : "secret-name-demo",
    "version_stages" : [ "SYSCURRENT" ]
  }
}
```

SDK 代码示例

SDK代码示例如下。

Java

更新凭据版本，过期时间为1696838472000

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
```

```
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.csms.v1.region.CsmsRegion;
import com.huaweicloud.sdk.csms.v1.*;
import com.huaweicloud.sdk.csms.v1.model.*;


public class UpdateVersionSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CsmsClient client = CsmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(CsmsRegion.valueOf("<YOUR REGION>"))
                .build();
        UpdateVersionRequest request = new UpdateVersionRequest();
        request.withSecretName("{secret_name}");
        request.withVersionId("{version_id}");
        UpdateVersionRequestBody body = new UpdateVersionRequestBody();
        body.withExpireTime(1696838472000L);
        request.withBody(body);
        try {
            UpdateVersionResponse response = client.updateVersion(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

更新凭据版本，过期时间为1696838472000

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcsms.v1.region.csms_region import CsmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcsms.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
```

```
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CsmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CsmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = UpdateVersionRequest()
        request.secret_name = "{secret_name}"
        request.version_id = "{version_id}"
        request.body = UpdateVersionRequestBody(
            expire_time=1696838472000
        )
        response = client.update_version(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

更新凭据版本，过期时间为1696838472000

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    csms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := csms.NewCsmsClient(
        csms.CsmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.UpdateVersionRequest{}
    request.SecretName = "{secret_name}"
    request.VersionId = "{version_id}"
    request.Body = &model.UpdateVersionRequestBody{
        ExpireTime: int64(1696838472000),
```

```
    }
    response, err := client.UpdateVersion(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.3.2.4 查询凭据的版本与凭据值

功能介绍

查询指定凭据版本的信息和版本中的明文凭据值，只能查询ENABLED状态的凭据。

通过/v1/{project_id}/secrets/{secret_name}/versions/latest（即将当前接口URL中的{version_id}赋值为latest）可访问凭据最新版本的凭据值。

调用方法

请参见**如何调用API**。

URI

GET /v1/{project_id}/secrets/{secret_name}/versions/{version_id}

表 4-1184 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。
secret_name	是	String	凭据名称。
version_id	是	String	凭据的版本标识符。

请求参数

表 4-1185 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码： 200

表 4-1186 响应 Body 参数

参数	参数类型	描述
version	**Version** object	凭据版本。

表 4-1187 Version

参数	参数类型	描述
version_metadata	**VersionMetadata** object	凭据版本被标记的状态。
secret_binary	String	二进制类型凭据在base64编码后的明文，凭据管理服务将其加密后，存入凭据的初始版本中。 类型：base64编码的二进制数据对象。
secret_string	String	文本类型凭据的明文，凭据管理服务将其加密后，存入凭据的初始版本中。

表 4-1188 VersionMetadata

参数	参数类型	描述
id	String	凭据的版本号标识符，凭据对象下唯一。
create_time	Long	凭据版本创建时间，时间戳，即从1970年1月1日至该时间的总秒数。
expire_time	Long	凭据版本过期时间，时间戳，即从1970年1月1日至该时间的总秒数。默认为空，凭据订阅“版本过期”事件类型时，有效期判断所依据的值。
kms_key_id	String	加密版本凭据值的KMS主密钥ID。
secret_name	String	凭据名称。
version_stages	Array of strings	凭据版本被标记的状态列表。每个版本标签对于凭据对象下版本是唯一存在的，如果创建版本时，指定的是同一凭据对象下的一个已经标记在其他版本上的状态，该标签将自动从其他版本上删除，并附加到此版本上。 如果未指定version_stage的值，则凭据管理服务会自动移动临时标签SYSCURRENT到此新版本。

状态码： 400

表 4-1189 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 401

表 4-1190 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 403

表 4-1191 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 404

表 4-1192 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 500

表 4-1193 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 502

表 4-1194 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 504

表 4-1195 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

请求示例

无

响应示例

状态码： 200

请求已成功

```
{
  "version" : {
    "version_metadata" : {
      "id" : "v6",
      "kms_key_id" : "b168fe00ff56492495a7d22974df2d0b",
      "create_time" : 1581507580000,
      "secret_name" : "secret-name-demo",
      "version_stages" : [ "SYSCURRENT" ]
    },
    "secret_string" : "\\\"demo_key\\\":\\\"demo_value\\\""
  }
}
```

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.3.3 凭据版本状态管理

4.3.3.1 更新凭据的版本状态

功能介绍

更新凭据的版本状态。

接口约束

- 凭据的版本状态在同一凭据对象下仅能唯一标识一个版本，将版本状态设置到新版本上的同时，将自动从老版本上移除其对应的版本状态。没有任何版本状态标识的版本将被视为已弃用，可由凭据管理服务自动删除。
- 凭据管理服务的每个凭据中最多可支持12个凭据版本状态，每个凭据版本状态同时仅能标识一个凭据版本。SYSCURRENT，SYSPREVIOUS为服务内建的凭据状态。

调用方法

请参见**如何调用API**。

URI

PUT /v1/{project_id}/secrets/{secret_name}/stages/{stage_name}

表 4-1196 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。
secret_name	是	String	凭据名称。
stage_name	是	String	凭据版本状态的名称。满足'^[a-zA-Z0-9_-]{1,64}$'

请求参数

表 4-1197 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-1198 请求 Body 参数

参数	是否必选	参数类型	描述
version_id	是	String	凭据的版本号标识符。

响应参数

状态码： 200

表 4-1199 响应 Body 参数

参数	参数类型	描述
stage	**Stage** object	凭据状态。

表 4-1200 Stage

参数	参数类型	描述
name	String	凭据的版本状态名称。 约束：最小长度1，最大长度64。
update_time	Long	凭据的版本状态更新的时间戳，时间戳，即从1970年1月1日至该时间的总秒数。
secret_name	String	凭据名称。
version_id	String	凭据的版本号标识符。

状态码： 400

表 4-1201 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 401

表 4-1202 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 403

表 4-1203 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码

参数	参数类型	描述
error_msg	String	错误描述

状态码： 404

表 4-1204 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 500

表 4-1205 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 502

表 4-1206 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 504

表 4-1207 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

请求示例

更新凭据的版本状态，版本为version_id。

```
{
  "version_id" : "version_id"
}
```

响应示例

状态码： 200

请求已成功

```
{
  "stage" : {
    "name" : "name",
    "version_id" : "v1",
    "update_time" : 1581507580000,
    "secret_name" : "secret-name-demo"
  }
}
```

SDK 代码示例

SDK代码示例如下。

Java

更新凭据的版本状态，版本为version_id。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.csms.v1.region.CsmsRegion;
import com.huaweicloud.sdk.csms.v1.*;
import com.huaweicloud.sdk.csms.v1.model.*;


public class UpdateSecretStageSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CsmsClient client = CsmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(CsmsRegion.valueOf("<YOUR REGION>"))
                .build();
        UpdateSecretStageRequest request = new UpdateSecretStageRequest();
        request.withSecretName("{secret_name}");
```

```
        request.withStageName("{stage_name}");
        UpdateSecretStageRequestBody body = new UpdateSecretStageRequestBody();
        body.withVersionId("version_id");
        request.withBody(body);
        try {
            UpdateSecretStageResponse response = client.updateSecretStage(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

更新凭据的版本状态，版本为version_id。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcsms.v1.region.csms_region import CsmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcsms.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CsmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CsmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = UpdateSecretStageRequest()
        request.secret_name = "{secret_name}"
        request.stage_name = "{stage_name}"
        request.body = UpdateSecretStageRequestBody(
            version_id="version_id"
        )
        response = client.update_secret_stage(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

更新凭据的版本状态，版本为version_id。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    csms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := csms.NewCsmsClient(
        csms.CsmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.UpdateSecretStageRequest{}
    request.SecretName = "{secret_name}"
    request.StageName = "{stage_name}"
    request.Body = &model.UpdateSecretStageRequestBody{
        VersionId: "version_id",
    }
    response, err := client.UpdateSecretStage(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到

状态码	描述
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.3.3.2 查询凭据的版本状态

功能介绍

查询指定凭据版本状态标记的版本信息。

调用方法

请参见**如何调用API**。

URI

GET /v1/{project_id}/secrets/{secret_name}/stages/{stage_name}

表 4-1208 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。
secret_name	是	String	凭据名称。
stage_name	是	String	凭据版本状态的名称。

请求参数

表 4-1209 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码： 200

表 4-1210 响应 Body 参数

参数	参数类型	描述
stage	**Stage** object	凭据状态。

表 4-1211 Stage

参数	参数类型	描述
name	String	凭据的版本状态名称。 约束：最小长度1，最大长度64。
update_time	Long	凭据的版本状态更新的时间戳，时间戳，即从1970年1月1日至该时间的总秒数。
secret_name	String	凭据名称。
version_id	String	凭据的版本号标识符。

状态码： 400

表 4-1212 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 401

表 4-1213 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 403

表 4-1214 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 404

表 4-1215 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 500

表 4-1216 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 502

表 4-1217 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 504

表 4-1218 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

请求示例

无

响应示例

状态码： 200

请求已成功

```
{
  "stage" : {
    "name" : "name",
    "version_id" : "v20",
    "update_time" : 1581507580000,
    "secret_name" : "secret-name-demo"
  }
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.csms.v1.region.CsmsRegion;
import com.huaweicloud.sdk.csms.v1.*;
import com.huaweicloud.sdk.csms.v1.model.*;


public class ShowSecretStageSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CsmsClient client = CsmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(CsmsRegion.valueOf("<YOUR REGION>"))
                .build();
        ShowSecretStageRequest request = new ShowSecretStageRequest();
        request.withSecretName("{secret_name}");
        request.withStageName("{stage_name}");
        try {
            ShowSecretStageResponse response = client.showSecretStage(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
```

```
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcsms.v1.region.csms_region import CsmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcsms.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CsmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CsmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ShowSecretStageRequest()
        request.secret_name = "{secret_name}"
        request.stage_name = "{stage_name}"
        response = client.show_secret_stage(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    csms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
```

```
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := csms.NewCsmsClient(
        csms.CsmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ShowSecretStageRequest{}
    request.SecretName = "{secret_name}"
    request.StageName = "{stage_name}"
    response, err := client.ShowSecretStage(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.3.3.3 删除凭据的版本状态

功能介绍

删除指定的凭据版本状态。

接口约束

不可删除内建的版本状态：SYSCURRENT，SYSPREVIOUS。

调用方法

请参见**如何调用API**。

URI

DELETE /v1/{project_id}/secrets/{secret_name}/stages/{stage_name}

表 4-1219 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。
secret_name	是	String	凭据的资源标识符。
stage_name	是	String	凭据版本状态的名称。

请求参数

表 4-1220 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码： 400

表 4-1221 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码

参数	参数类型	描述
error_msg	String	错误描述

状态码： 401

表 4-1222 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 403

表 4-1223 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 404

表 4-1224 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 500

表 4-1225 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 502

表 4-1226 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 504

表 4-1227 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

请求示例

无

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.csms.v1.region.CsmsRegion;
import com.huaweicloud.sdk.csms.v1.*;
import com.huaweicloud.sdk.csms.v1.model.*;


public class DeleteSecretStageSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
```

```
            .withProjectId(projectId)
            .withAk(ak)
            .withSk(sk);

        CsmsClient client = CsmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(CsmsRegion.valueOf("<YOUR REGION>"))
                .build();
        DeleteSecretStageRequest request = new DeleteSecretStageRequest();
        request.withSecretName("{secret_name}");
        request.withStageName("{stage_name}");
        try {
            DeleteSecretStageResponse response = client.deleteSecretStage(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcsms.v1.region.csms_region import CsmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcsms.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CsmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CsmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = DeleteSecretStageRequest()
        request.secret_name = "{secret_name}"
        request.stage_name = "{stage_name}"
        response = client.delete_secret_stage(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    csms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := csms.NewCsmsClient(
        csms.CsmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.DeleteSecretStageRequest{}
    request.SecretName = "{secret_name}"
    request.StageName = "{stage_name}"
    response, err := client.DeleteSecretStage(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
204	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误

状态码	描述
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.3.4 凭据标签管理

4.3.4.1 查询凭据实例

功能介绍

查询凭据实例。通过标签过滤，筛选用户凭据，返回凭据列表。

调用方法

请参见**如何调用API**。

URI

POST /v1/{project_id}/csms/{resource_instances}/action

表 4-1228 路径参数

参数	是否必选	参数类型	描述
resource_instances	是	String	定值为resource_instances。
project_id	是	String	项目ID

请求参数

表 4-1229 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-1230 请求 Body 参数

参数	是否必选	参数类型	描述
limit	否	String	查询记录数（“action”为“count”时，无需设置此参数），如果“action”为“filter”，默认为“10”。 limit的取值范围为“1-1000”。
offset	否	String	索引位置。从offset指定的下一条数据开始查询。查询第一页数据时，将查询前一页数据时响应体中的值带入此参数（“action”为“count”时，无需设置此参数）。如果“action”为“filter”，offset默认为“0”。 offset必须为数字，不能为负数。
action	是	String	操作标识（可设置为“filter”或者“count”）。 • filter：表示过滤。 • count：表示查询总条数。
tags	否	Array of **Tag** objects	标签列表，key和value键值对的集合。最多不超过10个。
matches	否	Array of **TagMatches** objects	搜索字段。 • key为搜索的字段，目前仅支持搜索凭据名称，值为“resource_name”。 • value为模糊匹配的值，最大长度为255个字符。为空返回空值。
sequence	否	String	请求消息序列号，36字节序列号。 例如：919c82d4-8046-4722-9094-35c3c6524cff

表 4-1231 Tag

参数	是否必选	参数类型	描述
key	否	String	标签键。

参数	是否必选	参数类型	描述
values	否	Array of strings	标签值集合。 约束：最多包含10个value。标签列表中的标签value值不允许重复。标签列表如果为空列表，表示匹配任意值。标签列表中多个value之间是“或”的关系，在key已经满足要求的前提下，满足请求中的某个value就会匹配出来。

表 4-1232 TagMatches

参数	是否必选	参数类型	描述
key	否	String	为要匹配的字段。 约束：值只能为resource_name。
value	否	String	模糊匹配的值。 约束：最大长度为255个字符，为空返回空值。

响应参数

状态码： 200

表 4-1233 响应 Body 参数

参数	参数类型	描述
resources	Array of **ActionResources** objects	资源实例列表，详情请参见resource字段数据结构说明。
total_count	Integer	总记录数。

表 4-1234 ActionResources

参数	参数类型	描述
resource_id	String	资源ID。
resource_detail	**Secret** object	凭据对象。

参数	参数类型	描述
resource_name	String	资源名称，默认为空字符串。
tags	Array of **TagItem** objects	标签列表，没有标签，数组默认为空。
sys_tags	Array of **SysTag** objects	系统标签列表，没有标签，数组默认为空

表 4-1235 Secret

参数	参数类型	描述
id	String	凭据的资源标识符。
name	String	凭据名称。
state	String	凭据状态，取值如下： ENABLED：表示启用状态 DISABLED：表示禁用状态 PENDING_DELETE：表示待删除状态 FROZEN：表示冻结状态
kms_key_id	String	用于加密凭据值的KMS主密钥的ID值。
description	String	凭据的描述信息。
create_time	Long	凭据创建时间，时间戳，即从1970年1月1日至该时间的总秒数。
update_time	Long	凭据上次更新时间，时间戳，即从1970年1月1日至该时间的总秒数。
scheduled_delete_time	Long	凭据计划删除时间，时间戳，即从1970年1月1日至该时间的总秒数。 凭据不在删除计划中时，本项值为null。

参数	参数类型	描述
secret_type	String	凭据类型 ● COMMON：通用凭据(默认)。用于应用系统中的各种敏感信息储存。 ● RDS：RDS凭据 。专门针对RDS的凭据，用于存储RDS的账号信息。（ 已不支持，使用RDS-FG替代 ） ● RDS-FG：RDS凭据 。专门针对RDS的凭据，用于存储RDS的账号信息。 ● GaussDB-FG：TaurusDB凭据。专门针对TaurusDB的凭据，用于存储TaurusDB的账号信息。
auto_rotation	Boolean	自动轮转 取值：true 开启, false 关闭(默认)
rotation_period	String	轮转周期 约束：6小时-8,760小时 （ 365天 ） 类型：Integer[unit] ，Integer表示时间长度 。unit表示时间单位，d（ 天 ）、h（ 小时 ）、m（ 分钟 ）、s（ 秒 ）。例如 1d 表示一天，24h也表示一天 说明：当开启自动轮转时，必须填写该值
rotation_config	String	轮转配置 约束：范围不超过1024个字符。 当secret_type为RDS-FG、GaussDB-FG时，配置为{"InstanceId":"","SecretSubType":""} 说明：当secret_type为RDS-FG、GaussDB-FG时，必须填写该值 InstanceId为实例ID,SecretSubType为轮转子类型，取值为：SingleUser，MultiUser。 SingleUser：指定轮转类型为单用户模式轮转，每次轮转将指定账号重置为新的口令。 MultiUser：指定轮转类型为双用户模式轮转，SYSCURRENT和SYSPREVIOUS分别引用其中一个账号。凭据轮转时，SYSPREVIOUS引用的账号口令会被重置为新的随机口令，随后凭据交换SYSCURRENT和SYSPREVIOUS对账号的引用。
rotation_time	Long	轮转时间戳
next_rotation_time	Long	下一次轮转时间戳
event_subscriptions	Array of strings	凭据订阅的事件列表，当前最大可订阅一个事件。当事件包含的基础事件触发时，通知消息将发送到事件对应的通知主题。

参数	参数类型	描述
enterprise_project_id	String	企业项目ID
rotation_func_urn	String	FunctionGraph函数的urn。

表 4-1236 TagItem

参数	参数类型	描述
key	String	标签的名称。 同一个凭据，一个标签键只能对应一个标签值；不同的凭据可以使用相同的标签键。 用户最多可以给单个凭据添加20个标签。 约束：取值范围为1到128个字符，满足正则匹配"^((?!\s)(?!*sys*)[\p{L}\p{Z}\p{N}_.:=+\-@]*)(?<!\s)$"
value	String	标签的值。 约束：取值范围不超过255个字符，满足正则匹配"^([\p{L}\p{Z}\p{N}_.:\/=+\-@]*)$"

表 4-1237 SysTag

参数	参数类型	描述
key	String	标签键。
value	String	标签值。

状态码： 400

表 4-1238 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1239 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-1240 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1241 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-1242 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1243 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-1244 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1245 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-1246 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1247 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-1248 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1249 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-1250 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1251 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

通过标签过滤，筛选用户凭据，返回凭据列表。

```
{
  "action" : "filter",
  "tags" : [ {
    "key" : "key1",
    "values" : [ "val1" ]
  } ]
}
```

响应示例

状态码： 200

请求已成功

```
{
  "total_count" : 1,
  "resources" : [ {
    "resource_id" : "2d1152f2-290d-4756-a1d2-e12c14992416"
  }, {
    "resource_detail" : {
      "id" : "2d1152f2-290d-4756-a1d2-e12c14992416",
      "name" : "example_name",
      "state" : "ENABLED",
      "description" : "",
      "kms_key_id" : "1213d410-ass1-1254-1a2d-3cca2sa2w554",
      "create_time" : 1581507580000,
```

```
    "update_time" : 1581507580000,
    "scheduled_delete_time" : 1581507580000
  }
 }, {
  "tags" : [ {
    "key" : "key1",
    "value" : "value1"
  }, {
    "key" : "key2",
    "value" : "value2"
  } ]
 }, {
  "sys_tags" : null
 }, {
  "resource_name" : "example_name"
 } ]
}
```

SDK 代码示例

SDK代码示例如下。

Java

通过标签过滤，筛选用户凭据，返回凭据列表。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.csms.v1.region.CsmsRegion;
import com.huaweicloud.sdk.csms.v1.*;
import com.huaweicloud.sdk.csms.v1.model.*;

import java.util.List;
import java.util.ArrayList;

public class ListResourceInstancesSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CsmsClient client = CsmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(CsmsRegion.valueOf("<YOUR REGION>"))
                .build();
        ListResourceInstancesRequest request = new ListResourceInstancesRequest();
        request.withResourceInstances("{resource_instances}");
        ListResourceInstancesRequestBody body = new ListResourceInstancesRequestBody();
        List<String> listTagsValues = new ArrayList<>();
        listTagsValues.add("val1");
        List<Tag> listbodyTags = new ArrayList<>();
```

```
        listbodyTags.add(
            new Tag()
                .withKey("key1")
                .withValues(listTagsValues)
        );
        body.withTags(listbodyTags);
        body.withAction("filter");
        request.withBody(body);
        try {
            ListResourceInstancesResponse response = client.listResourceInstances(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

通过标签过滤，筛选用户凭据，返回凭据列表。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcsms.v1.region.csms_region import CsmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcsms.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CsmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CsmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ListResourceInstancesRequest()
        request.resource_instances = "{resource_instances}"
        listValuesTags = [
            "val1"
        ]
        listTagsbody = [
            Tag(
                key="key1",
                values=listValuesTags
            )
        ]
        request.body = ListResourceInstancesRequestBody(
            tags=listTagsbody,
            action="filter"
```

