物联网通信

API 文档

腾讯云

文档目录

API 文档
产品版本

最近更新时间：2022-05-11 10:31:54

您正在浏览iotcloud产品文档的版本: 2021-04-08

最新版本

- 2021-04-08 (当前版本)

以往版本

- 2018-06-14

更新历史

最近更新时间：2024-01-31 01:16:38

第 10 次发布

发布时间：2024-01-31 01:16:26

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- UpdateOtaTaskStatus

修改接口：

- BatchUpdateFirmware
  - 新增入参：Type
- EditFirmware
  - 新增入参：FirmwareUserDefined
- UploadFirmware
  - 新增入参：FirmwareUserDefined

第 9 次发布

发布时间：2023-08-09 01:29:39

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- DescribeDevice
  - 新增出参：NBIoTDeviceID
  - **废弃出参：** NbiotDeviceID

修改数据结构：

- CLSLogItem
  - 新增成员：UserId
  - **废弃成员：** Userid
- DeviceInfo
  - 新增成员：NBIoTDeviceID
  - **废弃成员：** NbiotDeviceID
- ProductProperties
  - 新增成员：AppEUI
  - **废弃成员：** Appeui

第 8 次发布

发布时间：2023-06-14 01:27:08

本次发布包含了以下内容：

改善已有的文档。

修改数据结构：

- DeviceInfo
  - 新增成员：CreateUserId
- FirmwareInfo
  - 新增成员：FwType, CreateUserId

第 7 次发布

发布时间：2023-06-12 01:24:01

本次发布包含了以下内容：

改善已有的文档。

修改数据结构：

- ProductMetadata
  - 新增成员：CreateUserId, UserId
- ProductProperties
  - 新增成员：DeviceLimit, ForbiddenStatus

第 6 次发布

发布时间：2022-08-16 06:27:22

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- DeleteDeviceShadow

第 5 次发布

发布时间：2022-05-27 06:10:42

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- ListTopicRules

新增数据结构：

- TopicRuleInfo

第 4 次发布

发布时间：2022-05-20 06:13:13

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- GetAllVersion
- ListFirmwares

新增数据结构：

- FirmwareInfo

第 3 次发布

发布时间：2022-05-11 10:24:44

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- BatchUpdateFirmware
- BindDevices
- CancelDeviceFirmwareTask
- CreateMultiDevicesTask
- CreateProduct
- CreateTaskFileUrl
- CreateTopicPolicy
- CreateTopicRule
- DeleteDeviceResource
- DeleteTopicRule
- DescribeDeviceClientKey
- DescribeDeviceResource
- DescribeDeviceResources
- DescribeDeviceShadow
- DescribeFirmware
- DescribeFirmwareTask
- DescribeFirmwareTaskDevices
- DescribeFirmwareTaskDistribution
- DescribeFirmwareTaskStatistics
- DescribeFirmwareTasks
- DescribeGatewayBindDevices
- DescribeProductResource
- DescribeProductResources
- DescribeProductTask
- DescribeProductTasks
- DescribeProducts
- DescribePushResourceTaskStatistics
- DescribeResourceTasks
- DisableTopicRule
- DownloadDeviceResource
- EditFirmware
- EnableTopicRule
- GetCOSURL
- GetUserResourceInfo
- PublishMessage
- PublishRRPCMessage
- ReplaceTopicRule
- ResetDeviceState
- RetryDeviceFirmwareTask

- UnbindDevices
- UpdateDeviceAvailableState
- UpdateDeviceShadow
- UpdateTopicPolicy
- UploadFirmware

新增数据结构：

- BindDeviceInfo
- BrokerSubscribe
- DeviceResourceInfo
- DeviceUpdateStatus
- FirmwareTaskInfo
- ProductInfo
- ProductResourceInfo
- ProductTaskInfo
- ResetDeviceResult
- SearchKeyword
- StatusStatistic
- TopicRulePayload

第 2 次发布

发布时间：2022-04-01 15:20:13

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- DeleteProductPrivateCA
- ListLog
- ListLogPayload
- ListSDKLog
- PublishBroadcastMessage
- SetProductsForbiddenStatus
- UpdateDevicePSK
- UpdateProductDynamicRegister
- UpdateProductPrivateCA

修改接口：

- DescribeDevice
  - 新增出参：CreateUserId

新增数据结构：

- CLSLogItem
- PayloadLogItem
- SDKLogItem

第 1 次发布

发布时间：2021-07-21 08:06:35

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- CreateDevice
- CreatePrivateCA
- DeleteDevice
- DeletePrivateCA
- DeleteProduct
- DescribeDevice
- DescribeDevices
- DescribePrivateCA
- DescribePrivateCABindedProducts
- DescribePrivateCAs
- DescribeProduct
- DescribeProductCA
- UpdateDeviceLogLevel
- UpdateDevicesEnableState
- UpdatePrivateCA

新增数据结构：

- Attribute
- BindProductInfo
- CertInfo
- DeviceInfo
- DeviceLabel
- DeviceTag
- ProductMetadata
- ProductProperties

