密钥管理系统

常见问题

腾讯云

版权所有：腾讯云计算（北京）有限责任公司

【版权声明】

©2013-2025 腾讯云版权所有

本文档（含所有文字、数据、图片等内容）完整的著作权归腾讯云计算（北京）有限责任公司单独所有，未经腾讯云事先明确书面许可，任何主体不得以任何形式复制、修改、使用、抄袭、传播本文档全部或部分内容。前述行为构成对腾讯云著作权的侵犯，腾讯云将依法采取措施追究法律责任。

【商标声明】

腾讯云

及其它腾讯云服务相关的商标均为腾讯云计算（北京）有限责任公司及其关联公司所有。本文档涉及的第三方主体的商标，依法由权利人所有。未经腾讯云及有关权利人书面许可，任何主体不得以任何方式对前述商标进行使用、复制、修改、传播、抄录等行为，否则将构成对腾讯云及有关权利人商标权的侵犯，腾讯云将依法采取措施追究法律责任。

【服务声明】

本文档意在向您介绍腾讯云全部或部分产品、服务的当时的相关概况，部分产品、服务的内容可能不时有所调整。您所购买的腾讯云产品、服务的种类、服务标准等应由您与腾讯云之间的商业合同约定，除非双方另有约定，否则，腾讯云对本文档内容不做任何明示或默示的承诺或保证。

【联系我们】

我们致力于为您提供个性化的售前购买咨询服务，及相应的技术售后服务，任何问题请联系 4009100100或95716。

文档目录

常见问题
一般性问题
开发接入相关问题
指数回退策略应对服务限频

常见问题
一般性问题

最近更新时间：2025-07-04 18:15:02

腾讯云 KMS 可以做什么？

腾讯云 KMS 提供加密过程中的对称密钥与非对称密钥的全生命周期管理，包括生成、存储、启用/禁用、分发、轮换、审计、销毁等，满足用户多应用多业务的密钥管理需求，符合监管和合规要求。同时，KMS 提供敏感数据加密、信封加密、云产品集成等加密能力，方便用户专注于业务开发。

哪些云服务支持密钥管理系统加密数据？

KMS 服务无缝对接腾讯云 TencentDB、COS 和 CBS 等云产品，通过 KMS 提供信封加密的方式对云产品数据进行加密。

密钥管理系统控制台中根密钥与云产品密钥有什么区别吗？

- **根密钥**是用户通过控制台或 API 来创建的用户主密钥。您可以对用户密钥进行创建/启用/禁用/轮换/权限控制等操作。
- **云产品密钥**是腾讯云产品/服务（例如 CBS、COS、TDSQL 等）在调用密钥管理系统时，自动为用户创建的 CMK。您可以对云产品密钥进行查询及开启密钥轮换操作，不支持禁用、计划删除操作。

海量数据如何进行加密？

对于较大的数据加密，建议采用 信封加解密 方案，实现本地高性能数据加解密。

KMS 会限制用户访问频率吗？

KMS 会对用户访问的频率进行限制。
对于同一个主账号，主账号及其子账号访问 KMS 服务单一接口的频率之和最高为15000次/每秒。其中，主账号访问 KMS 服务单一接口的最高频率为15000次/每秒，子账号访问 KMS 服务单一接口的最高频率需参考对应接口的 API 文档说明。

开发接入相关问题

最近更新时间：2025-07-04 18:15:02

SDK 中的 SecretID 和 SecretKey 在哪里获取?

您需使用主账号登录 API 密钥管理控制台 获取您的 SecretID 和 SecretKey。**请您务必保存好您的 SecretID 和 SecretKey 不被泄露。**

如何创建根密钥 CMK？

创建根密钥有三种方式，分别是通过 密钥管理系统控制台 、腾讯云命令行工具 TCCLI 及 API 接口请求 。

创建根密钥 CMK 是否有个数限制?

是。每账户每区域下限制创建200个 CMK，计划删除状态下的除外。不包含云产品密钥。如需创建更多的 CMK，请 提交工单 或联系腾讯云商务。

创建密钥时，密钥材料来源可以选择外部，外部是指什么？BYOK 方案是指什么?

- **外部**是指使用用户自己的密钥材料。
- **BYOK**（Bring Your Own Key）是实现用户使用自己密钥材料的一个方案，其方式是通过 KMS 服务生成一个密钥材料为空的 CMK，并将自己的密钥材料导入到该根密钥中，形成一个外部密钥 CMK（EXTERNAL CMK），再由 KMS 服务进行该外部密钥的分发管理。

修改根密钥的别名或描述信息，通过接口请求的方式，需要多久才能生效?

接口成功请求后会**立即生效**。

是否支持轮换根密钥 CMK？如何开启?

支持轮换。可以通过 密钥管理系统控制台 、命令行工具 或 API 接口 三种方式进行开启操作。

> **注意：**
> 不支持轮换的根密钥：
> - 非对称的根密钥 CMK
> - 使用外部密钥材料的根密钥 CMK

开启轮换后，业务是否需要做更改?