```
        )
        response = client.list_resource_instances(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

通过标签过滤，筛选用户凭据，返回凭据列表。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    csms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := csms.NewCsmsClient(
        csms.CsmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ListResourceInstancesRequest{}
    request.ResourceInstances = "{resource_instances}"
    var listValuesTags = []string{
        "val1",
    }
    keyTags:= "key1"
    var listTagsbody = []model.Tag{
        {
            Key: &keyTags,
            Values: &listValuesTags,
        },
    }
    request.Body = &model.ListResourceInstancesRequestBody{
        Tags: &listTagsbody,
        Action: "filter",
    }
    response, err := client.ListResourceInstances(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.3.4.2 批量添加或删除凭据标签

功能介绍

- 功能介绍：批量添加或删除凭据标签。

调用方法

请参见**如何调用API**。

URI

POST /v1/{project_id}/csms/{secret_id}/tags/action

表 4-1252 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID
secret_id	是	String	凭据ID

请求参数

表 4-1253 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-1254 请求 Body 参数

参数	是否必选	参数类型	描述
tags	是	Array of **TagItem** objects	标签列表，key和value键值对的集合。
action	是	String	操作标识： 仅限于“create”和“delete”。
sequence	否	String	请求消息序列号，36字节序列号。 例如：919c82d4-8046-4722-9094-35c3c6524cff

表 4-1255 TagItem

参数	是否必选	参数类型	描述
key	是	String	标签的名称。 同一个凭据，一个标签键只能对应一个标签值；不同的凭据可以使用相同的标签键。 用户最多可以给单个凭据添加20个标签。 约束：取值范围为1到128个字符，满足正则匹配"^((?!\s)(?!*sys*)[\p{L}\p{Z}\p{N}_.:=+\-@]*)(?<!\s)$"
value	否	String	标签的值。 约束：取值范围不超过255个字符，满足正则匹配"^([\p{L}\p{Z}\p{N}_.:\/=+\-@]*)$"

响应参数

状态码： 400

表 4-1256 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1257 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-1258 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1259 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-1260 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1261 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-1262 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1263 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-1264 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1265 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-1266 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1267 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-1268 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1269 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

批量添加凭据标签。

```
{
  "action" : "create",
  "tags" : [ {
    "key" : "key1",
    "value" : "value1"
  }, {
    "key" : "key2",
    "value" : "value2"
  } ]
}
```

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

批量添加凭据标签。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.csms.v1.region.CsmsRegion;
import com.huaweicloud.sdk.csms.v1.*;
import com.huaweicloud.sdk.csms.v1.model.*;

import java.util.List;
import java.util.ArrayList;

public class BatchCreateOrDeleteTagsSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CsmsClient client = CsmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(CsmsRegion.valueOf("<YOUR REGION>"))
                .build();
        BatchCreateOrDeleteTagsRequest request = new BatchCreateOrDeleteTagsRequest();
        request.withSecretId("{secret_id}");
        BatchCreateOrDeleteTagsRequestBody body = new BatchCreateOrDeleteTagsRequestBody();
        List<TagItem> listbodyTags = new ArrayList<>();
        listbodyTags.add(
            new TagItem()
                .withKey("key1")
                .withValue("value1")
        );
        listbodyTags.add(
            new TagItem()
                .withKey("key2")
                .withValue("value2")
        );
        body.withAction("create");
        body.withTags(listbodyTags);
        request.withBody(body);
        try {
            BatchCreateOrDeleteTagsResponse response = client.batchCreateOrDeleteTags(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
```

```
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

批量添加凭据标签。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcsms.v1.region.csms_region import CsmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcsms.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CsmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CsmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = BatchCreateOrDeleteTagsRequest()
        request.secret_id = "{secret_id}"
        listTagsbody = [
            TagItem(
                key="key1",
                value="value1"
            ),
            TagItem(
                key="key2",
                value="value2"
            )
        ]
        request.body = BatchCreateOrDeleteTagsRequestBody(
            action="create",
            tags=listTagsbody
        )
        response = client.batch_create_or_delete_tags(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

批量添加凭据标签。

```
package main
```

```
import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    csms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := csms.NewCsmsClient(
        csms.CsmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.BatchCreateOrDeleteTagsRequest{}
    request.SecretId = "{secret_id}"
    valueTags:= "value1"
    valueTags1:= "value2"
    var listTagsbody = []model.TagItem{
        {
            Key: "key1",
            Value: &valueTags,
        },
        {
            Key: "key2",
            Value: &valueTags1,
        },
    }
    request.Body = &model.BatchCreateOrDeleteTagsRequestBody{
        Action: "create",
        Tags: listTagsbody,
    }
    response, err := client.BatchCreateOrDeleteTags(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
204	No Content

状态码	描述
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.3.4.3 查询凭据标签

功能介绍

查询凭据标签。

调用方法

请参见**如何调用API**。

URI

GET /v1/{project_id}/csms/{secret_id}/tags

表 4-1270 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID
secret_id	是	String	凭据ID

请求参数

表 4-1271 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码： 200

表 4-1272 响应 Body 参数

参数	参数类型	描述
tags	Array of **TagItem** objects	标签列表，key和value键值对的集合。 • key：表示标签键，一个凭据下最多包含20个key，key不能为空，不能重复，同一个key中value不能重复。key最大长度为128个字符。 • value：表示标签值。每个值最大长度255个字符，value之间为“与”的关系。
sys_tags	Array of **SysTag** objects	系统标签列表。

表 4-1273 TagItem

参数	参数类型	描述
key	String	标签的名称。 同一个凭据，一个标签键只能对应一个标签值；不同的凭据可以使用相同的标签键。 用户最多可以给单个凭据添加20个标签。 约束：取值范围为1到128个字符，满足正则匹配"^((?!\s)(?!*sys*)[\p{L}\p{Z}\p{N}_.:=+\-@]*)(?<!\s)$"
value	String	标签的值。 约束：取值范围不超过255个字符，满足正则匹配"^([\p{L}\p{Z}\p{N}_.:\/=+\-@]*)$"

表 4-1274 SysTag

参数	参数类型	描述
key	String	标签键。
value	String	标签值。

状态码： 400

表 4-1275 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1276 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-1277 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1278 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-1279 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1280 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-1281 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1282 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-1283 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1284 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-1285 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1286 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-1287 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1288 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

无

响应示例

状态码： 200

请求已成功

```
{
  "tags" : [ {
    "key" : "key1",
    "value" : "value1"
  }, {
    "key" : "key2",
    "value" : "value2"
  } ],
  "sys_tags" : null
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.csms.v1.region.CsmsRegion;
import com.huaweicloud.sdk.csms.v1.*;
import com.huaweicloud.sdk.csms.v1.model.*;


public class ListSecretTagsSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CsmsClient client = CsmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(CsmsRegion.valueOf("<YOUR REGION>"))
                .build();
        ListSecretTagsRequest request = new ListSecretTagsRequest();
        request.withSecretId("{secret_id}");
        try {
            ListSecretTagsResponse response = client.listSecretTags(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
```

```
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcsms.v1.region.csms_region import CsmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcsms.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CsmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CsmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ListSecretTagsRequest()
        request.secret_id = "{secret_id}"
        response = client.list_secret_tags(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    csms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
```

```
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := csms.NewCsmsClient(
        csms.CsmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ListSecretTagsRequest{}
    request.SecretId = "{secret_id}"
    response, err := client.ListSecretTags(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.3.4.4 添加凭据标签

功能介绍

添加凭据标签。

调用方法

请参见**如何调用API**。

URI

POST /v1/{project_id}/csms/{secret_id}/tags

表 4-1289 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID
secret_id	是	String	凭据ID

请求参数

表 4-1290 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-1291 请求 Body 参数

参数	是否必选	参数类型	描述
tag	是	**TagItem** object	标签信息。

表 4-1292 TagItem

参数	是否必选	参数类型	描述
key	是	String	标签的名称。 同一个凭据，一个标签键只能对应一个标签值；不同的凭据可以使用相同的标签键。 用户最多可以给单个凭据添加20个标签。 约束：取值范围为1到128个字符，满足正则匹配"^((?!\s)(?!*sys*)[\p{L}\p{Z}\p{N}_.:=+\-@]*)(?<!\s)$"
value	否	String	标签的值。 约束：取值范围不超过255个字符，满足正则匹配"^([\p{L}\p{Z}\p{N}_.:\/=+\-@]*)$"

响应参数

状态码： 400

表 4-1293 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1294 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-1295 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1296 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-1297 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1298 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-1299 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1300 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-1301 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1302 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-1303 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1304 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-1305 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1306 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

添加凭据标签。

```
{
  "tag" : {
    "key" : "DEV",
    "value" : "DEV1"
  }
}
```

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

添加凭据标签。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.csms.v1.region.CsmsRegion;
import com.huaweicloud.sdk.csms.v1.*;
import com.huaweicloud.sdk.csms.v1.model.*;


public class CreateSecretTagSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CsmsClient client = CsmsClient.newBuilder()
```

```
            .withCredential(auth)
            .withRegion(CsmsRegion.valueOf("<YOUR REGION>"))
            .build();
        CreateSecretTagRequest request = new CreateSecretTagRequest();
        request.withSecretId("{secret_id}");
        CreateSecretTagRequestBody body = new CreateSecretTagRequestBody();
        TagItem tagbody = new TagItem();
        tagbody.withKey("DEV")
            .withValue("DEV1");
        body.withTag(tagbody);
        request.withBody(body);
        try {
            CreateSecretTagResponse response = client.createSecretTag(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

添加凭据标签。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcsms.v1.region.csms_region import CsmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcsms.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CsmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CsmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = CreateSecretTagRequest()
        request.secret_id = "{secret_id}"
        tagbody = TagItem(
            key="DEV",
            value="DEV1"
        )
        request.body = CreateSecretTagRequestBody(
            tag=tagbody
        )
        response = client.create_secret_tag(request)
        print(response)
```

```
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

添加凭据标签。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    csms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := csms.NewCsmsClient(
        csms.CsmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.CreateSecretTagRequest{}
    request.SecretId = "{secret_id}"
    valueTag:= "DEV1"
    tagbody := &model.TagItem{
        Key: "DEV",
        Value: &valueTag,
    }
    request.Body = &model.CreateSecretTagRequestBody{
        Tag: tagbody,
    }
    response, err := client.CreateSecretTag(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
204	No Content
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.3.4.5 删除凭据标签

功能介绍

删除凭据标签。

调用方法

请参见**如何调用API**。

URI

DELETE /v1/{project_id}/csms/{secret_id}/tags/{key}

表 4-1307 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID
secret_id	是	String	凭据ID
key	是	String	标签键的值

请求参数

表 4-1308 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码： 400

表 4-1309 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1310 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-1311 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1312 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-1313 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1314 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-1315 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1316 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-1317 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1318 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-1319 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1320 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-1321 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1322 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

无

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.csms.v1.region.CsmsRegion;
import com.huaweicloud.sdk.csms.v1.*;
import com.huaweicloud.sdk.csms.v1.model.*;


public class DeleteSecretTagSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CsmsClient client = CsmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(CsmsRegion.valueOf("<YOUR REGION>"))
                .build();
        DeleteSecretTagRequest request = new DeleteSecretTagRequest();
        request.withSecretId("{secret_id}");
        request.withKey("{key}");
        try {
            DeleteSecretTagResponse response = client.deleteSecretTag(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8
```

```
import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcsms.v1.region.csms_region import CsmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcsms.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CsmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CsmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = DeleteSecretTagRequest()
        request.secret_id = "{secret_id}"
        request.key = "{key}"
        response = client.delete_secret_tag(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    csms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := csms.NewCsmsClient(
        csms.CsmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())
```

```
    request := &model.DeleteSecretTagRequest{}
    request.SecretId = "{secret_id}"
    request.Key = "{key}"
    response, err := client.DeleteSecretTag(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
204	No Content
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.3.4.6 查询项目标签

功能介绍

查询用户在指定项目下的所有凭据标签集合。

调用方法

请参见**如何调用API**。

URI

GET /v1/{project_id}/csms/tags

表 4-1323 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID

请求参数

表 4-1324 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码： 200

表 4-1325 响应 Body 参数

参数	参数类型	描述
tags	Array of **TagResponse** objects	标签列表，key和value键值对的集合。 • key：表示标签键，一个凭据下最多包含20个key，key不能为空，不能重复，同一个key中value不能重复。key最大长度为128个字符。 • value：表示标签值。每个值最大长度255个字符，value之间为“与”的关系。

表 4-1326 TagResponse

参数	参数类型	描述
key	String	键。
values	Array of strings	标签值集合

状态码： 400

表 4-1327 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1328 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-1329 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1330 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-1331 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1332 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-1333 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1334 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-1335 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1336 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-1337 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1338 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-1339 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1340 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

无

响应示例

状态码： 200

请求已成功

```
{
  "tags" : [ {
    "key" : "key1",
    "values" : [ "val1" ]
  }, {
    "key" : "key2",
    "values" : [ "val2" ]
```

```
  } ]
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.csms.v1.region.CsmsRegion;
import com.huaweicloud.sdk.csms.v1.*;
import com.huaweicloud.sdk.csms.v1.model.*;


public class ListProjectSecretsTagsSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CsmsClient client = CsmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(CsmsRegion.valueOf("<YOUR REGION>"))
                .build();
        ListProjectSecretsTagsRequest request = new ListProjectSecretsTagsRequest();
        try {
            ListProjectSecretsTagsResponse response = client.listProjectSecretsTags(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcsms.v1.region.csms_region import CsmsRegion
```

```
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcsms.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CsmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CsmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ListProjectSecretsTagsRequest()
        response = client.list_project_secrets_tags(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    csms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := csms.NewCsmsClient(
        csms.CsmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ListProjectSecretsTagsRequest{}
    response, err := client.ListProjectSecretsTags(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
```

```
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.3.5 事件管理

4.3.5.1 创建事件

功能介绍

创建事件，事件可配置在一个或多个凭据对象上。当事件为启用状态且包含的基础事件类型在凭据对象上触发时，云服务会将对应的事件通知发送至事件指定的通知主题上。

接口约束

创建的事件通知无法单独使用，需要配置在具体的凭据对象上。服务不会对事件通知中的通知对象进行有效性检查，如果用户账号下不存在该通知对象，则会造成事件触发时，事件通知失败。

调用方法

请参见**如何调用API**。

URI

POST /v1/{project_id}/csms/events

表 4-1341 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。

请求参数

表 4-1342 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-1343 请求 Body 参数

参数	是否必选	参数类型	描述
name	是	String	新创建事件通知的名称。 约束：取值范围为1到64个字符，满足正则匹配“^[a-zA-Z0-9_-]{1,64}$”。
event_types	是	Array of strings	本次事件通知的基础事件列表，基础事件类型如下。 ● SECRET_VERSION_CREATED：版本创建 ● SECRET_VERSION_EXPIRED：版本过期 ● SECRET_ROTATED：凭据轮转 ● SECRET_DELETED：凭据删除 ● SECRET_ROTATED_FAILED: 凭据轮转失败 列表包含的基础事件类型不能重复。

参数	是否必选	参数类型	描述
state	是	String	控制事件是否生效，只有启用状态才能触发包含的基础事件类型 ENABLED：启用 DISABLED：禁用
notification	是	**Notification** object	通知主题对象。

表 4-1344 Notification

参数	是否必选	参数类型	描述
target_type	是	String	事件通知的对象类型。
target_id	是	String	事件通知的对象ID。
target_name	是	String	事件通知的对象名称。

响应参数

状态码： 200

表 4-1345 响应 Body 参数

参数	参数类型	描述
event	**Event** object	事件通知对象。

表 4-1346 Event

参数	参数类型	描述
name	String	事件通知名称。
event_id	String	事件通知的资源标识符。
event_types	Array of strings	设置事件的基础事件类型列表,。 约束：数组大小：最小1，最大12。
state	String	事件通知状态，取值如下。 ENABLED：表示启用状态 DISABLED：表示禁用状态
create_time	Long	事件通知创建时间，时间戳，即从1970年1月1日至该时间的总秒数。

参数	参数类型	描述
update_time	Long	事件通知上次更新时间，时间戳，即从1970年1月1日至该时间的总秒数。
notification	**Notification** object	通知主题对象。

表 4-1347 Notification

参数	参数类型	描述
target_type	String	事件通知的对象类型。
target_id	String	事件通知的对象ID。
target_name	String	事件通知的对象名称。

状态码： 400

表 4-1348 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1349 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-1350 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1351 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-1352 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1353 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-1354 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1355 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-1356 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1357 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-1358 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1359 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-1360 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1361 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

创建事件。

```
{
  "name" : "demo-event",
  "event_types" : [ "SECRET_VERSION_CREATED", "SECRET_VERSION_EXPIRED" ],
  "state" : "ENABLED",
  "notification" : {
    "target_type" : "SMN",
    "target_id" : "urn:smn:cn-north-4:dc3b7c85759141a991da17423c0f2068:test-poc",
    "target_name" : "demo-smn-name"
  }
}
```

响应示例

状态码： 200

请求已成功

```
{
  "event" : {
    "name" : "event-test",
    "event_id" : "bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e",
    "state" : "ENABLED",
    "event_types" : [ "SECRET_VERSION_CREATED", "SECRET_VERSION_EXPIRED" ],
    "create_time" : 1581507580000,
    "update_time" : 1581507580000,
    "notification" : {
      "target_type" : "SMN",
      "target_id" : "urn:smn:cn-north-4:SecertExpirationTest",
      "target_name" : "SecertExpirationNotificationTest"
    }
  }
}
```

SDK 代码示例

SDK代码示例如下。

Java

创建事件。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.csms.v1.region.CsmsRegion;
```

```
import com.huaweicloud.sdk.csms.v1.*;
import com.huaweicloud.sdk.csms.v1.model.*;

import java.util.List;
import java.util.ArrayList;

public class CreateSecretEventSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CsmsClient client = CsmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(CsmsRegion.valueOf("<YOUR REGION>"))
                .build();
        CreateSecretEventRequest request = new CreateSecretEventRequest();
        CreateSecretEventRequestBody body = new CreateSecretEventRequestBody();
        Notification notificationbody = new Notification();
        notificationbody.withTargetType("SMN")
            .withTargetId("urn:smn:cn-north-4:dc3b7c85759141a991da17423c0f2068:test-poc")
            .withTargetName("demo-smn-name");
        List<CreateSecretEventRequestBody.EventTypesEnum> listbodyEventTypes = new ArrayList<>();

listbodyEventTypes.add(CreateSecretEventRequestBody.EventTypesEnum.fromValue("SECRET_VERSION_CREATED"));

listbodyEventTypes.add(CreateSecretEventRequestBody.EventTypesEnum.fromValue("SECRET_VERSION_EXPIRED"));
        body.withNotification(notificationbody);
        body.withState(CreateSecretEventRequestBody.StateEnum.fromValue("ENABLED"));
        body.withEventTypes(listbodyEventTypes);
        body.withName("demo-event");
        request.withBody(body);
        try {
            CreateSecretEventResponse response = client.createSecretEvent(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

创建事件。

```
# coding: utf-8

import os
```

```
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcsms.v1.region.csms_region import CsmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcsms.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CsmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CsmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = CreateSecretEventRequest()
        notificationbody = Notification(
            target_type="SMN",
            target_id="urn:smn:cn-north-4:dc3b7c85759141a991da17423c0f2068:test-poc",
            target_name="demo-smn-name"
        )
        listEventTypesbody = [
            "SECRET_VERSION_CREATED",
            "SECRET_VERSION_EXPIRED"
        ]
        request.body = CreateSecretEventRequestBody(
            notification=notificationbody,
            state="ENABLED",
            event_types=listEventTypesbody,
            name="demo-event"
        )
        response = client.create_secret_event(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

创建事件。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    csms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
```

```
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := csms.NewCsmsClient(
        csms.CsmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.CreateSecretEventRequest{}
    notificationbody := &model.Notification{
        TargetType: "SMN",
        TargetId: "urn:smn:cn-north-4:dc3b7c85759141a991da17423c0f2068:test-poc",
        TargetName: "demo-smn-name",
    }
    var listEventTypesbody = []model.CreateSecretEventRequestBodyEventTypes{
        model.GetCreateSecretEventRequestBodyEventTypesEnum().SECRET_VERSION_CREATED,
        model.GetCreateSecretEventRequestBodyEventTypesEnum().SECRET_VERSION_EXPIRED,
    }
    request.Body = &model.CreateSecretEventRequestBody{
        Notification: notificationbody,
        State: model.GetCreateSecretEventRequestBodyStateEnum().ENABLED,
        EventTypes: listEventTypesbody,
        Name: "demo-event",
    }
    response, err := client.CreateSecretEvent(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.3.5.2 查询事件

功能介绍

查询指定事件的信息。

接口约束

此接口返回的信息为凭据事件通知的元数据信息值。

调用方法

请参见**如何调用API**。

URI

GET /v1/{project_id}/csms/events/{event_name}

表 4-1362 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。
event_name	是	String	事件通知的名称。