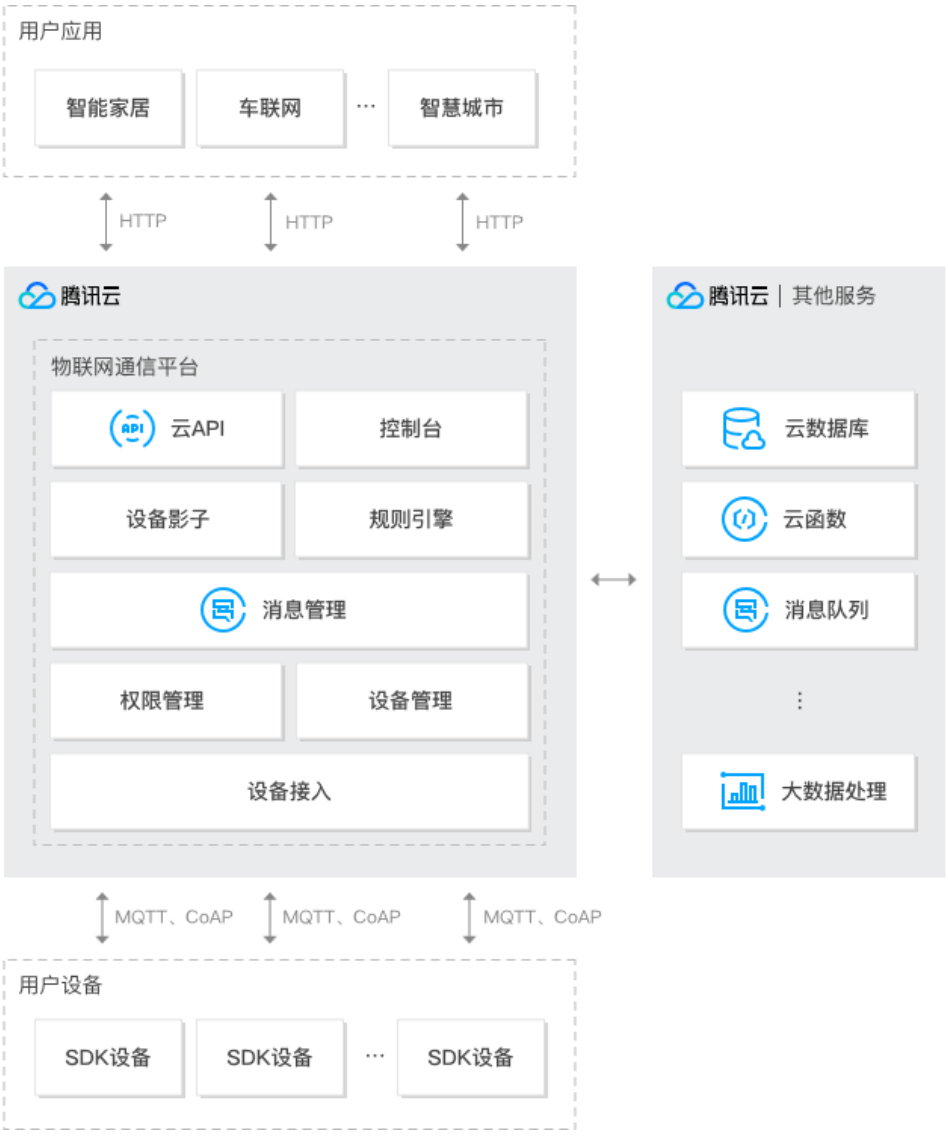
简介

最近更新时间：2022-05-11 10:31:53

欢迎使用 物联网通信 API 3.0 版本。全新的 API 接口文档更加规范和全面，统一的参数风格和公共错误码，统一的 SDK/CLI 版本与 API 文档严格一致，给您带来简单快捷的使用体验。支持全地域就近接入让您更快连接腾讯云产品。更多腾讯云 API 3.0 使用介绍请查看：快速入门

物联网通信 (IoT Hub) 服务，旨在提供一个安全、稳定、高效的连接平台，帮助开发者快速且低成本地实现“设备-设备”、“设备-用户应用”、“设备-云服务”之间可靠、高并发的数据通信。腾讯物联网通信可以实现设备之间的互动、设备的数据上报和配置下发，还可以基于规则引擎和腾讯云产品打通，方便快捷的实现海量设备数据的存储、计算以及智能分析。总的来说，基于腾讯物联网通信，开发者可以低成本实现“设备-数据-应用-云服务”的连接，快速搭建物联网应用平台。

产品架构



- 接入腾讯物联网通信
  用户设备可使用 SDK 接入腾讯物联网通信。底层数据传输基于 MQTT 或 CoAP 协议，可以有效减少网络带宽。同时也支持 HTTP、WebSocket 接入。安全方面引入网络安全传输协议（TLS、DTLS），可以防范非法接入和数据窃取、篡改等风险。介于设备资源和使用场景的多样性，支持选择非对称（设备证书加密验证、适用高安全要求场景）和对称加密（密钥加密验证、适用资源受限设备）方式。

- 用户设备基于 SDK 进行消息的发布和订阅
  为了实现设备数据安全隔离，目前腾讯云限制设备只能发布和订阅自身 topic ，但可以通过配置规则引擎实现设备与其他实体的消息互通。

- 用户可以在控制台配置规则引擎实现设备与其他实体的消息互通
  目前规则引擎支持类 SQL 语法操作，可通过 repub（重新发布消息）实现设备之间的消息通信能力、forward（转发消息到用户服务器）实现设备消息转发第三方服务。设备消息和腾讯云其他产品（存储、函数计算、大数据分析套件等）的打通功能也在建设中。

- 打通设备消息与第三方服务
  作为设备的唯一接入方，物联网通信平台通过开通消息队列服务，可快速将设备指定消息写入腾讯云 CMQ、CKafka 队列，第三方服务通过 CMQ、CKafka 队列 SDK 取用消费数据，实现设备与第三方服务的异步消息通信。

- 用户可以基于设备影子实现设备与应用之间配置数据、状态数据的双向同步
  一方面，用户可以通过云 API 将配置参数设置到设备影子里，设备在线或上线时，都可以从设备影子获取配置参数。另一方面，设备可以将最新状态上报到设备影子。用户查询设备状态时，只需查询设备影子，而不必与设备进行直接网络通信。

- 用户可以通过云 API 实现设备管理
  对于物联场景下设备的管理能力，提供便捷的 SDK 工具，可在后台快速、批量化创建、查询、操作设备，提高效率。当前支持 Python、PHP、java 工具包。

调用方式
请求结构

最近更新时间：2023-02-22 01:31:57

1. 服务地址

API 支持就近地域接入，本产品就近地域接入域名为 iotcloud.tencentcloudapi.com ，也支持指定地域域名访问，例如广州地域的域名为 iotcloud.ap-guangzhou.tencentcloudapi.com 。

推荐使用就近地域接入域名。根据调用接口时客户端所在位置，会自动解析到**最近的**某个具体地域的服务器。例如在广州发起请求，会自动解析到广州的服务器，效果和指定 iotcloud.ap-guangzhou.tencentcloudapi.com 是一致的。

注意：对时延敏感的业务，建议指定带地域的域名。

注意：域名是 API 的接入点，并不代表产品或者接口实际提供服务的地域。产品支持的地域列表请在调用方式/公共参数文档中查阅，接口支持的地域请在接口文档输入参数中查阅。

目前支持的域名列表为：

接入地域	域名
就近地域接入（推荐，只支持非金融区）	iotcloud.tencentcloudapi.com
华南地区(广州)	iotcloud.ap-guangzhou.tencentcloudapi.com
华东地区(上海)	iotcloud.ap-shanghai.tencentcloudapi.com
华北地区(北京)	iotcloud.ap-beijing.tencentcloudapi.com
西南地区(成都)	iotcloud.ap-chengdu.tencentcloudapi.com
西南地区(重庆)	iotcloud.ap-chongqing.tencentcloudapi.com
港澳台地区(中国香港)	iotcloud.ap-hongkong.tencentcloudapi.com
亚太东南(新加坡)	iotcloud.ap-singapore.tencentcloudapi.com
亚太东南(曼谷)	iotcloud.ap-bangkok.tencentcloudapi.com
亚太南部(孟买)	iotcloud.ap-mumbai.tencentcloudapi.com
亚太东北(首尔)	iotcloud.ap-seoul.tencentcloudapi.com
亚太东北(东京)	iotcloud.ap-tokyo.tencentcloudapi.com
美国东部(弗吉尼亚)	iotcloud.na-ashburn.tencentcloudapi.com
美国西部(硅谷)	iotcloud.na-siliconvalley.tencentcloudapi.com
北美地区(多伦多)	iotcloud.na-toronto.tencentcloudapi.com
欧洲地区(法兰克福)	iotcloud.eu-frankfurt.tencentcloudapi.com