- 密钥轮换只会更改根密钥的密钥材料，根密钥的属性（密钥 ID、别名、描述、权限）不会发生变化。
- 开启密钥轮换后，密钥管理服务会根据设置的轮换周期（默认365天）自动轮换密钥，每次轮换都会生成一个新版本的根密钥，轮换的密钥加解密数据的方式如下所示：
  - 加密数据时，KMS 会自动使用当前最新版本的根密钥来执行加密操作。

- 解密数据时，KMS 会自动使用加密时所使用的根密钥来执行解密操作。

如何选择数据加密算法?

- **对称加解密**：对称加解密算法包括 SM4和 AES256，其算法的选择是系统根据创建主密钥时上传的地区自动分配。如地区选择的是“中国国内站”，即系统会选择 SM4算法；当选择的是“非中国国内站”，则系统会选择 AES256算法。
- **非对称加解密**：非对称加解密算法包括模长为2048比特的 RSA 密钥和 SM2，其算法的选择由您创建主密钥时选择的地区和 KeyUsage 共同决定。

> **注意：**
> 通过 API 接口方式创建根密钥，建议在创建之前先查询当前地区支持的 **加密方式**，从而确保创建的正确性。

指数回退策略应对服务限频

最近更新时间：2025-05-30 18:02:02

异常处理的策略建议

请求 KMS API 接口，即应用程序的请求发送到 KMS 远程服务器时，若出现异常报错，建议您可以采用以下策略进行处理：

- **取消**：当返回错误显示故障是非暂时性，或重新执行后也无法成功，则应当终止/取消程序调用并报告异常。
- **重试**：当返回错误是不常见或比较少见，如网络包在传输过程中损坏，但被发送，这种情况下可以立即采取重试。
- **延迟重试**：当返回错误是普通的连接或繁忙相关导致，则服务可能需要短时间的恢复，从而清除堆积的工作等，此类问题，等待一个合适时间后进行重试。

本文以延迟重试的策略展开说明，上述提到的等待时间（即延迟时间），可以采取逐步增加方式，或使用定时策略（如指数回退）来实现。因为调用 KMS API 接口服务会限制频率，所以当您调用的并发过高时，可以采用延迟重试的方法来避免限频所带来的问题。

指数回退

伪代码

```
//使用逐步增加的方式来延迟重试某个操作
InitDelayValue = 100
For(Retries = 0; Retries < MAX_RETRIES; Retries = Retries+1)
        wait for (2^Retries * InitDelayValue) milliseconds
        Status = KmsApiRequest()
        IF Status == SUCCESS
            BREAK // Succeeded, stop calling the API again.
        ELSE IF Status == THROTTLED || Status == SERVER_NOT_READY
            CONTINUE // Failed due to throttling or server busy, try
again.
        ELSE
            BREAK // another error occurs, stop calling the API again.
        END IF
```

策略运用

Python 示例：调用 KMS Encrypt 接口遇到限频错误时，如何使用指数回退方法处理。

```
# -*- coding: utf-8 -*-
import base64
```

```
import math
import time
import os
from tencentcloud.common import credential
from tencentcloud.common.exception.tencent_cloud_sdk_exception import
TencentCloudSDKException
from tencentcloud.common.profile.client_profile import ClientProfile
from tencentcloud.common.profile.http_profile import HttpProfile
from tencentcloud.kms.v20190118 import kms_client, models


def KmsInit(region="ap-guangzhou", secretId="", secretKey=""):
        try:
            credProfile = credential.Credential(secretId, secretKey)
            client = kms_client.KmsClient(credProfile, region)
            return client
        except TencentCloudSDKException as err:
            print(err)
            return None

def BackoffFunction(RetryCount):
        InitDelayValue = 100
        DelayTime = math.pow(2, RetryCount) * InitDelayValue
        return DelayTime

if __name__ == '__main__':
        # 用户自定义参数
        secretId = os.getenv('SECRET_ID')         # read from
environment variable or use whitebox encryption to protect secret ID
        secretKey = os.getenv('SECRET_KEY')    # read from
environment variable or use whitebox encryption to protect secret key
region = "ap-guangzhou"
        region = "ap-guangzhou"
        keyId = "replace-with-realkeyid"
        plaintext = "abcdefg123456789abcdefg123456789abcdefg"
        Retries = 0
        MaxRetries = 10
        client = KmsInit(region, secretId, secretKey)
        req = models.EncryptRequest()
        req.KeyId = keyId
        req.Plaintext = base64.b64encode(plaintext)
        while Retries < MaxRetries:
            try:
```

```
                    Retries += 1
                    rsp = client.Encrypt(req)  # 调用加密接口
                    print 'plaintext: ',plaintext,'CiphertextBlob:
',rsp.CiphertextBlob
                    break
                except TencentCloudSDKException as err:
                    if err.code == 'InternalError' or err.code ==
'RequestLimitExceeded':
                        if Retries == MaxRetries:
                            break
                        time.sleep(BackoffFunction(Retries + 1))
                        continue
                    else:
                        print(err)
                        break
                except Exception as err:
                    print(err)
                    break
```

注意：

- 如需解决其他特定的错误，您可以直接对 except 语句的内容做更改、调整即可。
- 根据自己的代码逻辑、业务策略等，进行定时策略的规划与制定，从而设置最优的**初始延迟值**（InitDelayValue）及**重试次数**（Retries），避免阈值设置过低或过高，影响整体业务运转。