请求参数

表 4-1363 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码： 200

表 4-1364 响应 Body 参数

参数	参数类型	描述
event	**Event** object	事件通知对象。

表 4-1365 Event

参数	参数类型	描述
name	String	事件通知名称。
event_id	String	事件通知的资源标识符。
event_types	Array of strings	设置事件的基础事件类型列表,。 约束：数组大小：最小1，最大12。
state	String	事件通知状态，取值如下。 ENABLED：表示启用状态 DISABLED：表示禁用状态
create_time	Long	事件通知创建时间，时间戳，即从1970年1月1日至该时间的总秒数。
update_time	Long	事件通知上次更新时间，时间戳，即从1970年1月1日至该时间的总秒数。
notification	**Notification** object	通知主题对象。

表 4-1366 Notification

参数	参数类型	描述
target_type	String	事件通知的对象类型。
target_id	String	事件通知的对象ID。
target_name	String	事件通知的对象名称。

状态码： 400

表 4-1367 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1368 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-1369 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1370 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-1371 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1372 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-1373 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1374 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-1375 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1376 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-1377 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1378 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-1379 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1380 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

无

响应示例

状态码： 200

请求已成功

```
{
  "event" : {
    "name" : "event-test",
    "event_id" : "bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e",
    "state" : "ENABLED",
    "event_types" : [ "SECRET_VERSION_EXPIRED" ],
    "create_time" : 1581507580000,
    "update_time" : 1581507580000,
    "notification" : {
      "target_type" : "SMN",
      "target_id" : "urn:smn:cn-north-4:SecertExpirationTest",
      "target_name" : "SecertExpirationNotificationTest"
    }
  }
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.csms.v1.region.CsmsRegion;
import com.huaweicloud.sdk.csms.v1.*;
import com.huaweicloud.sdk.csms.v1.model.*;


public class ShowSecretEventSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CsmsClient client = CsmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(CsmsRegion.valueOf("<YOUR REGION>"))
                .build();
        ShowSecretEventRequest request = new ShowSecretEventRequest();
        request.withEventName("{event_name}");
        try {
            ShowSecretEventResponse response = client.showSecretEvent(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcsms.v1.region.csms_region import CsmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcsms.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this
```

```
example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CsmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CsmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ShowSecretEventRequest()
        request.event_name = "{event_name}"
        response = client.show_secret_event(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    csms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := csms.NewCsmsClient(
        csms.CsmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ShowSecretEventRequest{}
    request.EventName = "{event_name}"
    response, err := client.ShowSecretEvent(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.3.5.3 查询事件列表

功能介绍

查询当前用户在本项目下创建的所有事件。

接口约束

此接口返回的信息为事件通知的元数据信息，不包含凭据值。

调用方法

请参见**如何调用API**。

URI

GET /v1/{project_id}/csms/events

表 4-1381 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。

表 4-1382 Query 参数

参数	是否必选	参数类型	描述
limit	否	String	每页返回的个数。 默认值：50。
marker	否	String	分页查询起始的事件名称，为空时为查询第一页

请求参数

表 4-1383 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码： 200

表 4-1384 响应 Body 参数

参数	参数类型	描述
events	Array of **Event** objects	事件详情列表。
page_info	**PageInfo** object	分页信息。

表 4-1385 Event

参数	参数类型	描述
name	String	事件通知名称。
event_id	String	事件通知的资源标识符。
event_types	Array of strings	设置事件的基础事件类型列表,。 约束：数组大小：最小1，最大12。

参数	参数类型	描述
state	String	事件通知状态，取值如下。 ENABLED：表示启用状态 DISABLED：表示禁用状态
create_time	Long	事件通知创建时间，时间戳，即从1970年1月1日至该时间的总秒数。
update_time	Long	事件通知上次更新时间，时间戳，即从1970年1月1日至该时间的总秒数。
notification	**Notification** object	通知主题对象。

表 4-1386 Notification

参数	参数类型	描述
target_type	String	事件通知的对象类型。
target_id	String	事件通知的对象ID。
target_name	String	事件通知的对象名称。

表 4-1387 PageInfo

参数	参数类型	描述
next_marker	String	下一页查询地址（本页的末尾凭据名称，下一页起始凭据名称）。
previous_marker	String	本页的起始凭据名称，上一页末尾凭据名称。
current_count	Integer	本页返回条目数量。

状态码： 400

表 4-1388 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1389 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-1390 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1391 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-1392 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1393 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-1394 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1395 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-1396 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1397 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-1398 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1399 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-1400 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1401 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

无

响应示例

状态码： 200

请求已成功

```
{
  "events" : [ {
    "state" : "ENABLED",
    "name" : "TestEvent1",
    "event_id" : "120eeafd-eca5-4098-8733-c1b369f3a450",
    "event_types" : [ "SECRET_VERSION_CREATED", "SECRET_VERSION_EXPIRED",
"SECRET_VERSION_NEAR_EXPIRY", "SECRET_DELETED" ],
    "create_time" : 1669042498000,
    "update_time" : 1669042498000,
    "notification" : {
      "target_type" : "SMN",
      "target_id" : "urn:smn:cn-north-4:dc3b7c85759141a991da17423c0f2068:smn-target-name",
      "target_name" : "smn-target-name"
    }
  }, {
    "state" : "ENABLED",
    "name" : "TestEvent2",
    "event_id" : "1d251c99-37d2-4396-86ce-f000d3aa9850",
    "event_types" : [ "SECRET_VERSION_CREATED", "SECRET_VERSION_EXPIRED",
```

```
"SECRET_VERSION_NEAR_EXPIRY", "SECRET_DELETED" ],
    "create_time" : 1669041491000,
    "update_time" : 1669041491000,
    "notification" : {
      "target_type" : "SMN",
      "target_id" : "urn:smn:cn-north-4:dc3b7c85759141a991da17423c0f2068:smn-target-name",
      "target_name" : "smn-target-name"
    }
  } ],
  "page_info" : {
    "next_marker" : "TestEvent2",
    "previous_marker" : "TestEvent3",
    "current_count" : 2
  }
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.csms.v1.region.CsmsRegion;
import com.huaweicloud.sdk.csms.v1.*;
import com.huaweicloud.sdk.csms.v1.model.*;


public class ListSecretEventsSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CsmsClient client = CsmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(CsmsRegion.valueOf("<YOUR REGION>"))
                .build();
        ListSecretEventsRequest request = new ListSecretEventsRequest();
        try {
            ListSecretEventsResponse response = client.listSecretEvents(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
```

```
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcsms.v1.region.csms_region import CsmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcsms.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CsmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CsmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ListSecretEventsRequest()
        response = client.list_secret_events(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    csms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()
```

```
    client := csms.NewCsmsClient(
        csms.CsmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ListSecretEventsRequest{}
    response, err := client.ListSecretEvents(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.3.5.4 更新事件

功能介绍

更新指定事件的元数据信息。支持更新的元数据包含事件启用状态、基础类型列表、通知主题。

接口约束

此接口仅能修改凭据事件通知的元数据信息。

调用方法

请参见**如何调用API**。

URI

PUT /v1/{project_id}/csms/events/{event_name}

表 4-1402 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。
event_name	是	String	事件通知名称。

请求参数

表 4-1403 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

表 4-1404 请求 Body 参数

参数	是否必选	参数类型	描述
state	否	String	事件通知状态，取值如下。 ENABLED：表示启用状态 DISABLED：表示禁用状态

参数	是否必选	参数类型	描述
event_types	否	Array of strings	本次事件通知的基础事件列表，基础事件类型如下。 • SECRET_VERSION_CREATED:版本创建 • SECRET_VERSION_EXPIRED:版本过期 • SECRET_ROTATED:凭据轮转成功 • SECRET_DELETED:凭据删除 • SECRET_ROTATED_FAILED:凭据轮转失败 列表包含的基础事件类型不能重复。
notification	否	**Notification** object	通知主题对象。

表 4-1405 Notification

参数	是否必选	参数类型	描述
target_type	是	String	事件通知的对象类型。
target_id	是	String	事件通知的对象ID。
target_name	是	String	事件通知的对象名称。

响应参数

状态码： 200

表 4-1406 响应 Body 参数

参数	参数类型	描述
event	**Event** object	事件通知对象。

表 4-1407 Event

参数	参数类型	描述
name	String	事件通知名称。
event_id	String	事件通知的资源标识符。

参数	参数类型	描述
event_types	Array of strings	设置事件的基础事件类型列表,。 约束：数组大小：最小1，最大12。
state	String	事件通知状态，取值如下。 ENABLED：表示启用状态 DISABLED：表示禁用状态
create_time	Long	事件通知创建时间，时间戳，即从1970年1月1日至该时间的总秒数。
update_time	Long	事件通知上次更新时间，时间戳，即从1970年1月1日至该时间的总秒数。
notification	**Notification** object	通知主题对象。

表 4-1408 Notification

参数	参数类型	描述
target_type	String	事件通知的对象类型。
target_id	String	事件通知的对象ID。
target_name	String	事件通知的对象名称。

状态码： 400

表 4-1409 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1410 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-1411 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1412 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-1413 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1414 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-1415 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1416 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-1417 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1418 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-1419 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1420 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-1421 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1422 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

更新指定事件。

```
{
  "notification" : {
    "target_type" : "SMN",
    "target_id" : "demo-smn-id",
    "target_name" : "demo-smn-name"
  }
}
```

响应示例

状态码： 200

请求已成功

```
{
  "event" : {
    "name" : "event-test",
    "event_id" : "bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e",
    "state" : "ENABLED",
    "event_types" : [ "SECRET_VERSION_EXPIRED" ],
    "create_time" : 1581507580000,
    "update_time" : 1581507580000,
    "notification" : {
      "target_type" : "SMN",
      "target_id" : "urn:smn:cn-north-4:SecertExpirationTest",
      "target_name" : "SecertExpirationNotificationTest"
    }
  }
}
```

SDK 代码示例

SDK代码示例如下。

Java

更新指定事件。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.csms.v1.region.CsmsRegion;
import com.huaweicloud.sdk.csms.v1.*;
import com.huaweicloud.sdk.csms.v1.model.*;


public class UpdateSecretEventSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CsmsClient client = CsmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(CsmsRegion.valueOf("<YOUR REGION>"))
                .build();
        UpdateSecretEventRequest request = new UpdateSecretEventRequest();
        request.withEventName("{event_name}");
        UpdateSecretEventRequestBody body = new UpdateSecretEventRequestBody();
        Notification notificationbody = new Notification();
        notificationbody.withTargetType("SMN")
            .withTargetId("demo-smn-id")
            .withTargetName("demo-smn-name");
        body.withNotification(notificationbody);
        request.withBody(body);
        try {
            UpdateSecretEventResponse response = client.updateSecretEvent(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

更新指定事件。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcsms.v1.region.csms_region import CsmsRegion
from huaweicloudsdkcore.exceptions import exceptions
```

```
from huaweicloudsdkcsms.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CsmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CsmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = UpdateSecretEventRequest()
        request.event_name = "{event_name}"
        notificationbody = Notification(
            target_type="SMN",
            target_id="demo-smn-id",
            target_name="demo-smn-name"
        )
        request.body = UpdateSecretEventRequestBody(
            notification=notificationbody
        )
        response = client.update_secret_event(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

更新指定事件。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    csms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := csms.NewCsmsClient(
```

```
        csms.CsmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.UpdateSecretEventRequest{}
    request.EventName = "{event_name}"
    notificationbody := &model.Notification{
        TargetType: "SMN",
        TargetId: "demo-smn-id",
        TargetName: "demo-smn-name",
    }
    request.Body = &model.UpdateSecretEventRequestBody{
        Notification: notificationbody,
    }
    response, err := client.UpdateSecretEvent(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.3.5.5 立即删除事件

功能介绍

立即删除指定的事件，且无法恢复。如事件存在凭据引用，则无法删除，请先解除关联。

接口约束

调用此接口删除指定凭据后，不可恢复。

调用方法

请参见**如何调用API**。

URI

DELETE /v1/{project_id}/csms/events/{event_name}

表 4-1423 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。
event_name	是	String	事件通知的名称。

请求参数

表 4-1424 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码： 400

表 4-1425 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1426 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码

参数	参数类型	描述
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-1427 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1428 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-1429 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1430 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-1431 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1432 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-1433 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1434 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-1435 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1436 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-1437 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1438 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

无

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.csms.v1.region.CsmsRegion;
import com.huaweicloud.sdk.csms.v1.*;
import com.huaweicloud.sdk.csms.v1.model.*;


public class DeleteSecretEventSolution {
```

```
    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CsmsClient client = CsmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(CsmsRegion.valueOf("<YOUR REGION>"))
                .build();
        DeleteSecretEventRequest request = new DeleteSecretEventRequest();
        request.withEventName("{event_name}");
        try {
            DeleteSecretEventResponse response = client.deleteSecretEvent(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcsms.v1.region.csms_region import CsmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcsms.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CsmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CsmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = DeleteSecretEventRequest()
        request.event_name = "{event_name}"
        response = client.delete_secret_event(request)
```

```
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    csms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := csms.NewCsmsClient(
        csms.CsmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.DeleteSecretEventRequest{}
    request.EventName = "{event_name}"
    response, err := client.DeleteSecretEvent(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
204	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码

状态码	描述
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.3.5.6 查询已触发的事件通知记录

功能介绍

查询三个月内所有已触发的事件通知记录。

接口约束

仅保存三个月以内的事件通知记录。

调用方法

请参见**如何调用API**。

URI

GET /v1/{project_id}/csms/notification-records

表 4-1439 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。

请求参数

表 4-1440 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

响应参数

状态码： 200

表 4-1441 响应 Body 参数

参数	参数类型	描述
records	Array of **Record** objects	Record对象。
page_info	**PageInfo** object	分页信息。

表 4-1442 Record

参数	参数类型	描述
event_name	String	凭据名称。
trigger_event_type	String	事件类型。 • SECRET_VERSION_CREATED:版本创建 • SECRET_VERSION_EXPIRED:版本过期 • SECRET_ROTATED:凭据轮转成功 • SECRET_DELETED:凭据删除 • SECRET_ROTATED_FAILED:凭据轮转失败
create_time	Long	事件通知记录的创建时间，时间戳，即从1970年1月1日至该时间的总秒数。
secret_name	String	凭据名称。

参数	参数类型	描述
secret_type	String	凭据类型 ● COMMON：通用凭据(默认)。用于应用系统中的各种敏感信息储存。 ● RDS：RDS凭据 。专门针对RDS的凭据，用于存储RDS的账号信息。（已不支持，使用RDS-FG替代） ● RDS-FG：RDS凭据 。专门针对RDS的凭据，用于存储RDS的账号信息。 ● GaussDB-FG：TaurusDB凭据。专门针对TaurusDB的凭据，用于存储TaurusDB的账号信息。
notification_target_name	String	事件通知的对象名称。
notification_target_id	String	事件通知的对象ID。
notification_content	String	事件通知的内容。
notification_status	String	事件通知状态。 ● SUCCESS：事件通知成功。 ● FAIL：事件通知失败。 ● INVALID：事件通知配置主题信息无效或不正确，无法触发通知。

表 4-1443 PageInfo

参数	参数类型	描述
next_marker	String	下一页查询地址（本页的末尾凭据名称，下一页起始凭据名称）。
previous_marker	String	本页的起始凭据名称，上一页末尾凭据名称。
current_count	Integer	本页返回条目数量。

状态码： 400

表 4-1444 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1445 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 401

表 4-1446 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1447 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 403

表 4-1448 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1449 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 404

表 4-1450 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1451 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 500

表 4-1452 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1453 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 502

表 4-1454 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1455 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

状态码： 504

表 4-1456 响应 Body 参数

参数	参数类型	描述
error	**ErrorDetail** object	错误信息返回体。

表 4-1457 ErrorDetail

参数	参数类型	描述
error_code	String	错误请求返回的错误码
error_msg	String	错误请求返回的错误信息

请求示例

无

响应示例

状态码： 200

请求已成功

```
{
  "records" : [ {
    "event_name" : "demo-event",
    "trigger_event_type" : "SECRET_VERSION_EXPIRED",
    "create_time" : 1581507580000,
    "secret_name" : "demo-secret",
    "secret_type" : "COMMON",
    "notification_target_name" : "SecertExpirationNotificationTest",
```

```
    "notification_target_id" : "urn:smn:cn-north-4:SecertExpirationTest",
    "notification_content" : "{\"eventName\":\"TestEvent20221112\",\"eventType
\":\"SECRET_VERSION_EXPIRED\",\"eventTime\":\"2023-04-14T20:25:10.126Z\",\"data\":{\"secretId
\":\"fde3d6ba-cb31-40b0-b6c4-78757050f8c8\",\"secretName\":\"Secret20230325\",\"createTime
\":\"2023-03-22T20:43:04.000Z\",\"updateTime\":\"2023-04-14T20:18:29.000Z\",\"versionId
\":\"v18\",\"versionExpireTime\":\"2023-03-29T17:07:23.000Z\"}}",
    "notification_status" : "SUCCESS"
  } ]
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.csms.v1.region.CsmsRegion;
import com.huaweicloud.sdk.csms.v1.*;
import com.huaweicloud.sdk.csms.v1.model.*;


public class ListNotificationRecordsSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running
this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CsmsClient client = CsmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(CsmsRegion.valueOf("<YOUR REGION>"))
                .build();
        ListNotificationRecordsRequest request = new ListNotificationRecordsRequest();
        try {
            ListNotificationRecordsResponse response = client.listNotificationRecords(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcsms.v1.region.csms_region import CsmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcsms.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CsmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CsmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ListNotificationRecordsRequest()
        response = client.list_notification_records(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    csms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := csms.NewCsmsClient(
        csms.CsmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
```

```
        Build())

    request := &model.ListNotificationRecordsRequest{}
    response, err := client.ListNotificationRecords(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.3.6 凭据轮转管理

4.3.6.1 查询任务列表

功能介绍

查询任务列表。

调用方法

请参见**如何调用API**。

URI

GET /v1/{project_id}/csms/tasks

表 4-1458 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。

表 4-1459 Query 参数

参数	是否必选	参数类型	描述
secret_name	否	String	凭据的名称。
status	否	String	任务状态。取值： ● SUCCESS： 任务轮转成功。 ● FAILED： 任务轮转失败。
task_id	否	String	任务ID。 该参数与其他参数不能同时存在。
limit	否	Integer	每页返回的个数。 默认值：50。
marker	否	String	分页查询起始的任务ID，为空时为查询第一页。

请求参数

无

响应参数

状态码： 200

表 4-1460 响应 Body 参数

参数	参数类型	描述
total	Integer	任务数量
tasks	Array of **SecretTask** objects	凭据任务列表。
next_marker	String	下一页查询地址（本页的末尾任务ID）。

表 4-1461 SecretTask

参数	参数类型	描述
task_id	String	任务ID
secret_name	String	凭据名称。
rotation_func_urn	String	FunctionGraph函数的urn。
task_status	String	任务状态。
operate_type	String	轮转类型。
task_time	Long	任务创建时间。
attempt_nums	Integer	轮转尝试次数。
task_error_code	String	任务错误码。
task_error_msg	String	任务错误信息。

状态码： 400

表 4-1462 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 401

表 4-1463 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 403

表 4-1464 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 404

表 4-1465 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 500

表 4-1466 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 502

表 4-1467 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 504

表 4-1468 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

请求示例

- 查询凭据任务，任务ID为xxxxx。

  ```
  /v1/xxxxx/csms/tasks?task_id=xxxxx
  ```

- 查询凭据任务，凭据名称为xx，任务状态为FAILED，每页返回个数为10个，分页起始任务ID为xxxx。

  ```
  /v1/xxxxx/csms/tasks?secret_name=xx&status=FAILED&limit=10&marker=xxxx
  ```

响应示例

状态码： 200

请求已成功。

```
{
  "total" : 1,
  "tasks" : [ {
    "task_id" : "xxxx",
    "secret_name" : "xxxx",
    "rotation_func_urn" : "urn:fss:cn-north-4:xxxxxxxx:function:default:xxxx:xxxxx",
    "task_status" : "SUCCESS",
    "attempt_nums" : 3,
    "operate_type" : "MULTI_USER",
    "task_time" : 1715436899000
  } ],
  "next_marker" : "xxxxxx"
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.csms.v1.region.CsmsRegion;
import com.huaweicloud.sdk.csms.v1.*;
import com.huaweicloud.sdk.csms.v1.model.*;


public class ListSecretTaskSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CsmsClient client = CsmsClient.newBuilder()
```

```
            .withCredential(auth)
            .withRegion(CsmsRegion.valueOf("<YOUR REGION>"))
            .build();
        ListSecretTaskRequest request = new ListSecretTaskRequest();
        try {
            ListSecretTaskResponse response = client.listSecretTask(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcsms.v1.region.csms_region import CsmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcsms.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CsmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CsmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ListSecretTaskRequest()
        response = client.list_secret_task(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    csms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/region"
)
```

```
func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := csms.NewCsmsClient(
        csms.CsmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ListSecretTaskRequest{}
    response, err := client.ListSecretTask(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功。
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.3.6.2 创建服务委托