注意：由于金融区和非金融区是隔离不互通的，因此当访问金融区服务时（公共参数 Region 为金融区地域），需要同时指定带金融区地域的域名，最好和 Region 的地域保持一致。

金融区接入地域	金融区域名
华东地区(上海金融)	iotcloud.ap-shanghai-fsi.tencentcloudapi.com

金融区接入地域	金融区域名
华南地区(深圳金融)	iotcloud.ap-shenzhen-fsi.tencentcloudapi.com

2. 通信协议

腾讯云 API 的所有接口均通过 HTTPS 进行通信，提供高安全性的通信通道。

3. 请求方法

支持的 HTTP 请求方法:

- POST（推荐）
- GET

POST 请求支持的 Content-Type 类型:

- application/json（推荐），必须使用签名方法 v3（TC3-HMAC-SHA256）。
- application/x-www-form-urlencoded，必须使用签名方法 v1（HmacSHA1 或 HmacSHA256）。
- multipart/form-data（仅部分接口支持），必须使用签名方法 v3（TC3-HMAC-SHA256）。

GET 请求的请求包大小不得超过32KB。POST 请求使用签名方法 v1（HmacSHA1、HmacSHA256）时不得超过1MB。POST 请求使用签名方法 v3（TC3-HMAC-SHA256）时支持10MB。

4. 字符编码

均使用 UTF-8 编码。

公共参数

最近更新时间：2023-12-28 01:17:13

公共参数是用于标识用户和接口签名的参数，如非必要，在每个接口单独的文档中不再对这些参数进行说明，但每次请求均需要携带这些参数，才能正常发起请求。

公共参数的具体内容会因您使用的签名方法版本不同而有所差异。

使用签名方法 v3 的公共参数

签名方法 v3（有时也称作 TC3-HMAC-SHA256）相比签名方法 v1（有些文档可能会简称签名方法），更安全，支持更大的请求包，支持 POST JSON 格式，性能有一定提升，推荐使用该签名方法计算签名。完整介绍详见 签名方法 v3。

注意：出于简化的目的，部分接口文档中的示例使用的是签名方法 v1 GET 请求，而不是更安全的签名方法 v3。

使用签名方法 v3 时，公共参数需要统一放到 HTTP Header 请求头部中，如下表所示：

参数名称	类型	必选	描述
Action	String	是	HTTP 请求头：X-TC-Action。操作的接口名称。取值参考接口文档输入参数章节关于公共参数 Action 的说明。例如云服务器的查询实例列表接口，取值为 DescribeInstances。
Region	String	-	HTTP 请求头：X-TC-Region。地域参数，用来标识希望操作哪个地域的数据。取值参考接口文档中输入参数章节关于公共参数 Region 的说明。**注意：某些接口不需要传递该参数，接口文档中会对此特别说明，此时即使传递该参数也不会生效。**
Timestamp	Integer	是	HTTP 请求头：X-TC-Timestamp。当前 UNIX 时间戳，可记录发起 API 请求的时间。例如 1529223702。**注意：如果与服务器时间相差超过5分钟，会引起签名过期错误。**
Version	String	是	HTTP 请求头：X-TC-Version。操作的 API 的版本。取值参考接口文档中入参公共参数 Version 的说明。例如云服务器的版本 2017-03-12。
Authorization	String	是	HTTP 标准身份认证头部字段，例如： TC3-HMAC-SHA256 Credential=AKIDEXAMPLE/Date/service/tc3_request, SignedHeaders=content-type;host, Signature=fe5f80f77d5fa3beca038a248ff027d0445342fe2855ddc963176630326f1024 其中， - TC3-HMAC-SHA256：签名方法，目前固定取该值； - Credential：签名凭证，AKIDEXAMPLE 是 SecretId；Date 是 UTC 标准时间的日期，取值需要和公共参数 X-TC-Timestamp 换算的 UTC 标准时间日期一致；service 为具体产品名，通常为域名前缀。例如，域名 cvm.tencentcloudapi.com 意味着产品名是 cvm。本产品取值为 iotcloud；tc3_request 为固定字符串； - SignedHeaders：参与签名计算的头部信息，content-type 和 host 为必选头部； - Signature：签名摘要，计算过程详见 文档。
Token	String	否	HTTP 请求头：X-TC-Token。即 安全凭证服务 所颁发的临时安全凭证中的 Token，使用时需要将 SecretId 和 SecretKey 的值替换为临时安全凭证中的 TmpSecretId 和 TmpSecretKey。使用长期密钥时不能设置此 Token 字段。
Language	String	否	HTTP 请求头：X-TC-Language。指定接口返回的语言，仅部分接口支持此参数。取值：zh-CN，en-US。zh-CN 返回中文，en-US 返回英文。

假设用户想要查询广州地域的云服务器实例列表中的前十个，接口参数设置为偏移量 Offset=0，返回数量 Limit=10，则其请求结构按照请求 URL、请求头部、请求体示例如下：

HTTP GET 请求结构示例：

```
https://cvm.tencentcloudapi.com/?Limit=10&Offset=0

Authorization: TC3-HMAC-SHA256 Credential=AKID********EXAMPLE/2018-10-09/cvm/tc3_request, SignedHeaders=content-type;host, Signature=5da7a33f6993f0614b047e5df4582db9e9bf4672ba50567dba16c6ccf174c474
Content-Type: application/x-www-form-urlencoded
Host: cvm.tencentcloudapi.com
X-TC-Action: DescribeInstances
X-TC-Version: 2017-03-12
X-TC-Timestamp: 1539084154
X-TC-Region: ap-guangzhou
```

HTTP POST（application/json）请求结构示例：

```
https://cvm.tencentcloudapi.com/

Authorization: TC3-HMAC-SHA256 Credential=AKID********EXAMPLE/2018-05-30/cvm/tc3_request, SignedHeaders=content-type;host, Signature=582c400e06b5924a6f2b5d7d672d79c15b13162d9279b0855cfba6789a8edb4c
Content-Type: application/json
Host: cvm.tencentcloudapi.com
X-TC-Action: DescribeInstances
X-TC-Version: 2017-03-12
X-TC-Timestamp: 1527672334
X-TC-Region: ap-guangzhou

{"Offset":0,"Limit":10}
```