功能介绍

创建服务委托。用于创建凭据管理服务相关委托和函数工作流相关委托。

调用方法

请参见**如何调用API**。

URI

POST /v1/csms/agencies

请求参数

表 4-1469 请求 Body 参数

参数	是否必选	参数类型	描述
secret_type	是	String	凭据类型。

响应参数

状态码： 200

表 4-1470 响应 Body 参数

参数	参数类型	描述
agencies	Array of **Agency** objects	委托详情列表。

表 4-1471 Agency

参数	参数类型	描述
agency_name	String	委托名称。
agency_id	String	委托ID。
error_msg	String	异常信息。当委托创建失败时，返回的异常信息。

状态码： 400

表 4-1472 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 401

表 4-1473 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 403

表 4-1474 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 404

表 4-1475 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 500

表 4-1476 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 502

表 4-1477 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 504

表 4-1478 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

请求示例

创建委托，创建凭据类型为RDS-FG的委托。

```
{
  "secret_type" : "RDS-FG"
}
```

响应示例

状态码： 200

请求已成功。

```
{
  "agencies" : [ {
    "agency_name" : "xxxx1",
    "agency_id" : "xxxxxxx"
  }, {
    "agency_name" : "xxxxx2",
    "error_msg" : "xxxxxxxx"
  } ]
}
```

SDK 代码示例

SDK代码示例如下。

Java

创建委托，创建凭据类型为RDS-FG的委托。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
```

```
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.csms.v1.region.CsmsRegion;
import com.huaweicloud.sdk.csms.v1.*;
import com.huaweicloud.sdk.csms.v1.model.*;


public class CreateAgencySolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
                .withAk(ak)
                .withSk(sk);

        CsmsClient client = CsmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(CsmsRegion.valueOf("<YOUR REGION>"))
                .build();
        CreateAgencyRequest request = new CreateAgencyRequest();
        CreateAgencyRequestBody body = new CreateAgencyRequestBody();
        body.withSecretType(CreateAgencyRequestBody.SecretTypeEnum.fromValue("RDS-FG"));
        request.withBody(body);
        try {
            CreateAgencyResponse response = client.createAgency(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

创建委托，创建凭据类型为RDS-FG的委托。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcsms.v1.region.csms_region import CsmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcsms.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
```

```
    credentials = BasicCredentials(ak, sk)

    client = CsmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CsmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = CreateAgencyRequest()
        request.body = CreateAgencyRequestBody(
            secret_type="RDS-FG"
        )
        response = client.create_agency(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

创建委托，创建凭据类型为RDS-FG的委托。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    csms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()

    client := csms.NewCsmsClient(
        csms.CsmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.CreateAgencyRequest{}
    request.Body = &model.CreateAgencyRequestBody{
        SecretType: model.GetCreateAgencyRequestBodySecretTypeEnum().RDS_FG,
    }
    response, err := client.CreateAgency(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功。
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.3.6.3 查看是否有服务委托

功能介绍

查看是否有服务委托

调用方法

请参见**如何调用API**。

URI

GET /v1/csms/agencies

表 4-1479 Query 参数

参数	是否必选	参数类型	描述
secret_type	是	String	凭据类型。

请求参数

无

响应参数

状态码： 200

表 4-1480 响应 Body 参数

参数	参数类型	描述
agency_granted	String	委托是否存在。

状态码： 400

表 4-1481 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 401

表 4-1482 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 403

表 4-1483 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 404

表 4-1484 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 500

表 4-1485 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 502

表 4-1486 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 504

表 4-1487 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

请求示例

查询委托，查询凭据类型为RDS-FG的委托。

```
/v1/csms/agencies?secret_type=RDS-FG
```

响应示例

状态码： 200

请求已成功。

```
{
  "agency_granted" : "true"
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.csms.v1.region.CsmsRegion;
import com.huaweicloud.sdk.csms.v1.*;
import com.huaweicloud.sdk.csms.v1.model.*;


public class ShowAgencySolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
                .withAk(ak)
                .withSk(sk);

        CsmsClient client = CsmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(CsmsRegion.valueOf("<YOUR REGION>"))
                .build();
        ShowAgencyRequest request = new ShowAgencyRequest();
        try {
            ShowAgencyResponse response = client.showAgency(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcsms.v1.region.csms_region import CsmsRegion
from huaweicloudsdkcore.exceptions import exceptions
```

```
from huaweicloudsdkcsms.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]

    credentials = BasicCredentials(ak, sk)

    client = CsmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CsmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ShowAgencyRequest()
        response = client.show_agency(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    csms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()

    client := csms.NewCsmsClient(
        csms.CsmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ShowAgencyRequest{}
    response, err := client.ShowAgency(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功。
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

4.3.6.4 获取凭据轮转函数模板

功能介绍

获取凭据轮转函数模板。

调用方法

请参见**如何调用API**。

URI

GET /v1/csms/function-templates

表 4-1488 Query 参数

参数	是否必选	参数类型	描述
secret_type	是	String	凭据类型。

参数	是否必选	参数类型	描述
secret_sub_type	是	String	凭据轮转账号类型。 ● SingleUser：单用户模式轮转 ● MultiUser：双用户模式轮转
engine	否	String	数据库类型。凭据类型为RDS-FG时为必填参数，可传入mysql、postgresql、sqlserver。其余凭据类型不支持。

请求参数

无

响应参数

状态码： 200

表 4-1489 响应 Body 参数

参数	参数类型	描述
function_templates	String	凭据轮转函数模板。

状态码： 400

表 4-1490 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 401

表 4-1491 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 403

表 4-1492 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 404

表 4-1493 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 500

表 4-1494 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 502

表 4-1495 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码
error_msg	String	错误描述

状态码： 504

表 4-1496 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码

参数	参数类型	描述
error_msg	String	错误描述

请求示例

获取凭据轮转模板，获取凭据类型为RDS-FG，凭据轮转账号类型为SingleUser和数据库类型为mysql的凭据轮转模板。

```
/v1/csms/function-templates?secret_type=RDS-FG&secret_sub_type=SingleUser&engine=mysql
```

响应示例

状态码： 200

请求已成功。

```
{
  "function_templates" : "xxxxxx"
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.csms.v1.region.CsmsRegion;
import com.huaweicloud.sdk.csms.v1.*;
import com.huaweicloud.sdk.csms.v1.model.*;


public class ShowSecretFunctionTemplatesSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
                .withAk(ak)
                .withSk(sk);

        CsmsClient client = CsmsClient.newBuilder()
                .withCredential(auth)
                .withRegion(CsmsRegion.valueOf("<YOUR REGION>"))
                .build();
        ShowSecretFunctionTemplatesRequest request = new ShowSecretFunctionTemplatesRequest();
        try {
            ShowSecretFunctionTemplatesResponse response = client.showSecretFunctionTemplates(request);
            System.out.println(response.toString());
```

```
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcsms.v1.region.csms_region import CsmsRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcsms.v1 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]

    credentials = BasicCredentials(ak, sk)

    client = CsmsClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CsmsRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ShowSecretFunctionTemplatesRequest()
        response = client.show_secret_function_templates(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    csms "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/csms/v1/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
```

```
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()

    client := csms.NewCsmsClient(
        csms.CsmsClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ShowSecretFunctionTemplatesRequest{}
    response, err := client.ShowSecretFunctionTemplates(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	请求已成功。
400	请求参数有误
401	被请求的页面需要用户名和密码
403	认证失败
404	资源不存在，资源未找到
500	服务内部错误
502	请求未完成。服务器从上游服务器收到一个无效的响应
504	网关超时

错误码

请参见**错误码**。

5 历史 API

5.1 管理 SSH 密钥对(V2.1)

5.1.1 查询 SSH 密钥对列表(V2.1)

功能介绍

查询SSH密钥对列表。

URI

- URI格式
  GET /v2.1/{project_id}/os-keypairs
- 参数说明

  表 5-1 参数说明

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。

请求消息

无

响应消息

表 5-2 响应参数

参数	是否必选	参数类型	描述
keypairs	是	Array of objects	SSH密钥对信息列表，详情请参见表5-3。

表 5-3 keypairs 字段数据结构说明

参数	是否必选	参数类型	描述
keypair	是	Object	SSH密钥对信息详情，详情请参见**表 5-4**。

表 5-4 keypair 字段数据结构说明

参数	是否必选	参数类型	描述
fingerprint	是	String	SSH密钥对应指纹信息。
name	是	String	SSH密钥对的名称。
public_key	是	String	SSH密钥对对应的publicKey信息。
is_key_protection	是	Boolean	SSH密钥对是否私钥托管与保护。

示例

- 请求样例

  无

- 响应样例

```
{
 "keypairs": [{
   "keypair": {
    "fingerprint": "15:b0:f8:b3:f9:48:63:71:cf:7b:5b:38:6d:44:2d:4a",
    "name": "keypair-601a2305-4f25-41ed-89c6-2a966fc8027a",
    "public_key": "ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAAAgQC+Eo/
RZRngaGTkFs7I62ZjsIlO79KklKbMXi8F+KITD4bVQHHn+kV
+4gRgkgCRbdoDqoGfpaDFs877DYX9n4z6FrAIZ4PES8TNKhatifpn9NdQYWA+IkU8CuvlEKGuFpKRi/
k7JLos/gHi2hy7QUwgtRvcefvD/vgQZOVw/mGR9Q== Generated-by-Nova\n",
    "is_key_protection": true
   }
 }
 ]
}
```

  或

```
{
    "error_code": "KPS.XXXX",
    "error_msg": "XXXX"
}
```

状态码

请参考**状态码**。

5.1.2 查询 SSH 密钥对详情(V2.1)

功能介绍

查询SSH密钥对详细信息。

URI

- URI格式
  GET /v2.1/{project_id}/os-keypairs/{keypair_name}
- 参数说明

表 5-5 参数说明

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。
keypair_name	是	String	SSH密钥对名称。

请求消息

无

响应消息

表 5-6 响应参数

参数	是否必选	参数类型	描述
keypair	是	Object	SSH密钥对信息，详情请参见表5-7。

表 5-7 keypair 字段数据结构说明

参数	是否必选	参数类型	描述
public_key	是	String	SSH密钥对对应的publicKey信息。
name	是	String	SSH密钥对的名称。
fingerprint	是	String	SSH密钥对对应的指纹信息。
created_at	是	String	SSH密钥对创建的时间。时间戳，即从1970年1月1日至该时间的总秒数。
deleted	是	Boolean	SSH密钥对删除的标记。
deleted_at	是	String	SSH密钥对删除的时间。时间戳，即从1970年1月1日至该时间的总秒数。

参数	是否必选	参数类型	描述
id	是	String	SSH密钥对的ID。
updated_at	是	String	SSH密钥对的更新时间。时间戳，即从1970年1月1日至该时间的总秒数。
user_id	是	String	SSH密钥对所属的用户信息。
is_key_protection	是	Boolean	SSH密钥对是否私钥托管与保护。
description	是	String	SSH密钥对的描述信息

示例

- 请求样例

  无

- 响应样例

```
{
    "keypair": {
        "created_at": "2014-05-07T12:06:13.681238",
        "deleted": false,
        "deleted_at": null,
        "fingerprint": "9d:00:f4:d7:26:6e:52:06:4c:c1:d3:1d:fd:06:66:01",
        "id": 1,
        "name": "keypair-3582d8b7-e588-4aad-b7f7-f4e76f0e4314",
        "public_key": "ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDYJrTVpcMwFqQy/
oMvtUSRofZdSRHEwrsX8AYkRvn2ZnCXM+b6+GZ2NQuuWj+ocznlnwiGFQDsL/yeE+/
kurqcPJFKKp60mToXIMyzioFxW88fJtwEWawHKAclbHWpR1t4fQ4DS+/sIbX/
Yd9btlVQ2tpQjodGDbM9Tr9/+/3i6rcR+EoLqmbgCgAiGiVV6VbM2Zx79yUwd
+GnQejHX8BlYZoOjCnt3NREsITcmWE9FVFy6TnLmahs3FkEO/
QGgWGkaohAJlsgaVvSWGgDn2AujKYwyDokK3dXyeX3m2Vmc3ejiqPa/C4nRrCOlko5nSgV/
9IXRx1ERImsqZnE9usB Generated-by-Nova\n",
        "updated_at": null,
        "user_id": "fake",
        "is_key_protection": true,
        "description": "keypair test"
    }
}
```

  或

```
{
    "error_code": "KPS.XXXX",
    "error_msg": "XXXX"
}
```

状态码

请参考**状态码**。

5.1.3 创建及导入 SSH 密钥对(V2.1)

功能介绍

创建SSH密钥对和导入SSH密钥对，同时可选择对私钥进行托管。

URI

- URI格式

  POST /v2.1/{project_id}/os-keypairs

- 参数说明

  表 5-8 参数说明

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。

请求消息

说明

创建SSH密钥对时，只需要提交SSH密钥对的name属性。导入SSH密钥对时，才需要提交public_key属性。

表 5-9 请求参数

参数	是否必选	参数类型	描述
keypair	是	Object	创建或导入的SSH密钥对的信息，详情请参见表5-10。

表 5-10 keypair 字段数据结构说明

参数	是否必选	参数类型	描述
public_key	否	String	导入公钥的字符串信息。
name	是	String	SSH密钥对的名称。 新创建的密钥对名称不能和已有密钥对的名称相同。 SSH密钥对名称由英文字母、数字、下划线、中划线组成，长度不能超过64个字节。
user_id	否	String	SSH密钥对的用户ID。
key_protection	否	Object	SSH密钥对私钥托管与保护，详情请参见表5-11。

表 5-11 key_protection 字段数据结构说明

参数	是否必选	参数类型	描述
private_key	否	String	导入私钥的字符串信息。

参数	是否必选	参数类型	描述
encryption	是	Object	对私钥进行加密存储的方式，详情请参见表5-12。

表 5-12 encryption 字段数据结构说明

参数	是否必选	参数类型	描述
type	是	String	取值范围：“kms”或“default”。 ● “default”为默认加密方式，适用于没有kms服务的局点。 ● “kms”为采用kms服务加密方式。 若局点没有kms服务，请填“default”。
kms_key_name	否	String	kms密钥的名称。 若“type”为“kms”，则必须填入kms服务密钥名称。

响应消息

表 5-13 响应参数

参数	是否必选	参数类型	描述
keypair	是	Object	SSH密钥对的信息，详情请参见表5-14。

表 5-14 keypair 字段数据结构说明

参数	是否必选	参数类型	描述
fingerprint	是	String	SSH密钥对对应的指纹信息。
name	是	String	SSH密钥对的名称。
public_key	是	String	SSH密钥对对应的publicKey信息。
private_key	是	String	SSH密钥对对应的privateKey信息。 ● 创建SSH密钥对时，响应中包括private_key的信息。 ● 导入SSH密钥对时，响应中不包括private_key的信息。
user_id	是	String	SSH密钥对所属的用户ID。

示例

- 创建SSH密钥对
  - 创建SSH密钥对请求样例

```
{
   "keypair": {
      "name": "demo1"
   }
}
```

  - 创建SSH密钥对响应样例

```
{
   "keypair": {
      "public_key": "ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQCXKzohKbFOqubYNunFNsrEYlk9NEIJIFvbmTe/
LTeMzFIPKM53Zu2sYr/uuNcziPkWpFchXdkD+O/
Bf2ZzKaR1DYPMWss9TkaqU4RQ7CIBW7ChJF1Qzc1JPRBmKe6e8qs1QBBoS1QvXgSjbuf2Fb1yncSb
phmQV8+5KA8xkxz4XdM1/gSAZZ14rJrMjgp7jCdgxWiHNcDuKxaPt+0eO8rEG/gxR7J0b9Uk53ao/
xjLoKXYdLLiYUaha0fHdW3t6Lw1NdzUMmKnlLqN9O37Tbg7vM0nN4UJt0XXvM45KfnJiMx0HUKXd
WkUj9cE8VBDPw/gBbQzSpJHgQFG7mNDZubN Generated-by-Nova\n",
      "private_key": "-----BEGIN RSA PRIVATE KEY-----
\nMIIEowIBAAKCAQEAlys6ISmxTqrm2DbpxTbKxGJZPTRCCSBb25k3vy03jMxSDyjO\nd2btrGK/
7rjXM4j5FqRXIV3ZA/jvwX9mcymkdQ2DzFrLPU5GqlOEUOwiAVuwoSRd
\nUM3NST0QZinunvKrNUAQaEtUL14Eo27n9hW9cp3Em6YZkFfPuSgPMZMc+F3TNf4E
\ngGWdeKyazI4Ke4wnYMVohzXA7isWj7ftHjvKxBv4MUeydG/VJOd2qP8Yy6Cl2HSy
\n4mFGoWtHx3Vt7ei8NTXc1DJip5S6jfTt+024O7zNJzeFCbdF17zOOSn5yYjMdB1C\nl3VpFI/
XBPFQQz8P4AW0M0qSR4EBRu5jQ2bmzQIDAQABAoIBAFwm1s3Gi7bICec+
\nOm2lhCAJUBs2UxcbKcQJd+lL2DQ1jcR9/LbfAP34SMZu2Ykp86Zw0kid3CBGzWko
\nj7yeYwmUDocxxfl+USedt+hYujYXvenNsDEE9CK0Xd3ZrAQrLGFOx3G8kfo6FxvG\nDRN/
lzhaorK7o5PRY+icWf6/joZ8Q96scHmm0ob5rtBkUYcek+ckf3mLVIpzzdKA
\ndkSi57M78zwDA89MpVABEoO1DPVxEqrrMQZy5UnAmeGHh16mPS4qMCokPVz36pSG
\nIWSqHnVKzsbxvw5Da9y69NmpSi2E1wqDaU9IzwnLyQpHnE1nXsWmxNqKTHlDBbnb
\nXPGFdcECgYEAxf4IMqYBeBiq
+7RVwcTcT4gpApJmywigwMFaaX35E3O53ja8hk1/\n5OCRnvK7yrt9wnWY8DIh8GPJptKzuTb/l/
14L4kE1MYm7Gpho5SwXV5BqtjgjfZm\nQVNpWruXEugXALcfbHiH+peO
+3AmwgIqgkOLPLxY1Duw6/miDB8bOfECgYEAw3VO\nl9edXExJvJvSzAopSNmw
+ExpUZTgS3L2Pyn21QhfwNHyxPH8fNNKv0/x9ZzBn25U
\nUxXTPPbLFV3cq7kfuYFW0OZkh8QjCPDkIE116E2QvacxqkBuW774xr5msfWdxpcp
\nwccgWKci1vEHLtqj3RTNMFKcXtj4QrCES4ZsPp0CgYEAhzYYux4LWszd188r0Yxz
\nOA0wlIUOlhFqVri02d4hv1sEz/Bphv5eHRwP1pDGFok8NRTCQSa7bsN27uptuksI\n
+e8rqKrWFcjMxB9SVrgwSVMZXWeG0uw2oN4p0MDTRyIgs2hbmWQm2Mev6Z6JmWyJ
\nQzSFXhDySN4K0NxGAiksGsECgYAJvZTXGFWtPdgG4DUXGgKIsOCS3yv4dtTerbH7\n39l/
MFWlRFE242BT0CvPcOp79P3pNhRZt6K5TQs921md7THZisqKypCD+5BLZ8XW\nnnkWb
+BGYgfaFp4RYaiH3tZFkmPrt5KaeE5BXGq0vzP8wpJC5+cln+RX13BYzLJzL
\nLr3COQKBgDJjY0qop3aSG+1tSkMJtOFdb2+qXiqaE+Wxv03SOkPx8LBz+taNz+DE
\nR7THDNXSrS85WVc3ozE8hULBX12iIWpr5kb5lRaw3ZgX6wBnSSpdQu49v3nFoGJk\nUWY95pQ/
BWQaX0q4KxzRVxup+7gwT5sKFXU+ktFtsGMYoDoSzbZS\n-----END RSA PRIVATE KEY-----\n",
      "user_id": "e4f380899b1248918f3d37098dc63746",
      "name": "demo123",
      "fingerprint": "49:ef:73:2b:9b:7f:2e:0c:58:d3:e3:42:8e:28:04:3b"
   }
}
```

    或

```
{
   "error_code": "KPS.XXXX",
   "error_msg": "XXXX"
}
```

- 创建SSH密钥对并托管私钥
  - 创建SSH密钥对并托管私钥请求样例

```
{
   "keypair": {
      "name": "demo2",
      "key_protection": {
         "encryption": {
            "type": "kms",
            "kms_key_name": "demo"
         }
```

```
    }
  }
}
```

- 创建SSH密钥对并托管私钥响应样例

```
{
  "keypair": {
    "public_key": "ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQCXKzohKbFOqubYNunFNsrEYlk9NEIJIFvbmTe/
LTeMzFIPKM53Zu2sYr/uuNcziPkWpFchXdkD+O/
Bf2ZzKaR1DYPMWss9TkaqU4RQ7CIBW7ChJF1Qzc1JPRBmKe6e8qs1QBBoS1QvXgSjbuf2Fb1yncSb
phmQV8+5KA8xkxz4XdM1/gSAZZ14rJrMjgp7jCdgxWiHNcDuKxaPt+0eO8rEG/gxR7J0b9Uk53ao/
xjLoKXYdLLiYUaha0fHdW3t6Lw1NdzUMmKnlLqN9O37Tbg7vM0nN4UJt0XXvM45KfnJiMx0HUKXd
WkUj9cE8VBDPw/gBbQzSpJHgQFG7mNDZubN Generated-by-Nova\n",
    "private_key": "-----BEGIN RSA PRIVATE KEY-----
\nMIIEowIBAAKCAQEAlys6ISmxTqrm2DbpxTbKxGJZPTRCCSBb25k3vy03jMxSDyjO\nd2btrGK/
7rjXM4j5FqRXIV3ZA/jvwX9mcymkdQ2DzFrLPU5GqlOEUOwiAVuwoSRd
\nUM3NST0QZinunvKrNUAQaEtUL14Eo27n9hW9cp3Em6YZkFfPuSgPMZMc+F3TNf4E
\ngGWdeKyazI4Ke4wnYMVohzXA7isWj7ftHjvKxBv4MUeydG/VJOd2qP8Yy6Cl2HSy
\n4mFGoWtHx3Vt7ei8NTXc1DJip5S6jfTt+024O7zNJzeFCbdF17zOOSn5yYjMdB1C\nl3VpFI/
XBPFQQz8P4AW0M0qSR4EBRu5jQ2bmzQIDAQABAoIBAFwm1s3Gi7bICec+
\nOm2lhCAJUBs2UxcbKcQJd+lL2DQ1jcR9/LbfAP34SMZu2Ykp86Zw0kid3CBGzWko
\nj7yeYwmUDocxxfl+USedt+hYujYXvenNsDEE9CK0Xd3ZrAQrLGFOx3G8kfo6FxvG\nDRN/
lzhaorK7o5PRY+icWf6/joZ8Q96scHmm0ob5rtBkUYcek+ckf3mLVIpzzdKA
\ndkSi57M78zwDA89MpVABEoO1DPVxEqrrMQZy5UnAmeGHh16mPS4qMCokPVz36pSG
\nIWSqHnVKzsbxvw5Da9y69NmpSi2E1wqDaU9lzwnLyQpHnE1nXsWmxNqKTHlDBbnb
\nXPGFdcECgYEAxf4IMqYBeBiq
+7RVwcTcT4gpApJmywigwMFaaX35E3O53ja8hk1/\n5OCRnvK7yrt9wnWY8DIh8GPJptKzuTb/l/
14L4kE1MYm7Gpho5SwXV5BqtjgjfZm\nQVNpWruXEugXALcfbHiH+peO
+3AmwgIqgkOLPLxY1Duw6/miDB8bOfECgYEAw3VO\nl9edXExJvJvSzAopSNmw
+ExpUZTgS3L2Pyn21QhfwNHyxPH8fNNKv0/x9ZzBn25U
\nUxXTPPbLFV3cq7kfuYFW0OZkh8QjCPDkIE116E2QvacxqkBuW774xr5msfWdxpcp
\nwccgWKci1vEHLtqj3RTNMFKcXtj4QrCES4ZsPp0CgYEAhzYYux4LWszd188r0Yxz
\nOA0wlIUOlhFqVri02d4hv1sEz/Bphv5eHRwP1pDGFok8NRTCQSa7bsN27uptuksl\n
+e8rqKrWFcjMxB9SVrgwSVMZXWeG0uw2oN4p0MDTRyIgs2hbmWQm2Mev6Z6JmWyJ
\nQzSFXhDySN4K0NxGAiksGsECgYAJvZTXGFWtPdgG4DUXGgKIsOCS3yv4dtTerbH7\n39l/
MFWlRFE242BT0CvPcOp79P3pNhRZt6K5TQs921md7THZisqKypCD+5BLZ8XW\nnkWb
+BGYgfaFp4RYaiH3tZFkmPrt5KaeE5BXGq0vzP8wpJC5+cln+RX13BYzLJzL
\nLr3COQKBgDJjY0qop3aSG+1tSkMJtOFdb2+qXiqaE+Wxv03SOkPx8LBz+taNz+DE
\nR7THDNXSrS85WVc3ozE8hULBX12iIWpr5kb5lRaw3ZgX6wBnSSpdQu49v3nFoGJk\nUWY95pQ/
BWQaX0q4KxzRVxup+7gwT5sKFXU+ktFtsGMYoDoSzbZS\n-----END RSA PRIVATE KEY-----\n",
    "user_id": "e4f380899b1248918f3d37098dc63746",
    "name": "demo123",
    "fingerprint": "49:ef:73:2b:9b:7f:2e:0c:58:d3:e3:42:8e:28:04:3b"
  }
}
```

- 导入SSH密钥对公钥
  - 导入SSH密钥对公钥请求样例

```
{
  "keypair": {
    "public_key": "ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQCyNtFZM04PFxERvZU5OBKTKr3mtRZABe5/+zX81lTgDF
CBfq6OXia47M4qXOa3ciBEKKZF+fMfs8U2UNB9aK1R/
uORsoEFtxSgZnWG6p4Ct1vnrqwDD934VaDFPEn+h3JeAfvTB
+Ag1YQ9zh9uYyE9Z3qZcC9+Ui93BDGdBtQeav4odxdwXcr2mT2jJV0nsocV0O4UcKM8BaIm8eqbcro
ZEkyxqT3mUoSbmGx1hrngjBsP1ufgwJ6D85LFGQC1SjIOLvsR9i6v41BaLF8/
kygvKOh2HlNVSMx38g52sTqoQ/xb3f8vR1VDXliAuD0frrG2Fy5wK4rOAnjuX9nh0bC9 Generated-
by-Nova\n",
    "name": "demo3"
  }
}
```

  - 导入SSH密钥对公钥响应样例

```
{
  "keypair": {
    "public_key": "ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQCyNtFZM04PFxERvZU5OBKTKr3mtRZABe5/+zX81lTgDF
CBfq6OXia47M4qXOa3ciBEKKZF+fMfs8U2UNB9aK1R/
uORsoEFtxSgZnWG6p4Ct1vnrqwDD934VaDFPEn+h3JeAfvTB
```


+Ag1YQ9zh9uYyE9Z3qZcC9+Ui93BDGdBtQeav4odxdwXcr2mT2jJV0nsocV0O4UcKM8BaIm8eqbcro
ZEkyxqT3mUoSbmGx1hrngjBsP1ufgwJ6D85LFGQC1SjIOLvsR9i6v41BaLF8/
kygvKOh2HlNVSMx38g52sTqoQ/xb3f8vR1VDXliAuD0frrG2Fy5wK4rOAnjuX9nh0bC9 Generated-
by-Nova\n",
        "user_id": "e4f380899b1248918f3d37098dc63746",
        "name": "demo1",
        "fingerprint": "b4:9a:c3:12:c4:90:bf:8e:7a:e2:70:10:c3:00:55:3f"
    }
}

或

{
    "error_code": "KPS.XXXX",
    "error_msg": "XXXX"
}

- 导入SSH密钥对公钥并托管私钥
  - 导入SSH密钥对公钥并托管私钥请求样例

{
    "keypair": {
        "public_key": "ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDb26UrW0htRbE/
Ygf8EPhzanBCc+5yEhkgmeSb2hTe48YRE5FdJKH6tueyj+vw5guoKjAITLjqZCqffGYXz/
7aXpFt244b9tTzh2l43uNtEZC+XZtc6KiBgfWupFI8O2i9YjJqdadsr+4Ad4AtIBbF++qsSJN4YycPX//
Gl8ja6AGPy4sdv8DZ40Gr8d+dMQ4pAsnUEtZ3jJ6NLdQU2CE1JhBdg3hbVbeh44gqQtSjhxWaSTlr
+NbVxSERtXXpsQWsid6qM1RhqH2+02cqXq5oNs4JLdu56pcTgSO5azTsGYJi6j5qp5BAdjMrFtHjbaeV
VWtkO1XQxfpueCJ470lx Generated-by-Nova\n",
        "name": "demo4",
        "key_protection": {
            "private_key": "-----BEGIN RSA PRIVATE KEY-----
\nMIIEpQIBAAKCAQEA29ulK1tIbUWxP2IH/BD4c2pwQnPuchIZIJnkm9oU3uPGEROR\nXSSh
+rbnso/r8OYLqCowCEy46mQqn3xmF8/+2l6RbduOG/bU84dpeN7jbRGQvl2b
\nXOiogYH1rqRSPDtovWIyanWnbK/uAHeALSAWxfvqrEiTeGMnD1//xpfI2ugBj8uL\nHb/A2eNBq/
HfnTEOKQLJ1BLWd4yejS3UFNghNSYQXYN4W1W3oeOIKkLUo4cVmkk5\na/
jW1cUhEbV16bEFrIneqjNUYah9vtNnKl6uaDbOCS3bueqXE4EjuWs07BmCYuo+
\naqeQQHYzKxbR422nlVVrZDtV0MX6bngieO9JcQIDAQABAoIBAAVSEXM1KFGMqDdy
\ndeMBviF85+6Tw6d7DKSfVMr4whyKwpZTNOdeJZVdq8nEdd9Eke+l5bets6PofKeT
\naR0WaYJ7W2WfNJc0p/6kvkawjixrimcw+LuM3dcUgA+T5nGStnwuzi2JX13f/BCC
\n09VDu4lbCVjWAMufCqjyl8wEjFXP0Amhu8fpDvqHuhGvDkoVWRm9vDEeyz71P25K\n/
UUs7kXw5Qv0VRcm15b+2jO6tii3RTo+JaTvkYXoI/qrOjiQhQD88geiOPQVuffa
\nzJhDw4/2GdHaCwEN6mzwKCYCfcPTRbM503F1YlceiP9w2qScToao
+5B2okN9cIE8\nTV4vmIkCgYEA+vo/TKqFep2D6DPY3dNRu2bHIYikBMtYMCIKJ1bgQ1xS/FjEJfSj
\notdDcBEik+0VEV05BCHNduTiMt6rTUD9fpqDduV8PfsskAAZZjndbyUEGP/KrCvS\nJsd2BFa2G7In/
3wz3zw+3P77Aegb+zHJfDyqYWReNkYy5tSsaZ9Oz38CgYEA4EH5\nIXQPhJt9683JmK/
INiyhW6WqKOAZkvjKpUpdUpGdVHJI/9dfUYI9wxnybgAkOZVL\noErxMTdRCDev8nVAq/OwC/
4jI5YnBGkN3ZRxIkCrepzwxIXgWeJiwWqsvdDbosfM
\nN0Q6PvUnPTXEly8RGcH9ABTQvQ9Nq4rQyjQvXw8CgYEAuK8hmWb55iq3AE32zfVM\n9ZxB
+Jk2KRkBghnqYtx5Fth/cJZZcJy/NXs2cucJDNWvZSG2bSX6rfzrvwdAAw9J\n
+rn0968TaADG4AhSqHj4S2tvwn2oRF3SvRqV68drJqJI8KYS/bi1gaZYSyTkQkp2\nu
+dgcV6MPWAW4OmrHeZO9j0CgYEAkY8Az4/vipkKkIWP9oUSJOevDDdpTQK4VscZ
\ncVPlYvSU8/0CGN2IRvWMdRhgXLnGyYF4BuDd0J4hAH50u6ETiwivGfmogS6ywJAX\nqdzx2dOz
+e/n7wceafkhNH2zBbmM9glNKgok7DxfbcF6is7IALoDJ4xbOHu4ZEHD
\n55sbrE0CgYEAuOuiISgfbujENFFPW0nvUmNqbkAH5YW1oUIWYA+64z7wcWyvzxRS
\nYml2XLWyrJy3JNzHpLoe4mCBxz+HGrftZ0/qfQ/WDZrY/Djp7/xIkPyI9EwsRTYC
\nr1PtWvVws3y3hgdo6WVQMaeUqtLSiTugyuuPqidH+/QtwxObunNH6Ns=\n-----END RSA
PRIVATE KEY-----\n",
            "encryption": {
                "type": "kms",
                "kms_key_name": "testName"
            }
        }
    }
}

  - 导入SSH密钥对公钥并托管私钥响应样例

{
    "keypair": {
        "public_key": "ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQCyNtFZM04PFxERvZU5OBKTKr3mtRZABe5/+zX81lTgDF
CBfq6OXia47M4qXOa3ciBEKKZF+fMfs8U2UNB9aK1R/
uORsoEFtxSgZnWG6p4Ct1vnrqwDD934VaDFPEn+h3JeAfvTB


```
+Ag1YQ9zh9uYyE9Z3qZcC9+Ui93BDGdBtQeav4odxdwXcr2mT2jJV0nsocV0O4UcKM8BaIm8eqbcro
ZEkyxqT3mUoSbmGx1hrngjBsP1ufgwJ6D85LFGQC1SjIOLvsR9i6v41BaLF8/
kygvKOh2HlNVSMx38g52sTqoQ/xb3f8vR1VDXliAuD0frrG2Fy5wK4rOAnjuX9nh0bC9 Generated-
by-Nova\n",
        "user_id": "e4f380899b1248918f3d37098dc63746",
        "name": "demo1",
        "fingerprint": "b4:9a:c3:12:c4:90:bf:8e:7a:e2:70:10:c3:00:55:3f"
    }
}
```


或


```
{
    "error_code": "KPS.XXXX",
    "error_msg": "XXXX"
}
```


状态码

请参考**状态码**。

5.1.4 删除 SSH 密钥对(V2.1)

功能介绍

根据SSH密钥对的名称，删除指定SSH密钥对。

URI

- URI格式

  DELETE /v2.1/{project_id}/os-keypairs/{keypair_name}

- 参数说明

  表 5-15 参数说明

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。
keypair_name	是	String	SSH密钥对名称。

请求消息

无

响应消息

无

状态码

请参考**状态码**。

5.1.5 修改密钥对描述信息(V2.1)

功能介绍

根据SSH密钥对的名称，修改指定SSH密钥对的描述信息。

URI

- URI格式

  PUT /v2.1/{project_id}/os-keypairs/{keypair_name}

- 参数说明

  表 5-16 参数说明

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。
keypair_name	是	String	SSH密钥对名称。

请求消息

表 5-17 请求参数

参数	是否必选	参数类型	描述
description	是	String	SSH密钥对描述，取值0到255字符。

响应消息

无

示例

- 请求样例

```
{
    "keypair": {
        "description": "keypair test"
    }
}
```

- 响应样例

  无

状态码

请参考**状态码**。

5.2 管理 SSH 密钥对(V2)

5.2.1 查询 SSH 密钥对列表(V2)

功能介绍

查询SSH密钥对信息列表。

URI

- URI格式
  GET /v2/{project_id}/os-keypairs
- 参数说明

表 5-18 参数说明

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。

请求消息

无

响应消息

表 5-19 响应参数

参数	是否必选	参数类型	描述
keypairs	是	Array of objects	SSH密钥对信息列表，详情请参见**表 5-20**。

表 5-20 keypairs 字段数据结构说明

参数	是否必选	参数类型	描述
keypair	是	Object	SSH密钥对信息详情，详情请参见**表 5-21**。

表 5-21 keypair 字段数据结构说明

参数	是否必选	参数类型	描述
fingerprint	是	String	SSH密钥对应指纹信息。
name	是	String	SSH密钥对的名称。
public_key	是	String	SSH密钥对对应的publicKey信息。

示例

如下以查询SSH密钥列表为例。

- 请求样例

  无

- 响应样例

```
{
   "keypairs": [
      {
         "keypair": {
            "fingerprint": "15:b0:f8:b3:f9:48:63:71:cf:7b:5b:38:6d:44:2d:4a",
            "name": "keypair-601a2305-4f25-41ed-89c6-2a966fc8027a",
            "public_key": "ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAAAgQC+Eo/
RZRngaGTkFs7I62ZjsIlO79KklKbMXi8F+KITD4bVQHHn+kV
+4gRgkgCRbdoDqoGfpaDFs877DYX9n4z6FrAIZ4PES8TNKhatifpn9NdQYWA+IkU8CuvlEKGuFpKRi/
k7JLos/gHi2hy7QUwgtRvcefvD/vgQZOVw/mGR9Q== Generated-by-Nova\n"
         }
      }
   ]
}
```

状态码

请参考**状态码**。

5.2.2 查询 SSH 密钥对详情(V2)

功能介绍

根据SSH密钥对的名称查询指定SSH密钥对。

URI

- URI格式

  GET /v2/{project_id}/os-keypairs/{keypair_name}

- 参数说明

  表 5-22 参数说明

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。

参数	是否必选	参数类型	描述
keypair_name	是	String	SSH密钥对的名称信息。

请求消息

无

响应消息

表 5-23 响应参数

参数	是否必选	参数类型	描述
keypair	是	Object	SSH密钥对信息，详情请参见表5-24。

表 5-24 keypair 字段数据结构说明

参数	是否必选	参数类型	描述
public_key	是	String	SSH密钥对对应的publicKey信息。
name	是	String	SSH密钥对的名称。
fingerprint	是	String	SSH密钥对对应的指纹信息。
created_at	是	String	SSH密钥对创建的时间。时间戳，即从1970年1月1日至该时间的总秒数。
deleted	是	Boolean	SSH密钥对删除的标记。
deleted_at	是	String	SSH密钥对删除的时间。时间戳，即从1970年1月1日至该时间的总秒数。
id	是	String	SSH密钥对的ID。
updated_at	是	String	SSH密钥对的更新时间。时间戳，即从1970年1月1日至该时间的总秒数。
user_id	是	String	SSH密钥对所属的用户信息。

示例

如下以查询SSH密钥对详情为例。

- 请求样例

  无

- 响应样例

```
{
    "keypair": {
```

```
        "created_at": "2014-05-07T12:06:13.681238",
        "deleted": false,
        "deleted_at": null,
        "fingerprint": "9d:00:f4:d7:26:6e:52:06:4c:c1:d3:1d:fd:06:66:01",
        "id": 1,
        "name": "keypair-3582d8b7-e588-4aad-b7f7-f4e76f0e4314",
        "public_key": "ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDYJrTVpcMwFqQy/
oMvtUSRofZdSRHEwrsX8AYkRvn2ZnCXM+b6+GZ2NQuuWj+ocznlnwiGFQDsL/yeE+/
kurqcPJFKKp60mToXIMyzioFxW88fJtwEWawHKAclbHWpR1t4fQ4DS+/sIbX/
Yd9btlVQ2tpQjodGDbM9Tr9/+/3i6rcR+EoLqmbgCgAiGiVV6VbM2Zx79yUwd
+GnQejHX8BlYZoOjCnt3NREsITcmWE9FVFy6TnLmahs3FkEO/
QGgWGkaohAJlsgaVvSWGgDn2AujKYwyDokK3dXyeX3m2Vmc3ejiqPa/C4nRrCOlko5nSgV/
9IXRx1ERImsqZnE9usB Generated-by-Nova\n",
        "updated_at": null,
        "user_id": "fake"
    }
}
```


状态码

请参考**状态码**。

5.2.3 创建及导入 SSH 密钥对(V2)

功能介绍

创建SSH密钥对，或把公钥导入华为云中，生成SSH密钥对。

创建SSH密钥对成功后，请把响应数据中的私钥内容保存到本地文件，用户使用该私钥登录云服务器。为保证云服务器安全，私钥数据只能下载一次，请妥善保管。

URI

- URI格式

  POST /v2/{project_id}/os-keypairs

- 参数说明

  表 5-25 参数说明

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。

请求消息

说明

创建SSH密钥对时，只需要提交SSH密钥对的name属性。导入SSH密钥对时，才需要提交public_key属性。

表 5-26 请求参数

参数	是否必选	参数类型	描述
keypair	是	Object	创建或导入的SSH密钥对的信息，详情请参见表5-27。

表 5-27 keypair 字段数据结构说明

参数	是否必选	参数类型	描述
public_key	否	String	导入公钥的字符串信息。
type	否	String	SSH密钥对的类型，值为ssh或x509。
name	是	String	SSH密钥对的名称。 新创建的密钥对名称不能和已有密钥对的名称相同。 SSH密钥对名称由英文字母、数字、下划线、中划线组成，长度不能超过64个字节。
user_id	否	String	SSH密钥对的用户ID。

响应消息

表 5-28 响应参数

参数	是否必选	参数类型	描述
keypair	是	Object	SSH密钥对的信息，详情请参见表5-29。

表 5-29 keypair 字段数据结构说明

参数	是否必选	参数类型	描述
fingerprint	是	String	SSH密钥对对应的指纹信息。
name	是	String	SSH密钥对的名称。
public_key	是	String	SSH密钥对对应的publicKey信息。
private_key	否	String	SSH密钥对对应的privateKey信息。 • 创建SSH密钥对时，响应中包括private_key的信息。 • 导入SSH密钥对时，响应中不包括private_key的信息。

参数	是否必选	参数类型	描述
user_id	是	String	SSH密钥对所属的用户ID。

示例

- 创建SSH密钥对
  - 创建SSH密钥对请求样例

```
{    "keypair": {
        "type": "ssh",
        "name": "demo"
    }
}
```