HTTP POST（multipart/form-data）请求结构示例（仅特定的接口支持）：

```
https://cvm.tencentcloudapi.com/

Authorization: TC3-HMAC-SHA256 Credential=AKID********EXAMPLE/2018-05-30/cvm/tc3_request, SignedHeaders=content-type;host, Signature=582c400e06b5924a6f2b5d7d672d79c15b13162d9279b0855cfba6789a8edb4c
Content-Type: multipart/form-data; boundary=58731222010402
Host: cvm.tencentcloudapi.com
X-TC-Action: DescribeInstances
X-TC-Version: 2017-03-12
X-TC-Timestamp: 1527672334
X-TC-Region: ap-guangzhou

--58731222010402
Content-Disposition: form-data; name="Offset"

0
--58731222010402
Content-Disposition: form-data; name="Limit"

10
--58731222010402--
```

使用签名方法 v1 的公共参数

使用签名方法 v1（有时会称作 HmacSHA256 和 HmacSHA1），公共参数需要统一放到请求串中，完整介绍详见文档

参数名称	类型	必选	描述
Action	String	是	操作的接口名称。取值参考接口文档中输入参数章节关于公共参数 Action 的说明。例如云服务器的查询实例列表接口，取值为 DescribeInstances。
Region	String	-	地域参数，用来标识希望操作哪个地域的数据。接口接受的地域取值参考接口文档中输入参数公共参数 Region 的说明。**注意：某些接口不需要传递该参数，接口文档中会对此特别说明，此时即使传递该参数也不会生效。**
Timestamp	Integer	是	当前 UNIX 时间戳，可记录发起 API 请求的时间。例如1529223702，如果与当前时间相差过大，会引起签名过期错误。
Nonce	Integer	是	随机正整数，与 Timestamp 联合起来，用于防止重放攻击。
SecretId	String	是	在 云API密钥 上申请的标识身份的 SecretId，一个 SecretId 对应唯一的 SecretKey ，而 SecretKey 会用来生成请求签名 Signature。
Signature	String	是	请求签名，用来验证此次请求的合法性，需要用户根据实际的输入参数计算得出。具体计算方法参见文档。
Version	String	是	操作的 API 的版本。取值参考接口文档中入参公共参数 Version 的说明。例如云服务器的版本 2017-03-12。
SignatureMethod	String	否	签名方式，目前支持 HmacSHA256 和 HmacSHA1。只有指定此参数为 HmacSHA256 时，才使用 HmacSHA256 算法验证签名，其他情况均使用 HmacSHA1 验证签名。
Token	String	否	即 安全凭证服务 所颁发的临时安全凭证中的 Token，使用时需要将 SecretId 和 SecretKey 的值替换为临时安全凭证中的 TmpSecretId 和 TmpSecretKey。使用长期密钥时不能设置此 Token 字段。
Language	String	否	指定接口返回的语言，仅部分接口支持此参数。取值：zh-CN，en-US。zh-CN 返回中文，en-US 返回英文。

假设用户想要查询广州地域的云服务器实例列表，其请求结构按照请求 URL、请求头部、请求体示例如下：

HTTP GET 请求结构示例：

```
https://cvm.tencentcloudapi.com/?Action=DescribeInstances&Version=2017-03-12&SignatureMethod=HmacSHA256&Timestamp=1527672334&Signature=37ac2f4fde00b0ac9bd9eadeb459b1bbee224158d66e7ae5fcadb70b2d181d02&Region=ap-guangzhou&Nonce=23823223&SecretId=AKID********EXAMPLE

Host: cvm.tencentcloudapi.com
```

HTTP POST 请求结构示例：

```
https://cvm.tencentcloudapi.com/

Host: cvm.tencentcloudapi.com
Content-Type: application/x-www-form-urlencoded

Action=DescribeInstances&Version=2017-03-12&SignatureMethod=HmacSHA256&Timestamp=1527672334&Signature=37ac2f4fde00b0ac9bd9eadeb459b1bbee224158d66e7ae5fcadb70b2d181d02&Region=ap-guangzhou&Nonce=23823223&SecretId=AKID********EXAMPLE
```

地域列表

本产品所有接口 Region 字段的可选值如下表所示。如果接口不支持该表中的所有地域，则会在接口文档中单独说明。

地域	取值
亚太东南（曼谷）	ap-bangkok
华南地区（广州）	ap-guangzhou
华东地区（上海金融）	ap-shanghai-fsi
欧洲地区（法兰克福）	eu-frankfurt
美国东部（弗吉尼亚）	na-ashburn

签名方法 v3

最近更新时间：2023-12-27 01:21:29

以下文档说明了签名方法 v3 的签名过程，但仅在您编写自己的代码来调用腾讯云 API 时才有用。我们推荐您使用 腾讯云 API Explorer ，腾讯云 SDK 和 腾讯云命令行工具（TCCLI）等开发者工具，从而无需学习如何对 API 请求进行签名。

推荐使用 API Explorer　　点击调试

您可以通过 API Explorer 的【签名串生成】模块查看每个接口签名的生成过程。

腾讯云 API 会对每个请求进行身份验证，用户需要使用安全凭证，经过特定的步骤对请求进行签名（Signature），每个请求都需要在公共参数中指定该签名结果并以指定的方式和格式发送请求。

为什么要进行签名

签名通过以下方式帮助保护请求：

1. 验证请求者的身份
   签名确保请求是由持有有效访问密钥的人发送的。请参阅控制台 云 API 密钥 页面获取密钥相关信息。

2. 保护传输中的数据
   为了防止请求在传输过程中被篡改，腾讯云 API 会使用请求参数来计算请求的哈希值，并将生成的哈希值加密后作为请求的一部分，发送到腾讯云 API 服务器。服务器会使用收到的请求参数以同样的过程计算哈希值，并验证请求中的哈希值。如果请求被篡改，将导致哈希值不一致，腾讯云 API 将拒绝本次请求。

签名方法 v3（TC3-HMAC-SHA256）功能上覆盖了以前的签名方法 v1，而且更安全，支持更大的请求，支持 JSON 格式，POST 请求支持传空数组和空字符串，性能有一定提升，推荐使用该签名方法计算签名。

首次接触，建议使用 API Explorer 中的“签名串生成”功能，选择签名版本为“API 3.0 签名 v3”，可以对生成签名过程进行验证，也可直接生成 SDK 代码。推荐使用腾讯云 API 配套的 8 种常见的编程语言 SDK，已经封装了签名和请求过程，均已开源，支持 Python、Java、PHP、Go、NodeJS、.NET、C++、Ruby。

申请安全凭证

本文使用的安全凭证为密钥，密钥包括 SecretId 和 SecretKey。每个用户最多可以拥有两对密钥。

- SecretId：用于标识 API 调用者身份，可以简单类比为用户名。
- SecretKey：用于验证 API 调用者的身份，可以简单类比为密码。
- **用户必须严格保管安全凭证，避免泄露，否则将危及财产安全。如已泄露，请立刻禁用该安全凭证。**