  - 创建SSH密钥对响应样例

```
{
   "keypair": {
      "public_key": "ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQCyNtFZM04PFxERvZU5OBKTKr3mtRZABe5/+zX81lTgDF
CBfq6OXia47M4qXOa3ciBEKKZF+fMfs8U2UNB9aK1R/
uORsoEFtxSgZnWG6p4Ct1vnrqwDD934VaDFPEn+h3JeAfvTB
+Ag1YQ9zh9uYyE9Z3qZcC9+Ui93BDGdBtQeav4odxdwXcr2mT2jJV0nsocV0O4UcKM8BaIm8eqbcro
ZEkyxqT3mUoSbmGx1hrngjBsP1ufgwJ6D85LFGQC1SjIOLvsR9i6v41BaLF8/
kygvKOh2HlNVSMx38g52sTqoQ/xb3f8vR1VDXliAuD0frrG2Fy5wK4rOAnjuX9nh0bC9 Generated-
by-Nova\n",
      "private_key": "-----BEGIN RSA PRIVATE KEY-----
\nMIIEpAIBAAKCAQEAsjbRWTNODxcREb2VOTgSkyq95rUWQAXuf/s1/NZU4AxQgX6u
\njl4muOzOKlzmt3IgRCimRfnzH7PFNlDQfWitUf7jkbKBBbcUoGZ1huqeArdb566s\nAw/d
+FWgxTxJ/odyXgH70wfgINWEPc4fbmMhPWd6mXAvfIlvdwQxnQbUHmr+KHcX
\ncF3K9pk9oyVdJ7KHFdDuFHCjPAWiJvHqm3K6GRJMsak95lKEm5hsdYa54IwbD9bn\n4MCeg/
OSxRkAtUoyDi77EfYur+NQWixfP5MoLyjodh5TVUjMd/
IOdrE6qEP8W93/\nL0dVQ15YgLg9H66xthcucCuKzgJ47l/Z4dGwvQIDAQABAoIBAQCdTjXL/
rVQLJQs\njKNDNnNu47NsCTvyl0nGPf+Rhb61ZSlKpH9/uyuC38O7MPWVx28jup3J9q7btNrG
\n7t6ZU+RpFAvbdyzb1pamXsoupLmEvESrZEsBCOhtY2fdsTG/Md+Ji0a1J6Z2VQG9\nbEviLC4S/
VwCRDwnzHOJInKIoJZroZv6SdK+KonQBS0Rq9bZrlvtBUUhaSGjBcJx\nmWKO78ikNOXP/
5Yl92SAw2vOYWhZdMZQrkp1EUFMG18Akuj+jC9QKXXfsLYYfzsQ
\nIGgpRdf6zYIV84QVMZ7NhQABM5DNmQfxrSlUSdbvOzOJzmShp41tH3sn9d+XS+bS
\nLIoyuaQhAoGBAN7tpwgkcKddKI/Lp/CPqjkxP6lfO+xHEXjtnZd1Y//BavPSgq4v
\nWuFHgx1sPQK49KcSLZfF6UxkPw0KHBc5R9RkfYBAIdGNwENF2xyoYLLdnUtF4hRq
\n1q2DC3oklBZibH2tc6+hQ2aCWSeMvQbIvxTYV70EFzwR5f4O5LIskCm/AoGBAMyn
\nA7DOQdvcf4aexSYL4kGp70ERMOCtwr/d+O5RswARoyAQOxp4a7/TyFuGjnlT//bR
\nEYacXV1AieIdeJF3PgeUIR1QnUlNYD9Rufs14fs+5idQ7Evn1gvXv0HpBYTY7wNu
\nWTrWbsznY0fNIrGT4bQR6QpdvIuR5TBJf6HIAKyDAoGAFhKf3D2HbfraXkqC6V3A
\nNAN9Uy7bxwxOXZPha7Ky4QrspRGt4MNNk0q6X7ps3A0mJDi3jPSKoga2+3qJx37j
\nbtM4Xe97qb0IUWDkThUZ5fvtbBuSRAVEFlAIXeKrSwAZz+PRtY0ZGFhFrZXQzZAo
\n4058eXmjN05qYFpnKIEjEQ8CgYEAwELzW6oaAzR+dfk428p0UB4W0HkXAy0a9efS
\nUgpc8Oag6qF09SRGjdunshySQvegU78MCPtjVxUntE7dk0OD+di213SBn3jawAHG
\niHORjtkDndIPfCwcUdnpK0GAVtL6kK2dlIIZa9TB15WnT07Pzry4w21WkYSJ3Thf
\neJyNzYMCgYA8OvpKMdaEXFeNZWHDE1Q2VmpxvP/D6u6s4SBuyy8eac1qqku/s7zc\nsuFd/
o9wbBgzsf4eN8tNJ4bxrArRXvf9WyH7xd4PE3DvVJnz5S+8Nqj2Z0KCAqPD
\nibDbFxBYHcMIdwC2JBGQZIXpkST2jG9wZho5KghX4yiHSOPr2V25/g==\n-----END RSA PRIVATE
KEY-----\n",
      "user_id": "6fc0d2cbbfab40b199874b97097e913d",
      "name": "demo",
      "fingerprint": "b4:9a:c3:12:c4:90:bf:8e:7a:e2:70:10:c3:00:55:3f"
   }
}
```

- 导入SSH密钥对
  - 导入SSH密钥对请求样例

```
{
   "keypair": {
      "public_key": "ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQCyNtFZM04PFxERvZU5OBKTKr3mtRZABe5/+zX81lTgDF
CBfq6OXia47M4qXOa3ciBEKKZF+fMfs8U2UNB9aK1R/
```

```
uORsoEFtxSgZnWG6p4Ct1vnrqwDD934VaDFPEn+h3JeAfvTB
+Ag1YQ9zh9uYyE9Z3qZcC9+Ui93BDGdBtQeav4odxdwXcr2mT2jJV0nsocV0O4UcKM8BaIm8eqbcro
ZEkyxqT3mUoSbmGx1hrngjBsP1ufgwJ6D85LFGQC1SjIOLvsR9i6v41BaLF8/
kygvKOh2HlNVSMx38g52sTqoQ/xb3f8vR1VDXliAuD0frrG2Fy5wK4rOAnjuX9nh0bC9 Generated-
by-Nova\n",
        "type": "ssh",
        "name": "demo1",
        "user_id": "fake"
    }
}
```


- 导入SSH密钥对响应样例


```
{
    "keypair": {
        "public_key": "ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQCyNtFZM04PFxERvZU5OBKTKr3mtRZABe5/+zX81lTgDF
CBfq6OXia47M4qXOa3ciBEKKZF+fMfs8U2UNB9aK1R/
uORsoEFtxSgZnWG6p4Ct1vnrqwDD934VaDFPEn+h3JeAfvTB
+Ag1YQ9zh9uYyE9Z3qZcC9+Ui93BDGdBtQeav4odxdwXcr2mT2jJV0nsocV0O4UcKM8BaIm8eqbcro
ZEkyxqT3mUoSbmGx1hrngjBsP1ufgwJ6D85LFGQC1SjIOLvsR9i6v41BaLF8/
kygvKOh2HlNVSMx38g52sTqoQ/xb3f8vR1VDXliAuD0frrG2Fy5wK4rOAnjuX9nh0bC9 Generated-
by-Nova\n",
        "user_id": "6fc0d2cbbfab40b199874b97097e913d",
        "name": "demo1",
        "fingerprint": "b4:9a:c3:12:c4:90:bf:8e:7a:e2:70:10:c3:00:55:3f"
    }
}
```


状态码

请参考**状态码**。

5.2.4 删除 SSH 密钥对(V2)

功能介绍

根据SSH密钥对的名称，删除指定SSH密钥对。

URI

- URI格式

  DELETE /v2/{project_id}/os-keypairs/{keypair_name}

- 参数说明

  表 5-30 参数说明

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。
keypair_name	是	String	SSH密钥对的名称。

请求消息

无

响应消息

无

状态码

请参考**状态码**。

5.2.5 复制 SSH 密钥对(V2)

功能介绍

在同一个租户下可能包含多个用户账号，将同一租户下目标用户账号下的密钥对复制到当前用户账号下。

URI

- URI格式

  POST /v2/{project_id}/os-keypairs/copy

- 参数说明

  表 5-31 参数说明

参数	是否必选	参数类型	描述
project_id	是	String	项目ID。

请求消息

表 5-32 请求参数

参数	是否必选	参数类型	描述
user_name	是	String	同一租户下的目标用户名。
force	否	Boolean	是否强制覆盖已有密钥对。

响应消息

表 5-33 响应参数

参数	是否必选	参数类型	描述
changed	是	Integer	复制的密钥对数量。
success	是	Array of objects	复制成功的密钥对列表，详情请参见**表 5-34**。

参数	是否必选	参数类型	描述
failed	是	Array of objects	复制失败的密钥对列表，详情请参见**表 5-34**。

表 5-34 success/failed 字段数据结构说明

参数	是否必选	参数类型	描述
keypair	是	String	密钥对名称。
message	是	String	任务消息

示例

- 请求样例

```
{
 "user_name": "kpsuser"
}
```

- 响应样例

```
{
   "changed": 2,
   "success": [
      {
         "keypair": "KeyPair-test1",
         "message": "imported"
      },
      {
         "keypair": "KeyPair-test2",
         "message": "imported"
      }
   ],
   "failed": [
      {
         "keypair": "KeyPair-test3",
         "message": "exist"
      }
   ]
}
```

或

```
{
   "error_code": "KPS.XXXX",
   "error_msg": "XXXX"
}
```

状态码

请参考**状态码**。

6 应用示例

6.1 示例 1：加解密小量数据

场景描述

当您需**加解密不大于4KB的小量数据**（例如：口令、证书、电话号码等）时，您可以通过KMS界面使用在线工具加解密数据，或者调用KMS的API接口使用指定的用户主密钥直接加密、解密数据。

流程如下：

1. 用户需要在KMS中创建一个用户主密钥。
2. 用户调用KMS的“encrypt-data”接口，使用指定的用户主密钥将明文数据加密为密文数据。
3. 用户在服务器上部署密文证书。
4. 当服务器需要使用证书时，调用KMS的“decrypt-data”接口，将密文数据解密为密文证书。

加解密 API

加解密小数据时，涉及的API如下：

- **创建用户主密钥**：创建用户主密钥，用来加密数据。
- **加密数据**：用指定的主密钥加密数据密钥。
- **解密数据**：用指定的主密钥解密数据密钥。

操作步骤

步骤1 创建用户主密钥。

- 接口相关信息

  URI格式：POST /v1.0/{project_id}/kms/create-key

  详情请参见**创建密钥**。

> **说明**
>
> 别名“/default”为服务默认主密钥的后缀名，由服务自动创建。因此用户创建的主密钥别名不能与服务默认主密钥的别名相同，即后缀名不能为“/default”。

- 请求示例

  POST: https://*{endpoint}*/v1.0/53d1aefc533f4ce9a59c26b01667cbcf/kms/create-key

  {endpoint}信息请从**地区和终端节点**获取。

  Body：

```
{
   "key_alias": "test"
}
```

- 响应示例

```
{
   "key_info": {
      "key_id": "bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e",
      "domain_id": "b168fe00ff56492495a7d22974df2d0b"
   }
}
```

步骤2 加密数据。

- 接口相关信息

  URI格式：POST /v1.0/{project_id}/kms/encrypt-data

  详情请参见**加密数据**。

- 请求示例

  POST https://*{endpoint}*/v1.0/53d1aefc533f4ce9a59c26b01667cbcf/kms/encrypt-data

  {endpoint}信息请从**地区和终端节点**获取。

  您可使用**查询密钥列表**接口获取当前用户密钥列表，包括key_id等信息 。

  Body：

```
{
   "key_id": "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
   "plain_text": "12345678"
}
```

- 响应示例

```
{    "key_id": "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
    "cipher_text": "AgDoAG7EsEc2OHpQxz4gDFDH54CqwaelpTdEl
+RFPjbKn5klPTvOywYIeZX60kPbFsYOpXJwkL32HUM50MY22Eb1fOSpZK7WJpYjx66EWOkJvO
+Ey3r1dLdNAjrZrYzQlxRwNS05CaNKoX5rr3NoDnmv+UNobaiS25muLLiqOt6UrStaWow9AUyOHSzl
+BrX2Vu0whv74djK
+3COO6cXT2CBO6WajTJsOgYdxMfv24KWSKw0TqvHe8XDKASQGKdgfI74hzI1YWJlNjlmLWFlMTAtNDRjZ
C1iYzg3LTFiZGExZGUzYjdkNwAAAACdcfNpLXwDUPH3023MvZK8RPHe129k6VdNIi3zNb0eFQ==" 
}
```

步骤3 解密数据。

- 接口相关信息

  URI格式：POST /v1.0/{project_id}/kms/decrypt-data

  详情请参见**解密数据**。

- 请求示例

  POST https://*{endpoint}*/v1.0/53d1aefc533f4ce9a59c26b01667cbcf/kms/decrypt-data

  {endpoint}信息请从**地区和终端节点**获取。

您可使用**查询密钥列表**接口获取当前用户密钥列表，包括key_id等信息 。

Body：

```
{      "cipher_text": "AgDoAG7EsEc2OHpQxz4gDFDH54CqwaelpTdEl
+RFPjbKn5klPTvOywYIeZX60kPbFsYOpXJwkL32HUM50MY22Eb1fOSpZK7WJpYjx66EWOkJvO
+Ey3r1dLdNAjrZrYzQlxRwNS05CaNKoX5rr3NoDnmv+UNobaiS25muLLiqOt6UrStaWow9AUyOHSzl
+BrX2Vu0whv74djK
+3COO6cXT2CBO6WajTJsOgYdxMfv24KWSKw0TqvHe8XDKASQGKdgfI74hzI1YWJlNjlmLWFlMTAtNDRjZ
C1iYzg3LTFiZGExZGUzYjdkNwAAAACdcfNpLXwDUPH3023MvZK8RPHe129k6VdNIi3zNb0eFQ==" 
 }
```

- 响应示例

```
{
   "key_id": "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
    "plain_text": "12345678"
 }
```

----结束

6.2 示例 2：加解密大量数据

场景描述

在**大量数据加解密**的场景，您的程序会经常使用到对数据密钥的加解密。

- 大量数据加密的流程如下：
  a. 在KMS中创建一个用户主密钥。
  b. 调用KMS的“create-datakey”接口创建数据加密密钥。用户得到一个明文的数据加密密钥和一个密文的数据加密密钥。其中密文的数据加密密钥是由指定的用户主密钥加密明文的数据加密密钥生成的。
  c. 用户使用明文的数据加密密钥来加密明文文件，生成密文文件。
  d. 用户将密文的数据加密密钥和密文文件一同存储到持久化存储设备或服务中。
- 大量数据解密的流程如下：
  a. 用户从持久化存储设备或服务中读取密文的数据加密密钥和密文文件。
  b. 用户调用KMS的“decrypt-datakey”接口，使用对应的用户主密钥（即生成密文的数据加密密钥时所使用的用户主密钥）来解密密文的数据加密密钥，取得明文的数据加密密钥。

     若对应的用户主密钥被误删除，会导致解密失败。因此，需要妥善管理好用户主密钥。
  c. 用户使用明文的数据加密密钥来解密密文文件。

涉及接口

加解密数据密钥时，需要进行用户主密钥创建、数据密钥创建、加密数据密钥、解密数据密钥等操作，涉及的接口如下：

- **创建用户主密钥**：创建用户主密钥，用来加密数据密钥。
- **创建数据密钥**：创建数据密钥。
- **加密数据密钥**：用指定的主密钥加密数据密钥。
- **解密数据密钥**：用指定的主密钥解密数据密钥。

操作步骤

步骤1 创建用户主密钥。

- 接口相关信息

  URI格式：POST /v1.0/{project_id}/kms/create-key

  详情请参见**创建密钥**。

  说明

  别名“/default”为服务默认主密钥的后缀名，由服务自动创建。因此用户创建的主密钥别名不能与服务默认主密钥的别名相同，即后缀名不能为“/default”。

- 请求示例

  POST: https://*{endpoint}*/v1.0/53d1aefc533f4ce9a59c26b01667cbcf/kms/create-key

  {endpoint}信息请从**地区和终端节点**获取。

  Body：

```
{
   "key_alias": "test"
}
```

- 响应示例

```
{
   "key_info": {
      "key_id": "bb6a3d22-dc93-47ac-b5bd-88df7ad35f1e",
      "domain_id": "b168fe00ff56492495a7d22974df2d0b"
   }
}
```

步骤2 创建数据密钥。

- 接口相关信息

  URI格式：POST /v1.0/{project_id}/kms/create-datakey

  详情请参见**创建数据密钥**。

- 请求示例

  POST https://*{endpoint}*/v1.0/53d1aefc533f4ce9a59c26b01667cbcf/kms/create-datakey

  {endpoint}信息请从**地区和终端节点**获取。

  您可使用**查询密钥列表**接口获取当前用户密钥列表，包括key_id等信息 。

  Body：

```
{
   "key_id": "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
   "datakey_length": "512"
}
```

- 响应示例

```
{
   "key_id": "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
   "plain_text":
"8151014275E426C72EE7D44267EF11590DCE0089E19863BA8CC832187B156A72A5A17F17B5EF0D525872C59ECEB72948AF85E18427F8BE0D46545C979306C08D",
   "cipher_text":
"020098009EEAFCE122CAA5927D2E020086F9548BA1675FDB022E4ECC01B96F2189CF4B85E78357E73E1CEB518DAF7A4960E7C7DE8885ED3FB2F1471ABF400119CC1B20BD3C4A9B80AF590EFD0AEDABFDBB0E2B689DA7B6C9E7D3C5645FCD9274802586BE63779471F9156F2CDF07CD8412FFBE9230643034363662302D653732372D346439632D623335642D663834626234373461333766000000000045B05321483BD9F9561865EE7DFE9BE267A42EB104E98C16589CE46940B18E52"
}
```

步骤3 加密数据密钥。

- 接口相关信息

  URI格式：POST /v1.0/{project_id}/kms/encrypt-datakey

  详情请参见加密数据密钥。

- 请求示例

  POST https://*{endpoint}*/v1.0/53d1aefc533f4ce9a59c26b01667cbcf/kms/encrypt-datakey

  {endpoint}信息请从地区和终端节点获取。

  您可使用查询密钥列表接口获取当前用户密钥列表，包括key_id等信息 。

  Body：


```
{
    "key_id": "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
    "plain_text": 
"00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000F5A5FD42D16A20302798EF6ED309979B43003D2320D9F0E8EA9831A92759FB4B",
    "datakey_plain_length": "64"
}
```


- 响应示例


```
{
    "key_id": "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
    "cipher_text": 
"020098005273E14E6E8E95F5463BECDC27E80AF820B9FC086CB47861899149F67CF07DAFF2810B7D27BDF19AB7632488E0926A48DB2FC85BEA905119411B46244C5E6B8036C60A0B0B4842FFE6994518E89C19B1C1D688D9043BCD6053EA7BA0652642CE59F2543C80669139F4F71ABB9BD9A24330643034363662302D653732372D346439632D623335642D663834626234373461333766000000000D34457984F9730D57F228C210FD22CA6017913964B21D4ECE45D81092BB9112E",
    "datakey_length": "64"
}
```


步骤4 解密数据密钥。

- 接口相关信息

  URI格式：POST /v1.0/{project_id}/kms/decrypt-datakey

  详情请参见解密数据密钥。

- 请求示例

  POST https://*{endpoint}*/v1.0/53d1aefc533f4ce9a59c26b01667cbcf/kms/decrypt-datakey

  {endpoint}信息请从地区和终端节点获取。

  您可使用查询密钥列表接口获取当前用户密钥列表，包括key_id等信息 。

  Body：


```
{
    "key_id": "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
    "datakey_cipher_length": "64",
    "cipher_text": 
"020098005273E14E6E8E95F5463BECDC27E80AF820B9FC086CB47861899149F67CF07DAFF2810B7D27BDF19AB7632488E0926A48DB2FC85BEA905119411B46244C5E6B8036C60A0B0B4842FFE6994518E89C19B1C1D688D9043BCD6053EA7BA0652642CE59F2543C80669139F4F71ABB9BD9A24330643034363662302D653732372D346439632D623335642D663834626234373461333766000000000D34457984F9730D57F228C210FD22CA6017913964B21D4ECE45D81092BB9112E"
}
```


- 响应示例


```
{
    "data_key": 
"000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000",
```

```
  "datakey_length": "64",
  "datakey_dgst": "F5A5FD42D16A20302798EF6ED309979B43003D2320D9F0E8EA9831A92759FB4B"
}
```

----结束

6.3 示例 3：查询密钥相关信息

场景描述

本章节指导您通过KMS接口查询密钥列表、密钥信息、密钥实例以及密钥标签。

涉及接口

- **查询密钥列表**：查询本账号下所有的密钥列表。
- **查询密钥信息**：查询指定密钥的详细信息。
- **查询密钥实例**：通过标签过滤，查询指定用户主密钥的详细信息。
- **查询密钥标签**：查询指定密钥的标签信息。标签管理服务需要使用该接口查询指定密钥的全部标签数据。

操作步骤

步骤1 查询密钥列表。

- 接口相关信息

  URI格式：POST /v1.0/{project_id}/kms/list-keys

  详情请参见“查询密钥列表”。

- 请求示例

  POST: https://*{endpoint}*/v1.0/53d1aefc533f4ce9a59c26b01667cbcf/kms/list-keys

  {endpoint}信息请从**地区和终端节点**获取。

  Body：

```
{
  "limit": "2",
  "marker": "1"
}
```

- 响应示例

```
{
   "keys": [
       "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
      "2e258389-bb1e-4568-a1d5-e1f50adf70ea"
    ],
    "key_details": [
    {
     "key_id":"0d0466b0-e727-4d9c-b35d-f84bb474a37f",
      "domain_id":"00074811d5c27c4f8d48bb91e4a1dcfd",
       "key_alias":"caseuirpr",
       "realm":"aaaa",
       "key_description":"123",
      "creation_date":"1502799822000",
       "scheduled_deletion_date":"",
       "key_state":"2",
       "default_key_flag":"0",
       "key_type":"1",
       "expiration_time":"1501578672000",
```

```
        "origin":"kms"
},
      {
     "key_id":"2e258389-bb1e-4568-a1d5-e1f50adf70ea",
      "domain_id":"00074811d5c27c4f8d48bb91e4a1dcfd",
      "key_alias":"casehvniz",
       "realm":"aaaa",
     "key_description":"234",
       "creation_date":"1502799820000",
        "scheduled_deletion_date":"",
        "key_state":"2",
        "default_key_flag":"0",
        "key_type":"1",
       "expiration_time":"1501578673000",
        "origin":"kms"
}
    ],
    "next_marker": "",
   "truncated": "false",
    "total":2
}
```


步骤2 查询密钥信息。

- 接口相关信息

  URI格式：POST /v1.0/{project_id}/kms/describe-key

  详情请参见“查询密钥信息”。

- 请求示例

  POST: https://*{endpoint}*/v1.0/53d1aefc533f4ce9a59c26b01667cbcf/kms/describe-key

  {endpoint}信息请从**地区和终端节点**获取。

  您可使用**查询密钥列表**接口获取当前用户密钥列表，包括key_id等信息 。

  Body：

```
{
  "key_id": "0d0466b0-e727-4d9c-b35d-f84bb474a37f"
}
```

- 响应示例


```
{
   "key_info": {
      "key_id": "0d0466b0-e727-4d9c-b35d-f84bb474a37f",
      "domain_id": "b168fe00ff56492495a7d22974df2d0b",
      "key_alias": "kms_test",
      "realm": "aaa",
       "key_description": "",
     "creation_date": "1472442386000",
      "scheduled_deletion_date": "",
      "key_state": "2",
      "default_key_flag": "0",
      "key_type": "1",
      "expiration_time":"1501578672000",
      "origin":"kms",
      "key_rotation_enabled":"false",
      "sys_enterprise_project_id ": "0",
    }
}
```


步骤3 查询密钥实例。

- 接口相关信息

  URI格式：POST /v1.0/{project_id}/kms/resource_instances/action

  详情请参见“查询密钥实例”。

- 请求示例

  POST: https://*{endpoint}*/v1.0/53d1aefc533f4ce9a59c26b01667cbcf/kms//resource_instances/action

  {endpoint}信息请从**地区和终端节点**获取。

  Body：

```
{
      "offset": "100",
      "limit": "100",
      "action": "filter",
       "matches":[
    {
          "key": "resource_name",
            "value": "resource1"
       }
      ],
      "tags": [
         {
         "key": "key1",
           "values": [
                 "value1",
                   "value2"
           ]
         }
    ]
}
```

- 响应示例

```
{
 "resources" : [ {
  "resource_id" : "90c03e67-5534-4ed0-acfa-89780e47a535",
  "resource_detail" : [ {
    "key_id" : "90c03e67-5534-4ed0-acfa-89780e47a535",
    "domain_id" : "4B688Fb77412Aee5570E7ecdbeB5afdc",
    "key_alias" : "tagTest_xmdmi",
    "key_description" : "123",
    "creation_date" : 1521449277000,
    "scheduled_deletion_date" : "",
    "key_state" : 2,
    "default_key_flag" : 0,
    "key_type" : 1,
    "key_rotation_enabled" : false,
    "expiration_time" : "",
    "origin" : "kms",
    "sys_enterprise_project_id" : "0",
    "realm" : "cn-north-7"
  } ],
  "resource_name" : "tagTest_xmdmi",
  "tags" : [ {
    "key" : "key",
    "value" : "testValue!"
  }, {
    "key" : "haha",
    "value" : "testValue"
  } ]
 } ],
 "total_count" : 1
}
```

步骤4 查询密钥标签。

- 接口相关信息

  URI格式：GET /v1.0/{project_id}/kms/{key_id}/tags

  详情请参见“查询密钥标签”。

- 请求示例

GET: https://*{endpoint}*/v1.0/53d1aefc533f4ce9a59c26b01667cbcf/kms/94752282-805e-4032-ada8-34966f70e02f/tags

{endpoint}信息请从**地区和终端节点**获取。

Body：

无

- 响应示例

```
{
       "tags": [
        {
         "key": "key1",
         "value": "value1"
          },
          {
          "key": "key2",
          "value": "value3"
          }
          ],
          "existTagsNum":2
}
```

----结束

7 权限和授权项

7.1 权限及授权项说明

如果您需要对您所拥有的数据加密服务（Data Encryption Workshop，DEW）进行精细的权限管理，您可以使用统一身份认证服务（Identity and Access Management，IAM），如果华为账号已经能满足您的要求，不需要创建独立的IAM用户，您可以跳过本章节，不影响您使用DEW的其它功能。

默认情况下，新建的IAM用户没有任何权限，您需要将其加入用户组，并给用户组授予策略或角色，才能使用户组中的用户获得相应的权限，这一过程称为授权。授权后，用户就可以基于已有权限对云服务进行操作。

权限根据授权的精细程度，分为**角色**和**策略**。角色以服务为粒度，是IAM最初提供的一种根据用户的工作职能定义权限的粗粒度授权机制。策略以API接口为粒度进行权限拆分，授权更加精细，可以精确到某个操作、资源和条件，能够满足企业对权限最小化的安全管控要求。

说明

如果您要允许或是禁止某个接口的操作权限，请使用策略。

账号具备所有接口的调用权限，如果使用账号下的IAM用户发起API请求时，该IAM用户必须具备调用该接口所需的权限，否则，API请求将调用失败。每个接口所需要的权限，与各个接口所对应的授权项相对应，只有发起请求的用户被授予授权项所对应的策略，该用户才能成功调用该接口。例如，用户要调用接口来查询账号密钥对的SSH密钥列表，那么这个IAM用户被授予的策略中必须包含允许“kps:domainKeypairs:list”的授权项，该接口才能调用成功。

支持的授权项

策略包含系统策略和自定义策略，如果系统策略不满足授权要求，管理员可以创建自定义策略，并通过给用户组授予自定义策略来进行精细的访问控制。策略支持的操作与API相对应，授权项列表说明如下：

- 权限：允许或拒绝某项操作。
- 对应API接口：自定义策略实际调用的API接口。
- 授权项：自定义策略中支持的Action，在自定义策略中的Action中写入授权项，可以实现授权项对应的权限功能。

- 依赖的授权项：部分Action存在对其他Action的依赖，需要将依赖的Action同时写入授权项，才能实现对应的权限功能。
- IAM项目(Project)/企业项目(Enterprise Project)：自定义策略的授权范围，包括IAM项目与企业项目。授权范围如果同时支持IAM项目和企业项目，表示此授权项对应的自定义策略，可以在IAM和企业管理两个服务中给用户组授权并生效。如果仅支持IAM项目，不支持企业项目，表示仅能在IAM中给用户组授权并生效，如果在企业管理中授权，则该自定义策略不生效。关于IAM项目与企业项目的区别，详情请参见：**IAM与企业管理的区别**。

数据加密服务（DEW）支持的自定义策略授权项如下所示：

- **加密密钥管理**，包含密钥管理对应的授权项，如创建密钥、查询密钥、创建授权等接口。
- **密钥对管理**，包含密钥对管理对应的授权项，如创建密钥对、查询密钥对、删除密钥对等接口。

7.2 加密密钥管理

权限	对应API接口	授权项（Action）	依赖的授权项	IAM项目(Project)	企业项目(Enterprise Project)
创建密钥	POST /v1.0/{project_id}/kms/create-key	kms:cmk:create	-	√	√
启用密钥	POST /v1.0/{project_id}/kms/enable-key	kms:cmk:enable	-	√	√
禁用密钥	POST /v1.0/{project_id}/kms/disable-key	kms:cmk:disable	-	√	√
计划删除密钥	POST /v1.0/{project_id}/kms/schedule-key-deletion	kms:cmk:update	-	√	√
取消计划删除密钥	POST /v1.0/{project_id}/kms/cancel-key-deletion	kms:cmk:update	-	√	√
查询密钥列表	POST /v1.0/{project_id}/kms/list-keys	kms:cmk:list	-	√	√
查询密钥信息	POST /v1.0/{project_id}/kms/describe-key	kms:cmk:get	-	√	√

权限	对应API接口	授权项（Action）	依赖的授权项	IAM项目(Project)	企业项目(Enterprise Project)
创建随机数	POST /v1.0/{project_id}/kms/gen-random	kms:cmk:generate	-	√	×
创建数据密钥	POST /v1.0/{project_id}/kms/create-datakey	kms:dek:create	-	√	√
创建不含明文数据密钥	POST /v1.0/{project_id}/kms/create-datakey-without-plaintext	kms:dek:create	-	√	√
加密数据密钥	POST /v1.0/{project_id}/kms/encrypt-datakey	kms:dek:crypto	-	√	√
解密数据密钥	POST /v1.0/{project_id}/kms/decrypt-datakey	kms:dek:crypto	-	√	√
查询实例数	GET /v1.0/{project_id}/kms/user-instances	kms:cmk:getInstance	-	√	×
查询配额	GET /v1.0/{project_id}/kms/user-quotas	kms:cmk:getQuota	-	√	×
修改密钥别名	POST /v1.0/{project_id}/kms/update-key-alias	kms:cmk:update	-	√	√
修改密钥描述	POST /v1.0/{project_id}/kms/update-key-description	kms:cmk:update	-	√	√
创建授权	POST /v1.0/{project_id}/kms/create-grant	kms:grant:create	-	√	√
撤销授权	POST /v1.0/{project_id}/kms/revoke-grant	kms:grant:revoke	-	√	√
退役授权	POST /v1.0/{project_id}/kms/retire-grant	kms:grant:retire	-	√	√
查询授权列表	POST /v1.0/{project_id}/kms/list-grants	kms:grant:list	-	√	×

权限	对应API接口	授权项（Action）	依赖的授权项	IAM项目(Project)	企业项目(Enterprise Project)
查询可退役授权列表	POST /v1.0/{project_id}/kms/list-retirable-grants	kms:grant:list	-	√	×
加密数据	POST /v1.0/{project_id}/kms/encrypt-data	kms:cmk:crypto	-	√	√
解密数据	POST /v1.0/{project_id}/kms/decrypt-data	kms:cmk:crypto	-	√	√
获取密钥导入参数	POST /v1.0/{project_id}/kms/get-parameters-for-import	kms:cmk:getMaterial	-	√	√
导入密钥材料	POST /v1.0/{project_id}/kms/import-key-material	kms:cmk:importMaterial	-	√	√
删除密钥材料	POST /v1.0/{project_id}/kms/delete-imported-key-material	kms:cmk:deleteMaterial	-	√	√
开启密钥轮换	POST /v1.0/{project_id}/kms/enable-key-rotation	kms:cmk:enableRotation	-	√	√
修改密钥轮换周期	POST /v1.0/{project_id}/kms/update-key-rotation-interval	kms:cmk:updateRotation	-	√	√
关闭密钥轮换	POST /v1.0/{project_id}/kms/disable-key-rotation	kms:cmk:disableRotation	-	√	√
查询密钥轮换状态	POST /v1.0/{project_id}/kms/get-key-rotation-status	kms:cmk:getRotation	-	√	√
查询密钥实例	POST /v1.0/{project_id}/kms/resource_instances/action	kms:cmkTag:listInstance	-	√	√
查询密钥标签	GET /v1.0/{project_id}/kms/{key_id}/tags	kms:cmkTag:list	-	√	√
查询项目标签	GET /v1.0/{project_id}/kms/tags	kms:cmkTag:list	-	√	×

权限	对应API接口	授权项（Action）	依赖的授权项	IAM项目(Project)	企业项目(Enterprise Project)
批量添加删除密钥标签	POST /v1.0/{project_id}/kms/{key_id}/tags/action	kms:cmkTag:batch	-	√	√
添加密钥标签	POST /v1.0/{project_id}/kms/{key_id}/tags	kms:cmkTag:create	-	√	√
删除密钥标签	POST /v1.0/{project_id}/kms/{key_id}/tags/{key}	kms:cmkTag:delete	-	√	√
创建专属密钥库	/v1.0/{project_id}/keystores/	kms:keystore:create	-	√	×
删除专属密钥库	/v1.0/{project_id}/keystores/{keystore_id}	kms:keystore:delete	-	√	×
启用专属密钥库	/v1.0/{project_id}/keystores/{keystore_id}/enable	kms:keystore:enable	-	√	×
禁用专属密钥库	/v1.0/{project_id}/keystores/{keystore_id}/disable	kms:keystore:disable	-	√	×
查询专属密钥库列表	/v1.0/{project_id}/keystores/	kms:keystore:list	-	√	×
获取专属密钥库	/v1.0/{project_id}/keystores/{keystore_id}	kms:keystore:get	-	√	×

7.3 密钥对管理

权限	对应API接口	授权项（Action）	依赖的授权项	IAM项目(Project)	企业项目(Enterprise Project)
创建和导入SSH密钥对（OpenStack原生）	POST /v2.1/{project_id}/os-keypairs	ecs:serverKeypairs:create	-	√	x
查询SSH密钥对详情（OpenStack原生）	GET /v2.1/{project_id}/os-keypairs/{keypair_name}	ecs:serverKeypairs:get	-	√	x
查询SSH密钥对列表（OpenStack原生）	GET /v2.1/{project_id}/os-keypairs	ecs:serverKeypairs:list	-	√	x
删除SSH密钥对（OpenStack原生）	DELETE /v2.1/{project_id}/os-keypairs/{keypair_name}	ecs:serverKeypairs:delete	-	√	x

A 附录

A.1 状态码

状态码	编码	状态说明
200	OK	请求已成功。
202	Accept	任务提交成功，当前系统繁忙，下发的任务会延迟处理。
204	No Content	请求已成功，无内容返回。
300	multiple choices	被请求的资源存在多个可供选择的响应。
400	Bad Request	请求参数有误。
401	Unauthorized	被请求的页面需要用户名和密码。
403	Forbidden	服务器已经理解请求，但是拒绝执行它。
404	Not Found	资源不存在，资源未找到。
405	Method Not Allowed	请求中指定的方法不被允许。
406	Not Acceptable	服务器生成的响应无法被客户端所接受。
407	Proxy Authentication Required	用户必须首先使用代理服务器进行验证，这样请求才会被处理。
408	Request Timeout	请求超出了服务器的等待时间。
409	Conflict	由于冲突，请求无法被完成。
500	Internal Server Error	服务内部错误。
501	Not Implemented	请求未完成。服务器不支持所请求的功能。
502	Bad Gateway	请求未完成。服务器从上游服务器收到一个无效的响应。
503	Service Unavailable	请求未完成。系统暂时异常。

状态码	编码	状态说明
504	Gateway Timeout	网关超时。

A.2 错误码

更多服务错误码请参见**API错误中心**。

状态码	错误码	错误信息	描述	处理措施
400	CSMS.0002	The value of parameter is invalid.	请求XX参数错误	输入合法参数
400	CSMS.0003	the request is invalud.	请求非法	传递正确URL
400	CSMS.0004	The requested body format is wrong.	请求Body体格式错误	输入合法body体
400	CSMS.0005	The resource does not exist.	资源不存在	输入正确的凭据信息
400	CSMS.0103	The secret is in the scheduled deletion state.	凭据处于“计划删除”状态，不能使用.	确认凭据处于enable或disable状态
400	CSMS.0105	Can not delete the system internal stage.	禁止删除系统内置的版本状态	禁止删除内置版本
400	CSMS.0106	The secret name not found int the db.	数据库中不存在凭据名称	输入系统中存在的凭据的名称
400	CSMS.0107	The secret is in schedule delete state.	凭据已处于“计划删除"状态.	凭据已处于计划删除状态，输入其余凭据
400	CSMS.0108	The secret is not in schedule delete state.	凭据未处于“计划删除”状态.	凭据未处于计划删除状态，输入其他凭据
400	CSMS.0202	The number of secret has reached the upper limit.	凭据数量达到配额上限.	删除其余凭据

状态码	错误码	错误信息	描述	处理措施
400	CSMS.0203	The number of stage has reached the upper limit.	凭据的版本状态数量达到配额上限.	删除其余版本状态
400	CSMS.0204	The number of stage is greater than the quota limit.	凭据的版本状态数量超出配额上限.	删除其余版本状态
400	CSMS.0301	Invalid X-Auth-Token.	X-Auth-Token无效.	请重新获取token，并在使用时确保token字符串的完整性。
400	CSMS.0401	Can not get the protected secret value using the provided KMS key.	凭据值通过KMS服务加密解密失败.	请确保密钥存在且处于启用状态
403	CSMS.0109	The secret state is not enabled.	凭据对象未处于启用状态.	启用凭据后再进行其他操作
403	CSMS.0302	The user role has no permission to access the interface.	用户角色无权限访问接口.	请联系管理员给账户添加CSMS Fullacess权限。
404	CSMS.0205	Version quota not found for secret	凭据XX未发现相应的版本配额.	凭据已被删除，请选择其他凭据操作
404	CSMS.0206	Stage quota not found for secret	凭据XX未发现相应的版本状态配额.	凭据已被删除，请选择其他凭据操作
404	CSMS.0207	Secret quota not found.	未发现凭据配额信息.	凭据已被删除，请选择其他凭据操作
409	CSMS.0101	The secret name already exists.	凭据名称已存在.	输入其他合法的凭据名称
409	CSMS.0102	The version id already exists.	凭据版本号已存在.	输入其他合法的凭据版本号
409	CSMS.0104	The stage name already exists.	版本状态名称已存在.	输入其他合法的版本状态名称

状态码	错误码	错误信息	描述	处理措施
500	CSMS.0001	An internal error occurred.	服务内部错误	请重试或联系客服。
500	CSMS.0006	AES encrypt secret value occurred an error.	AES加密凭据值时发生错误.	请重试或联系客服。
500	CSMS.0201	The number of secret has reached the upper limit.	超出配额上限.	删除其余凭据
400	KMS.0105	A system exception occurred. Contact technical support.	发生系统异常。请联系技术支持。	请联系技术支持。
400	KMS.0106	It is replica service, readonly api allowed.	副本服务只允许只读操作。	副本服务只允许只读操作。
400	KMS.0201	Invalid request URL.	请求URL非法。	请传递正确的URL。
400	KMS.0202	Invalid JSON format of the request message.	请求消息JSON格式非法。	请传递正确的消息体。
400	KMS.0203	Request message too long.	请求消息长度超出限制。	请传递正确的消息体。
400	KMS.0204	Parameters missing in the request message.	请求消息缺少参数。	请传递正确的消息体。
400	KMS.0205	Invalid key ID.	密钥key_id非法。	请传递正确的密钥id。
400	KMS.0206	Invalid sequence number.	sequence序号非法。	请传递正确的序列号。
400	KMS.0208	Invalid value of value encryption_context.	encryption_context参数非法。	请检查encryption_context字段是否合法。

状态码	错误码	错误信息	描述	处理措施
400	KMS.0209	The key has been disabled.	密钥已被禁用，不能使用。	请启用该密钥。
400	KMS.0210	The key is in Scheduled deletion state and cannot be used.	密钥处于计划删除状态，不能使用。	请启用该密钥。
400	KMS.0211	Cannot perform this operation on Default Master Keys.	默认主密钥不支持该操作。	请使用普通主密钥操作该任务。
400	KMS.0212	Invalid resource type.	资源类型非法。	请使用正确的资源类型。
400	KMS.0214	The request format is invalid.	请求格式非法。	请使用正确的请求格式。
400	KMS.0308	Invalid parameter.	字段非法。	请传递正确的参数。
400	KMS.0309	External keys required.	密钥来源应为外部导入。	请使用外部导入密钥进行此操作。
400	KMS.0310	The key is not in Pending import state.	密钥未处于等待导入状态。	请确保密钥状态处于“等待导入”状态。
400	KMS.0311	Failed to decrypt data using the RSA private key.	RSA私钥解密数据失败。	请确保传入的密文的正确性，或联系技术支持。
400	KMS.0312	External keys cannot be rotated.	外部密钥不支持轮换操作。	请使用普通主密钥。
400	KMS.0313	Key rotation is not enabled.	密钥轮换未被启用。	请启用密钥轮换。
400	KMS.0315	Invalid partition_id.	分区类型非法。	输入正确的分区类型。
400	KMS.0317	The cmk partition is not enabled.	密钥分区类型未启用。	输入启用的分区类型。