申请安全凭证的具体步骤如下：

1. 登录 腾讯云管理中心控制台 。
2. 前往 云API密钥 的控制台页面。
3. 在 云API密钥 页面，单击【新建密钥】创建一对密钥。

签名版本 v3 签名过程

云 API 支持 GET 和 POST 请求。对于GET方法，只支持 Content-Type: application/x-www-form-urlencoded 协议格式。对于POST方法，目前支持 Content-Type: application/json 以及 Content-Type: multipart/form-data 两种协议格式，json 格式绝大多数接口均支持，multipart 格式只有特定接口支持，此时该接口不能使用 json 格式调用，参考具体业务接口文档说明。推荐使用 POST 请求，因为两者的结果并无差异，但 GET 请求只支持 32 KB 以内的请求包。

下面以云服务器查询广州区实例列表作为例子，分步骤介绍签名的计算过程。我们选择该接口是因为：

1. 云服务器默认已开通，该接口很常用；
2. 该接口是只读的，不会改变现有资源的状态；
3. 接口覆盖的参数种类较全，可以演示包含数据结构的数组如何使用。

在示例中，不论公共参数或者接口的参数，我们尽量选择容易犯错的情况。在实际调用接口时，请根据实际情况来，每个接口的参数并不相同，不要照抄这个例子的参数和值。此外，这里只展示了部分公共参数和接口输入参数，用户可以根据实际需要添加其他参数，例如 Language 和 Token 公共参数（在 HTTP 头部设置，添加 X-TC- 前缀）。

假设用户的 SecretId 和 SecretKey 分别是：AKIDz8krbsJ5yKBZQpn74WFkmLPx3******* 和 Gu5t9xGARNpq86cd98joQYCN3*******。用户想查看广州区云服务器名为“未命名”的主机状态，只返回一条数据。则请求可能为：

```
curl -X POST https://cvm.tencentcloudapi.com \
-H "Authorization: TC3-HMAC-SHA256 Credential=AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******/2019-02-25/cvm/tc3_request, SignedHeaders=content-type;host;x-tc-action, Signature=be4f67d323c78ab9acb7395e43c0dbcf822a9cfac32fea2449a7bc7726b770a3" \
-H "Content-Type: application/json; charset=utf-8" \
-H "Host: cvm.tencentcloudapi.com" \
-H "X-TC-Action: DescribeInstances" \
-H "X-TC-Timestamp: 1551113065" \
-H "X-TC-Version: 2017-03-12" \
-H "X-TC-Region: ap-guangzhou" \
-d '{"Limit": 1, "Filters": [{"Values": ["\u672a\u547d\u540d"], "Name": "instance-name"}]}'
```

下面详细解释签名计算过程。

1. 拼接规范请求串

按如下伪代码格式拼接规范请求串（CanonicalRequest）：

```
CanonicalRequest =
HTTPRequestMethod + '\n' +
CanonicalURI + '\n' +
CanonicalQueryString + '\n' +
CanonicalHeaders + '\n' +
SignedHeaders + '\n' +
HashedRequestPayload
```

字段名称	解释
HTTPRequestMethod	HTTP 请求方法（GET、POST ）。此示例取值为 POST。
CanonicalURI	URI 参数，API 3.0 固定为正斜杠（/）。
CanonicalQueryString	发起 HTTP 请求 URL 中的查询字符串，对于 POST 请求，固定为空字符串""，对于 GET 请求，则为 URL 中问号（?）后面的字符串内容，例如：Limit=10&Offset=0。 注意：CanonicalQueryString 需要参考 RFC3986 进行 URLEncode 编码（特殊字符编码后需大写字母），字符集 UTF-8。推荐使用编程语言标准库进行编码。

字段名称	解释
CanonicalHeaders	参与签名的头部信息，至少包含 host 和 content-type 两个头部，也可加入其他头部参与签名以提高自身请求的唯一性和安全性，此示例额外增加了接口名头部。 拼接规则： 1. 头部 key 和 value 统一转成小写，并去掉首尾空格，按照 key:value\n 格式拼接； 2. 多个头部，按照头部 key（小写）的 ASCII 升序进行拼接。 此示例计算结果是 content-type:application/json; charset=utf-8\nhost:cvm.tencentcloudapi.com\nx-tc-action:describeinstances\n。 注意：content-type 必须和实际发送的相符合，有些编程语言网络库即使未指定也会自动添加 charset 值，如果签名时和发送时不一致，服务器会返回签名校验失败。
SignedHeaders	参与签名的头部信息，说明此次请求有哪些头部参与了签名，和 CanonicalHeaders 包含的头部内容是一一对应的。content-type 和 host 为必选头部。 拼接规则： 1. 头部 key 统一转成小写； 2. 多个头部 key（小写）按照 ASCII 升序进行拼接，并且以分号（;）分隔。 此示例为 content-type;host;x-tc-action
HashedRequestPayload	请求正文（payload，即 body，此示例为 {"Limit": 1, "Filters": [{"Values": ["\u672a\u547d\u540d"], "Name": "instance-name"}]}）的哈希值，计算伪代码为 Lowercase(HexEncode(Hash.SHA256(RequestPayload)))，即对 HTTP 请求正文做 SHA256 哈希，然后十六进制编码，最后编码串转换成小写字母。对于 GET 请求，RequestPayload 固定为空字符串。此示例计算结果是 35e9c5b0e3ae67532d3c9f17ead6c90222632e5b1ff7f6e89887f1398934f064。

根据以上规则，示例中得到的规范请求串如下：

```
POST
/

content-type:application/json; charset=utf-8
host:cvm.tencentcloudapi.com
x-tc-action:describeinstances

content-type;host;x-tc-action
35e9c5b0e3ae67532d3c9f17ead6c90222632e5b1ff7f6e89887f1398934f064
```

2. 拼接待签名字符串

按如下格式拼接待签名字符串：

```
StringToSign =
Algorithm + "\n" +
RequestTimestamp + "\n" +
CredentialScope + "\n" +
HashedCanonicalRequest
```

字段名称	解释
Algorithm	签名算法，目前固定为 TC3-HMAC-SHA256。

字段名称	解释
RequestTimestamp	请求时间戳，即请求头部的公共参数 X-TC-Timestamp 取值，取当前时间 UNIX 时间戳，精确到秒。此示例取值为 1551113065。
CredentialScope	凭证范围，格式为 Date/service/tc3_request，包含日期、所请求的服务和终止字符串（tc3_request）。**Date 为 UTC 标准时间的日期，取值需要和公共参数 X-TC-Timestamp 换算的 UTC 标准时间日期一致**；service 为产品名，必须与调用的产品域名一致。此示例计算结果是 2019-02-25/cvm/tc3_request。
HashedCanonicalRequest	前述步骤拼接所得规范请求串的哈希值，计算伪代码为 Lowercase(HexEncode(Hash.SHA256(CanonicalRequest)))。此示例计算结果是 7019a55be8395899b900fb5564e4200d984910f34794a27cb3fb7d10ff6a1e84。

注意：

1. **Date 必须从时间戳 X-TC-Timestamp 计算得到，且时区为 UTC+0。如果加入系统本地时区信息，例如东八区，将导致白天和晚上调用成功，但是凌晨时调用必定失败。假设时间戳为 1551113065，在东八区的时间是 2019-02-26 00:44:25，但是计算得到的 Date 取 UTC+0 的日期应为 2019-02-25，而不是 2019-02-26。**
2. **Timestamp 必须是当前系统时间，且需确保系统时间和标准时间是同步的，如果相差超过五分钟则必定失败。如果长时间不和标准时间同步，可能运行一段时间后，请求失败，返回签名过期错误。**

根据以上规则，示例中得到的待签名字符串如下：

```
TC3-HMAC-SHA256
1551113065
2019-02-25/cvm/tc3_request
7019a55be8395899b900fb5564e4200d984910f34794a27cb3fb7d10ff6a1e84
```

3. 计算签名

1）计算派生签名密钥，伪代码如下：

```
SecretKey = "Gu5t9xGARNpq86cd98joQYCN3*******"
SecretDate = HMAC_SHA256("TC3" + SecretKey, Date)
SecretService = HMAC_SHA256(SecretDate, Service)
SecretSigning = HMAC_SHA256(SecretService, "tc3_request")
```

派生出的密钥 SecretDate 、SecretService 和 SecretSigning 是二进制的数据，可能包含不可打印字符，将其转为十六进制字符串打印的输出分别为：f1cb4d518a0eda9d5cbbfdb7850983f1e603eeae484edea76e4dd8d8deb5556e，e7c609ce81bea53546bed2cc904778bef9ca14082e48e67883443ed64e227cd7，8aa8ab5755582f576e94bcfe383b8e29325b0ca90c3590d569221c6a63a091ed。

请注意，不同的编程语言，HMAC 库函数中参数顺序可能不一样，请以实际情况为准。此处的伪代码密钥参数 key 在前，消息参数 data 在后。通常标准库函数会提供二进制格式的返回值，也可能会提供打印友好的十六进制格式的返回值，此处使用的是二进制格式。

字段名称	解释
SecretKey	原始的 SecretKey，即 Gu5t9xGARNpq86cd98joQYCN3*******。
Date	即 Credential 中的 Date 字段信息。此示例取值为 2019-02-25。

字段名称	解释
Service	即 Credential 中的 Service 字段信息。此示例取值为 cvm。

2）计算签名，伪代码如下：

```
Signature = HexEncode(HMAC_SHA256(SecretSigning, StringToSign))
```

此示例计算结果是 be4f67d323c78ab9acb7395e43c0dbcf822a9cfac32fea2449a7bc7726b770a3 。

4. 拼接 Authorization

按如下格式拼接 Authorization：

```
Authorization =
Algorithm + ' ' +
'Credential=' + SecretId + '/' + CredentialScope + ', ' +
'SignedHeaders=' + SignedHeaders + ', ' +
'Signature=' + Signature
```

字段名称	解释
Algorithm	签名方法，固定为 TC3-HMAC-SHA256。
SecretId	密钥对中的 SecretId，即 AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******。
CredentialScope	见上文，凭证范围。此示例计算结果是 2019-02-25/cvm/tc3_request。
SignedHeaders	见上文，参与签名的头部信息。此示例取值为 content-type;host;x-tc-action。
Signature	签名值。此示例计算结果是 be4f67d323c78ab9acb7395e43c0dbcf822a9cfac32fea2449a7bc7726b770a3。

根据以上规则，示例中得到的值为：

```
TC3-HMAC-SHA256 Credential=AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******/2019-02-25/cvm/tc3_request, SignedHeaders=content-type;host;x-tc-action, Signature=be4f67d323c78ab9acb7395e43c0dbcf822a9cfac32fea2449a7bc7726b770a3
```

最终完整的调用信息如下：

```
POST https://cvm.tencentcloudapi.com/
Authorization: TC3-HMAC-SHA256 Credential=AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******/2019-02-25/cvm/tc3_request, SignedHeaders=content-type;host;x-tc-action, Signature=be4f67d323c78ab9acb7395e43c0dbcf822a9cfac32fea2449a7bc7726b770a3
Content-Type: application/json; charset=utf-8
Host: cvm.tencentcloudapi.com
X-TC-Action: DescribeInstances
X-TC-Version: 2017-03-12
X-TC-Timestamp: 1551113065
X-TC-Region: ap-guangzhou

{"Limit": 1, "Filters": [{"Values": ["\u672a\u547d\u540d"], "Name": "instance-name"}]}
```

> ⚠ 注意：
>
> 请求发送时的 HTTP 头部（Header）和请求体（Payload）必须和签名计算过程中的内容完全一致，否则会返回签名不一致错误。可以通过打印实际请求内容，网络抓包等方式对比排查。

签名演示

在实际调用 API 3.0 时，推荐使用配套的腾讯云 SDK 3.0，SDK 封装了签名的过程，开发时只关注产品提供的具体接口即可。详细信息参见 SDK 中心。当前支持的编程语言有：

- Python
- Java
- PHP
- Go
- NodeJS
- .NET
- C++
- Ruby

下面提供了不同产品的生成签名 demo，您可以找到对应的产品参考签名的生成：

- Signature Demo

为了更清楚地解释签名过程，下面以实际编程语言为例，将上述的签名过程完整实现。请求的域名、调用的接口和参数的取值都以上述签名过程为准，代码只为解释签名过程，并不具备通用性，实际开发请尽量使用 SDK。