状态码	错误码	错误信息	描述	处理措施
400	KMS.0318	Partition name has exist.	分区类型名称已存在。	输入正确的分区类型名称。
400	KMS.0319	Rotation not supported in the current KMS version.	当前版本密钥管理系统不支持密钥轮换操作。	请重试或联系客服。
400	KMS.0320	Resource frozen.	资源已冻结，请您尽快续费。	请续费后重试。
400	KMS.0323	Failed to obtain the partition of the key.	获取密钥所在分区信息失败。	请重试或联系客服。
400	KMS.0324	RSA keys cannot be rotated.	RSA密钥不支持轮换。	请使用普通主密钥。
400	KMS.0325	The asymmetric key is not support this operation.	非对称密钥不支持此操作。	请使用对称密钥重试。
400	KMS.0327	Failed to obtain user permissions.	获取用户操作权限失败。	请联系管理员给账户添加KMS CMKFullaccess权限。
400	KMS.0329	Hash algorithm does not match the digest length.	哈希算法与消息摘要长度不匹配。	请传递正确的参数，或者联系技术支持。
400	KMS.0331	The symmetric key is not support this operation.	对称密钥不支持此操作。	请使用正确的密钥类型重试。
400	KMS.0332	This key is not support the signing algorithm.	此密钥不支持签名算法。	请使用正确的签名算法，或者联系技术支持。

状态码	错误码	错误信息	描述	处理措施
400	KMS.0333	The signing algorithm SM2DSA_SM3 not support RAW signing.	签名算法SM2DSA_SM3不支持RAW签名。	请使用正确的签名算法，或者联系技术支持。
400	KMS.0334	The cmk is used to encrypt and decrypt.	该密钥用于加解密。	该密钥用于加解密。
400	KMS.0335	The cmk is used to sign and verify.	该密钥用于签名和验证。	该密钥用于签名和验证。
400	KMS.0336	The current region does not support SM4.	当前区域不支持SM4。	更换算法，或者联系技术支持。
400	KMS.0337	The current region does not support SM2.	当前区域不支持SM2。	更换算法，或者联系技术支持。
400	KMS.0338	The custom keystore do not support rotation.	自定义密钥库不支持轮转。	联系技术支持。
400	KMS.0339	The custom keystore do not support import key.	自定义密钥库不支持导入密钥。	联系技术支持。
400	KMS.0340	not support keystore.	当前不支持密钥库。	联系技术支持。
400	KMS.0341	The hmac key do not support this operation.	hmac密钥不支持该操作。	请传递正确的参数。
400	KMS.0401	Tag list cannot be empty.	标签列表不能为空。	请传递正确的参数。
400	KMS.0402	Invalid match value.	match中value字段不合法。	请传递正确的参数。
400	KMS.0403	Invalid match key.	match中key字段不合法。	请传递正确的参数。

状态码	错误码	错误信息	描述	处理措施
400	KMS.0404	Invalid action.	action字段不合法。	请传递正确的参数。
400	KMS.0405	Invalid tag value.	tag中value字段不合法。	请传递正确的参数。
400	KMS.0406	Invalid tag key.	tag中key字段不合法。	请传递正确的参数。
400	KMS.0407	Invalid tag list size.	tag列表长度不合法。	请传递正确的参数。
400	KMS.0408	Invalid resourceType.	resourceType字段不合法。	请传递正确的参数。
400	KMS.0409	Too many tags.	tag达到上限。	标签配额已达到上限，请删除部分标签后再重试。
400	KMS.0410	Invalid tag value length.	tag中value长度不合法。	请传递正确的参数。
400	KMS.0411	Invalid tag key length.	tag中key长度不合法。	请传递正确的参数。
400	KMS.0412	Invalid tag list.	tag list不合法。	请传递正确的参数。
400	KMS.0413	Too many tag values.	tag中values列表长度超过限制。	请传递正确的参数。
400	KMS.0414	Invalid tags.	tags字段不合法。	请传递正确的参数。
400	KMS.0415	Invalid matches.	matches字段不合法。	请传递正确的参数。
400	KMS.0417	Invalid offset.	offset不在有效数字范围内。	请传递正确的参数。
400	KMS.0418	Offset is not required.	不需要offset。	请传递正确的参数。
400	KMS.1101	Invalid key_alias.	key_alias密钥别名非法。	请传递正确的参数。
400	KMS.1102	Invalid realm.	realm密钥区域非法。	请传递正确的参数。
400	KMS.1103	Invalid key_description.	key_description密钥描述非法。	请传递正确的参数。

状态码	错误码	错误信息	描述	处理措施
400	KMS.1104	Duplicate key aliases.	密钥别名已经存在。	请更换别名。
400	KMS.1105	Too many keys.	密钥个数已达上限。	配额已达到上限，增加配额或者删除部分密钥。
400	KMS.1108	Failed to create the default partition for the key.	创建密钥所在默认分区失败。	请重试或联系客服。
400	KMS.1109	Failed to create the route for the key.	创建密钥路由信息失败。	请重试或联系客服。
400	KMS.1110	Invalid alg_type.	非法的算法类型。	请使用正确的算法类型。
400	KMS.1114	EC keys do not support to encrypt/ decrypt.	该密钥不支持加解密。	请使用正确的密钥去加解密。
400	KMS.1115	Symmetric keys do not support to sign/verify.	对称密钥不支持签名/验证。	请使用非对称密钥重试。
400	KMS.1201	The key is not disabled.	密钥未被禁用。	请先禁用密钥。
400	KMS.1301	The key is not enabled.	密钥未被启用。	请先启用密钥。
400	KMS.1401	Set the pending deletion period between 7 to 1096 days.	计划删除密钥时间范围：7天h至1096天。	请传递正确的参数。
400	KMS.1402	The key is already in Pending deletion state.	密钥已处于计划删除状态。	密钥已经处于“计划删除”状态，无需再操作。

状态码	错误码	错误信息	描述	处理措施
400	KMS.1404	This region is not the original region of the key, and it is not allowed to import-key-material.	此区域不是密钥的原始区域，不允许导入密钥材料。	当前局点不支持导入密钥材料。
400	KMS.1501	The key is not in Pending deletion state.	密钥未处于计划删除状态。	请先“计划删除”密钥。
400	KMS.1601	Invalid limit.	limit不在有效数字范围内。	请传递正确的参数。
400	KMS.1602	marker must be greater than or equals 0.	marker参数需大于等于0。	请传递正确的参数。
400	KMS.1603	Invalid offset.	offset不在有效数字范围内。	请传递正确的参数。
400	KMS.1801	random_data_length must be 512 bits.	random_data_length随机数长度需等于512位。	请传递正确的参数。
400	KMS.1802	random_data_length must be a multiple of 8.	random_data_length必须能被8整除，即必须为整字节数。	请传递正确的参数。
400	KMS.1901	datakey_length must be in the range 8 bits to 8,192 bits.	datakey_length必须介于8-8192比特之间。	请传递正确的参数。
400	KMS.1902	key_spec can only be AES_128 or AES_256.	key_spec必须为AES_128或者AES_256。	请传递正确的参数。
400	KMS.1903	datakey_length must be a multiple of 8.	datakey_length必须能被8整除，即必须为整字节数。	请传递正确的参数。

状态码	错误码	错误信息	描述	处理措施
400	KMS.1904	The rsa wrapping data key is over length 1520 bits.	rsa wrapping data key超过了1520字节。	请传递正确的参数。
400	KMS.2001	datakey_length must be 512 bits.	datakey_length数据密钥长度需等于512位。	请传递正确的参数。
400	KMS.2101	Invalid plain_text.	plain_text数据密钥明文非法。	请传递正确的参数。
400	KMS.2102	datakey_plain_length must be 64 bytes.	datakey_plain_length数据密钥明文长度需等于64字节。	请传递正确的参数。
400	KMS.2103	Failed to verify the DEK hash.	数据密钥hash校验失败。	请确认数据密钥是否合法或联系客服。
400	KMS.2104	The length of plain_text does not match datakey_plain_length.	plain_text实际长度与datakey_plain_length不匹配。	请传递正确的参数。
400	KMS.2105	The symmetric key not support this encryption algorithm.	symmetric key不支持该加密算法。	请传递正确的参数。
400	KMS.2106	The rsa key not support this encryption algorithm.	rsa key不支持该加密算法。	请传递正确的参数。
400	KMS.2107	The sm2 key not support this encryption algorithm.	sm2 key不支持该加密算法。	请传递正确的参数。

状态码	错误码	错误信息	描述	处理措施
400	KMS.2108	The rsa encryption plain_text is over length.	rsa encryption plain_text长度过长。	请传递正确的参数。
400	KMS.2201	Invalid cipher_text.	cipher_text数据密钥密文非法。	请传递正确的参数。
400	KMS.2202	datakey_cipher_length must be 64 bytes.	datakey_cipher_length数据密钥密文长度需等于64字节。	请传递正确的参数。
400	KMS.2203	Failed to verify the DEK hash.	数据密钥hash校验失败。	请确认数据密钥是否合法或联系客服。
400	KMS.2204	The length of cipher_text does not match datakey_cipher_length.	cipher_text实际长度与datakey_cipher_length不匹配。	请传递正确的参数。
400	KMS.2301	The quota value is beyond the maximum configurable limit.	配额值超出可配置的最大限制。	请传递正确的参数。
400	KMS.2302	New quota value must not less than old.	新配额值不得小于旧配额值。	请传递正确的参数。
400	KMS.2303	The quota type is not supported.	配额类型不支持。	请传递正确的参数。
400	KMS.2304	Can not update grant quota when no grant has created.	未创建授权时无法更新授权配额。	创建授权后重试。
400	KMS.2305	Can not update cmk quota when no cmk has created.	未创建cmk时无法更新cmk配额。	创建密钥后重试。

状态码	错误码	错误信息	描述	处理措施
400	KMS.2401	Specify an operation in addition to create-grant.	操作不能只包含create-grant。	请传递正确的参数。
400	KMS.2402	Invalid user ID.	授权/退役主体非法。	请传递正确的参数。
400	KMS.2403	Failed to create the grant.	创建授权失败。	请重试或联系客服。
400	KMS.2404	Too many CMK grants.	用户主密钥授权超过上限。	配额已达到上限，增加配额或者删除部分授权。
400	KMS.2405	Too many grants.	主体授权超过上限。	配额已达到上限，增加配额或者删除部分授权。
400	KMS.2406	The basic partition has no right to create grant.	基本分区没有创建授权的权限。	联系技术支持。
400	KMS.2501	Invalid grant ID.	grant不存在。	请传递存在的授权ID。
400	KMS.2502	grant_id and key_id do not match.	grant_id与key_id不匹配。	请确保grant_id与key_id匹配。
400	KMS.2601	Token expired.	令牌已失效。	请重新获取令牌。
400	KMS.2602	Key expiration time must be later than the current time.	导入密钥失效时间必须大于当前时间。	请重新选择导入密钥失效时间。
400	KMS.2603	Key IDs in the imported key and token do not match.	导入密钥key_id与令牌中key_id不匹配。	请确保导入密钥key_id与令牌中key_id匹配。
400	KMS.2604	The external key plaintext length must be 32 bits.	外部密钥明文长度必须为32位。	请传递正确的参数。
400	KMS.2605	Token verification failed.	令牌校验失败。	请重新获取令牌。

状态码	错误码	错误信息	描述	处理措施
400	KMS.2606	You are importing a deleted key again. The imported plaintext must be the same as the deleted key plaintext.	重新导入一个已删除的密钥材料时，外部密钥明文应与之前导入的一致。	请确保导入密钥明文与之前导入密钥明文数据一致。
400	KMS.2607	The import sm4 plain key length must 16.	导入sm4纯密钥长度必须为16。	请传递正确的参数。
400	KMS.2608	The imported asymmetric private key can not be null.	导入的非对称私钥不能为空。	请传递正确的参数。
400	KMS.2609	The imported asymmetric private is invalid.	导入的非对称私有非法。	请传递正确的参数。
400	KMS.2610	The temporary key length must be 16 or 32.	临时密钥长度必须为16或32。	请传递正确的参数。
400	KMS.2701	Key material is not in Enabled or Disabled state and cannot be deleted.	密钥材料只有在启用、禁用状态下方可被删除。	请确保密钥在“启用”、“禁用”状态。
400	KMS.2702	The imported private key material can not be deleted.	导入的私钥材料无法删除。	导入的私钥材料无法删除。
400	KMS.2801	End_time must bigger than start_time.	End time必须大于start time。	请传递正确的参数。

状态码	错误码	错误信息	描述	处理措施
400	KMS.2901	Key rotation is not disabled.	密钥轮换未被禁用。	请禁用密钥轮换。
400	KMS.3001	Invalid rotation_interval.	rotation_interval不在有效数字范围内。	请传递正确的参数。
400	KMS.3103	Invalid id of tenant.	租户id非法。	请输入合法的租户id。
400	KMS.3201	Generate order id failed.	生成订单失败。	请重试或联系技术支持。
400	KMS.3202	KMS error.	订单资源错误。	请重试或联系技术支持。
400	KMS.3702	Invalid kms status parameter.	kms状态参数非法。	请传递正确的参数。
400	KMS.3801	KMS key sync enable and configuration failed.	KMS密钥同步和配置失败。	请重试或联系技术支持。
400	KMS.3802	Failed to sync operation convert to born region.	当前不支持密钥库。	联系技术支持。
400	KMS.3803	The key of born region is turn off sync ,so no operating can be performed.	当前不支持密钥库。	联系技术支持。
400	KMS.3804	AttestationDocument parsing failed.	文档解析失败。	请重试或联系技术支持。
400	KMS.3902	Invalid Service-Transaction-Id in request header.	请求header的Service-Transaction-Id参数非法。	请联系技术支持。
400	KMS.3903	Service-Transaction-Id is missing in request header.	请求header不存在Service-Transaction-Id参数。	请联系技术支持。

状态码	错误码	错误信息	描述	处理措施
400	KMS.4001	Service name to notify is illegal.	用于通知的服务名称非法。	请联系技术支持。
400	KMS.4002	Service url to notify is illegal.	用于通知的服务url非法。	请联系技术支持。
400	KMS.5022	Tags are not compliant.	标签不合规。	请重试或联系技术支持。
400	KMS.5023	Pdp5 Header is invalid.	Header参数非法。	请重试或联系技术支持。
403	KMS.0301	Invalid or null X-Auth-Token.	X-Auth-Token为null或字符非法。	请重新获取token，并在使用时确保token字符串的完整性。
403	KMS.0302	Invalid X-Auth-Token.	X-Auth-Token无效。	请重新获取token，并在使用时确保token字符串的完整性。
403	KMS.0303	X-Auth-Token expired.	X-Auth-Token过期。	请重新获取token，并在使用时确保token字符串的完整性。
403	KMS.0304	X-Auth-Token contains the OBT tag and cannot be used to access services.	X-Auth-Token包含公测标签，不能访问服务。	请重新获取token，并在使用时确保token字符串的完整性。
403	KMS.0305	Invalid X-Auth-Token project name.	X-Auth-Token Project Name区域非法。	请重新获取token，并在使用时确保token字符串的完整性。
403	KMS.0306	No access permissions.	用户无权限访问密钥。	请联系管理员给账户添加KMS CMKFullaccess权限。
403	KMS.0307	No access permissions.	用户角色无权限访问接口。	请联系管理员给账户添加KMS CMKFullaccess权限。

状态码	错误码	错误信息	描述	处理措施
403	KMS.0314	Real-name authentication is required to access the API.	用户未通过实名认证，不能访问该接口。	请完成实名认证后重试。
403	KMS.0321	URIs in URL and X-Auth-Token do not match.	URL中包含的URL和X-Auth-Token中包含的URL不一致。	请确保URI和token中的项目ID一致后重试。
403	KMS.0322	The user has no permission to access the partition.	用户缺少分区类型的权限。	请配置权限后重试。
403	KMS.0326	No access permissions.	用户角色无权限访问接口。	请联系管理员给账户添加KMS CMKFullaccess权限。
403	KMS.0328	KMS has been frozen. Renew it and try again.	密钥管理已冻结，请续费解冻后重试。	请续费后重试。
403	KMS.0330	User has no permission.	用户缺少相关权限。	配置KMS CMKFullaccess权限后重试。
404	KMS.0207	The key does not exist.	密钥不存在。	请先创建密钥。
404	KMS.0316	No exist partition_id.	分区类型不存在。	输入分区类型。
404	KMS.0416	Invalid tag ID.	tag不存在。	请传递存在的密钥标签。
404	KMS.3901	The requested jobId cannot be found.	任务id未找到。	请联系技术支持。
405	KMS.0215	Request method not supported.	请求方法不支持。	请使用支持的请求方式。
500	KMS.0100	Get-status error.	获取服务状态失败。	请联系技术支持。
500	KMS.0101	KMS error.	KMS服务错误。	请重试或联系客服。

状态码	错误码	错误信息	描述	处理措施
500	KMS.0102	Abnormal KMS I/O.	KMS I/O异常。	请重试或联系客服。
500	KMS.0103	A system exception occurred. Contact technical support.	发生系统异常。请联系技术支持。	请联系技术支持。
500	KMS.3805	Failed to build digital envelope.	无法构建数字信封。	请重试或联系技术支持。
500	KMS.3806	Unable to obtain enclave root certificate, please contact technical support.	无法获取根证书，请联系技术支持。	联系技术支持。
500	KMS.5012	Request PDP service failed.	访问PDP服务失败。	请重试或联系技术支持。
500	KMS.5021	Check pdp5 auth failed.	PDP5鉴权失败。	请重试或联系技术支持。
503	KMS.0104	A system exception occurred. Contact technical support.	发生系统异常。请联系技术支持。	请联系技术支持。
400	KPS.0001	taskId is illegal.	task id非法。	使用合法的task id。
400	KPS.0002	parameter error.	参数错误。	使用正确的参数。
400	KPS.0005	Failed task is not found.	task id错误。	填入正确的task id。
400	KPS.0006	User not found.	用户名错误。	使用正确的用户名。
400	KPS.4016	The key pair is not exist.	密钥对名称错误。	填入正确的密钥对名称。
400	KPS.6004	No Keypair find.	未找到密钥对。	填入正确的密钥对名称。

状态码	错误码	错误信息	描述	处理措施
400	KPS.6005	No private key managed.	未找到托管的私钥。	确认私钥是否已托管。
400	KPS.6008	Encrypt private key failed.	加密私钥失败。	确认kms密钥是否存在以及状态是否可用。
400	KPS.6010	Save privatekey failed.	保存私钥失败。	确认kms密钥是否存在以及状态是否可用。
400	KPS.6011	The imported private key not match public key.	导入的私钥和公钥不匹配。	确认导入的公私钥对是否匹配。
401	KPS.9001	The token of the request is not or failed to be authenticated.	token不合法。	使用合法的token。
401	KPS.9002	Public test service denied.	访问失败。	使用非公测账号。
403	KPS.6009	Keypair verify failed.	密钥对校验失败。	使用正确的托管校验码。
403	KPS.9003	No operation permission.	无权限访问。	添加对应的用户DEW KeypairFullaccess权限。
403	KPS.9004	The account is frozen.	账户被冻结。	账户被冻结。
403	KPS.9005	The account is restricted.	账户被限制。	账户被限制。
403	KPS.9006	Unknown user type.	账户无权限。	添加对应的用户DEW KeypairFullaccess权限。

A.3 获取项目 ID

调用 API 获取项目 ID

项目ID可以通过调用**查询指定条件下的项目信息**API获取。

获取项目ID的接口为“GET https://{Endpoint}/v3/projects”，其中{Endpoint}为IAM的终端节点，可以从**地区和终端节点**获取。接口的认证鉴权请参见**认证鉴权**。

响应示例如下，其中projects下的“id”即为项目ID。

```
{
    "projects": [
        {
            "domain_id": "65382450e8f64ac0870cd180d14e684b",
            "is_domain": false,
            "parent_id": "65382450e8f64ac0870cd180d14e684b",
            "name": "xxxxxxxx",
            "description": "",
            "links": {
                "next": null,
                "previous": null,
                "self": "https://www.example.com/v3/projects/a4a5d4098fb4474fa22cd05f897d6b99"
            },
            "id": "a4a5d4098fb4474fa22cd05f897d6b99",
            "enabled": true
        }
    ],
    "links": {
        "next": null,
        "previous": null,
        "self": "https://www.example.com/v3/projects"
    }
}
```

从控制台获取项目 ID

在调用接口的时候，部分URL中需要填入项目编号，所以需要获取到项目编号。项目编号获取步骤如下：

1. 登录管理控制台。
2. 单击用户名，在下拉列表中单击“基本信息”。
3. 在基本信息页面单击“管理我的凭证”。
   在“API凭证”页面的项目列表中查看项目ID。

图 A-1 查看项目 ID