Java

```
import java.nio.charset.Charset;
import java.nio.charset.StandardCharsets;
import java.security.MessageDigest;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.TimeZone;
import java.util.TreeMap;
import javax.crypto.Mac;
import javax.crypto.spec.SecretKeySpec;
import javax.xml.bind.DatatypeConverter;

public class TencentCloudAPITC3Demo {
private final static Charset UTF8 = StandardCharsets.UTF_8;
// 需要设置环境变量 TENCENTCLOUD_SECRET_ID，值为示例的 AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******
private final static String SECRET_ID = System.getenv("TENCENTCLOUD_SECRET_ID");
// 需要设置环境变量 TENCENTCLOUD_SECRET_KEY，值为示例的 Gu5t9xGARNpq86cd98joQYCN3*******
private final static String SECRET_KEY = System.getenv("TENCENTCLOUD_SECRET_KEY");
private final static String CT_JSON = "application/json; charset=utf-8";

public static byte[] hmac256(byte[] key, String msg) throws Exception {
Mac mac = Mac.getInstance("HmacSHA256");
SecretKeySpec secretKeySpec = new SecretKeySpec(key, mac.getAlgorithm());
mac.init(secretKeySpec);
```

```
return mac.doFinal(msg.getBytes(UTF8));
}

public static String sha256Hex(String s) throws Exception {
MessageDigest md = MessageDigest.getInstance("SHA-256");
byte[] d = md.digest(s.getBytes(UTF8));
return DatatypeConverter.printHexBinary(d).toLowerCase();
}

public static void main(String[] args) throws Exception {
String service = "cvm";
String host = "cvm.tencentcloudapi.com";
String region = "ap-guangzhou";
String action = "DescribeInstances";
String version = "2017-03-12";
String algorithm = "TC3-HMAC-SHA256";
String timestamp = "1551113065";
//String timestamp = String.valueOf(System.currentTimeMillis() / 1000);
SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");
// 注意时区，否则容易出错
sdf.setTimeZone(TimeZone.getTimeZone("UTC"));
String date = sdf.format(new Date(Long.valueOf(timestamp + "000")));

// ************ 步骤 1：拼接规范请求串 ************
String httpRequestMethod = "POST";
String canonicalUri = "/";
String canonicalQueryString = "";
String canonicalHeaders = "content-type:application/json; charset=utf-8\n"
+ "host:" + host + "\n" + "x-tc-action:" + action.toLowerCase() + "\n";
String signedHeaders = "content-type;host;x-tc-action";

String payload = "{\"Limit\": 1, \"Filters\": [{\"Values\": [\"\\u672a\\u547d\\u540d\"], \"Name\": \"instance-name\"}]}";
String hashedRequestPayload = sha256Hex(payload);
String canonicalRequest = httpRequestMethod + "\n" + canonicalUri + "\n" + canonicalQueryString + "\n"
+ canonicalHeaders + "\n" + signedHeaders + "\n" + hashedRequestPayload;
System.out.println(canonicalRequest);

// ************ 步骤 2：拼接待签名字符串 ************
String credentialScope = date + "/" + service + "/" + "tc3_request";
String hashedCanonicalRequest = sha256Hex(canonicalRequest);
String stringToSign = algorithm + "\n" + timestamp + "\n" + credentialScope + "\n" + hashedCanonicalRequest;
System.out.println(stringToSign);

// ************ 步骤 3：计算签名 ************
byte[] secretDate = hmac256(("TC3" + SECRET_KEY).getBytes(UTF8), date);
byte[] secretService = hmac256(secretDate, service);
byte[] secretSigning = hmac256(secretService, "tc3_request");
String signature = DatatypeConverter.printHexBinary(hmac256(secretSigning, stringToSign)).toLowerCase();
System.out.println(signature);
```

```
// ************* 步骤 4：拼接 Authorization *************
String authorization = algorithm + " " + "Credential=" + SECRET_ID + "/" + credentialScope + ", "
+ "SignedHeaders=" + signedHeaders + ", " + "Signature=" + signature;
System.out.println(authorization);

TreeMap<String, String> headers = new TreeMap<String, String>();
headers.put("Authorization", authorization);
headers.put("Content-Type", CT_JSON);
headers.put("Host", host);
headers.put("X-TC-Action", action);
headers.put("X-TC-Timestamp", timestamp);
headers.put("X-TC-Version", version);
headers.put("X-TC-Region", region);

StringBuilder sb = new StringBuilder();
sb.append("curl -X POST https://").append(host)
.append(" -H \"Authorization: ").append(authorization).append("\"")
.append(" -H \"Content-Type: application/json; charset=utf-8\"")
.append(" -H \"Host: ").append(host).append("\"")
.append(" -H \"X-TC-Action: ").append(action).append("\"")
.append(" -H \"X-TC-Timestamp: ").append(timestamp).append("\"")
.append(" -H \"X-TC-Version: ").append(version).append("\"")
.append(" -H \"X-TC-Region: ").append(region).append("\"")
.append(" -d '").append(payload).append("'");
System.out.println(sb.toString());
}
}
```

Python

```
# -*- coding: utf-8 -*-
import hashlib, hmac, json, os, sys, time
from datetime import datetime

# 密钥参数
# 需要设置环境变量 TENCENTCLOUD_SECRET_ID，值为示例的 AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******
secret_id = os.environ.get("TENCENTCLOUD_SECRET_ID")
# 需要设置环境变量 TENCENTCLOUD_SECRET_KEY，值为示例的 Gu5t9xGARNpq86cd98joQYCN3*******
secret_key = os.environ.get("TENCENTCLOUD_SECRET_KEY")

service = "cvm"
host = "cvm.tencentcloudapi.com"
endpoint = "https://" + host
region = "ap-guangzhou"
action = "DescribeInstances"
version = "2017-03-12"
algorithm = "TC3-HMAC-SHA256"
#timestamp = int(time.time())
timestamp = 1551113065
```

```
date = datetime.utcfromtimestamp(timestamp).strftime("%Y-%m-%d")
params = {"Limit": 1, "Filters": [{"Values": [u"未命名"], "Name": "instance-name"}]}

# ************* 步骤 1：拼接规范请求串 *************
http_request_method = "POST"
canonical_uri = "/"
canonical_querystring = ""
ct = "application/json; charset=utf-8"
payload = json.dumps(params)
canonical_headers = "content-type:%s\nhost:%s\nx-tc-action:%s\n" % (ct, host, action.lower())
signed_headers = "content-type;host;x-tc-action"
hashed_request_payload = hashlib.sha256(payload.encode("utf-8")).hexdigest()
canonical_request = (http_request_method + "\n" +
canonical_uri + "\n" +
canonical_querystring + "\n" +
canonical_headers + "\n" +
signed_headers + "\n" +
hashed_request_payload)
print(canonical_request)

# ************* 步骤 2：拼接待签名字符串 *************
credential_scope = date + "/" + service + "/" + "tc3_request"
hashed_canonical_request = hashlib.sha256(canonical_request.encode("utf-8")).hexdigest()
string_to_sign = (algorithm + "\n" +
str(timestamp) + "\n" +
credential_scope + "\n" +
hashed_canonical_request)
print(string_to_sign)


# ************* 步骤 3：计算签名 *************
# 计算签名摘要函数
def sign(key, msg):
return hmac.new(key, msg.encode("utf-8"), hashlib.sha256).digest()
secret_date = sign(("TC3" + secret_key).encode("utf-8"), date)
secret_service = sign(secret_date, service)
secret_signing = sign(secret_service, "tc3_request")
signature = hmac.new(secret_signing, string_to_sign.encode("utf-8"), hashlib.sha256).hexdigest()
print(signature)

# ************* 步骤 4：拼接 Authorization *************
authorization = (algorithm + " " +
"Credential=" + secret_id + "/" + credential_scope + ", " +
"SignedHeaders=" + signed_headers + ", " +
"Signature=" + signature)
print(authorization)

print('curl -X POST ' + endpoint
+ ' -H "Authorization: ' + authorization + '"'
+ ' -H "Content-Type: application/json; charset=utf-8"'
```

```
+ ' -H "Host: ' + host + '"'
+ ' -H "X-TC-Action: ' + action + '"'
+ ' -H "X-TC-Timestamp: ' + str(timestamp) + '"'
+ ' -H "X-TC-Version: ' + version + '"'
+ ' -H "X-TC-Region: ' + region + '"'
+ " -d '" + payload + "'")
```

Golang

```
package main

import (
"crypto/hmac"
"crypto/sha256"
"encoding/hex"
"fmt"
"os"
"strings"
"time"
)

func sha256hex(s string) string {
b := sha256.Sum256([]byte(s))
return hex.EncodeToString(b[:])
}

func hmacsha256(s, key string) string {
hashed := hmac.New(sha256.New, []byte(key))
hashed.Write([]byte(s))
return string(hashed.Sum(nil))
}

func main() {
// 需要设置环境变量 TENCENTCLOUD_SECRET_ID，值为示例的 AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******
secretId := os.Getenv("TENCENTCLOUD_SECRET_ID")
// 需要设置环境变量 TENCENTCLOUD_SECRET_KEY，值为示例的 Gu5t9xGARNpq86cd98joQYCN3*******
secretKey := os.Getenv("TENCENTCLOUD_SECRET_KEY")
host := "cvm.tencentcloudapi.com"
algorithm := "TC3-HMAC-SHA256"
service := "cvm"
version := "2017-03-12"
action := "DescribeInstances"
region := "ap-guangzhou"
//var timestamp int64 = time.Now().Unix()
var timestamp int64 = 1551113065

// step 1: build canonical request string
httpRequestMethod := "POST"
canonicalURI := "/"
```

```
canonicalQueryString := ""
canonicalHeaders := fmt.Sprintf("content-type:%s\nhost:%s\nx-tc-action:%s\n",
"application/json; charset=utf-8", host, strings.ToLower(action))
signedHeaders := "content-type;host;x-tc-action"
payload := `{"Limit": 1, "Filters": [{"Values": ["\u672a\u547d\u540d"], "Name": "instance-name"}]}`
hashedRequestPayload := sha256hex(payload)
canonicalRequest := fmt.Sprintf("%s\n%s\n%s\n%s\n%s\n%s",
httpRequestMethod,
canonicalURI,
canonicalQueryString,
canonicalHeaders,
signedHeaders,
hashedRequestPayload)
fmt.Println(canonicalRequest)

// step 2: build string to sign
date := time.Unix(timestamp, 0).UTC().Format("2006-01-02")
credentialScope := fmt.Sprintf("%s/%s/tc3_request", date, service)
hashedCanonicalRequest := sha256hex(canonicalRequest)
string2sign := fmt.Sprintf("%s\n%d\n%s\n%s",
algorithm,
timestamp,
credentialScope,
hashedCanonicalRequest)
fmt.Println(string2sign)

// step 3: sign string
secretDate := hmacsha256(date, "TC3"+secretKey)
secretService := hmacsha256(service, secretDate)
secretSigning := hmacsha256("tc3_request", secretService)
signature := hex.EncodeToString([]byte(hmacsha256(string2sign, secretSigning)))
fmt.Println(signature)

// step 4: build authorization
authorization := fmt.Sprintf("%s Credential=%s/%s, SignedHeaders=%s, Signature=%s",
algorithm,
secretId,
credentialScope,
signedHeaders,
signature)
fmt.Println(authorization)

curl := fmt.Sprintf(`curl -X POST https://%s\
-H "Authorization: %s"\
-H "Content-Type: application/json; charset=utf-8"\
-H "Host: %s" -H "X-TC-Action: %s"\
-H "X-TC-Timestamp: %d"\
-H "X-TC-Version: %s"\
-H "X-TC-Region: %s"\
-d '%s'`, host, authorization, host, action, timestamp, version, region, payload)
```

```
fmt.Println(curl)
}
```

PHP

```
<?php
// 需要设置环境变量 TENCENTCLOUD_SECRET_ID，值为示例的 AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******
$secretId = getenv("TENCENTCLOUD_SECRET_ID");
// 需要设置环境变量 TENCENTCLOUD_SECRET_KEY，值为示例的 Gu5t9xGARNpq86cd98joQYCN3*******
$secretKey = getenv("TENCENTCLOUD_SECRET_KEY");
$host = "cvm.tencentcloudapi.com";
$service = "cvm";
$version = "2017-03-12";
$action = "DescribeInstances";
$region = "ap-guangzhou";
// $timestamp = time();
$timestamp = 1551113065;
$algorithm = "TC3-HMAC-SHA256";

// step 1: build canonical request string
$httpRequestMethod = "POST";
$canonicalUri = "/";
$canonicalQueryString = "";
$canonicalHeaders = implode("\n", [
"content-type:application/json; charset=utf-8",
"host:".$host,
"x-tc-action:".strtolower($action),
""
]);
$signedHeaders = implode(";", [
"content-type",
"host",
"x-tc-action",
]);
$payload = '{"Limit": 1, "Filters": [{"Values": ["\u672a\u547d\u540d"], "Name": "instance-name"}]}';
$hashedRequestPayload = hash("SHA256", $payload);
$canonicalRequest = $httpRequestMethod."\n"
.$canonicalUri."\n"
.$canonicalQueryString."\n"
.$canonicalHeaders."\n"
.$signedHeaders."\n"
.$hashedRequestPayload;
echo $canonicalRequest.PHP_EOL;

// step 2: build string to sign
$date = gmdate("Y-m-d", $timestamp);
$credentialScope = $date."/".$service."/tc3_request";
$hashedCanonicalRequest = hash("SHA256", $canonicalRequest);
$stringToSign = $algorithm."\n"
```

```
.$timestamp."\n"
.$credentialScope."\n"
.$hashedCanonicalRequest;
echo $stringToSign.PHP_EOL;

// step 3: sign string
$secretDate = hash_hmac("SHA256", $date, "TC3".$secretKey, true);
$secretService = hash_hmac("SHA256", $service, $secretDate, true);
$secretSigning = hash_hmac("SHA256", "tc3_request", $secretService, true);
$signature = hash_hmac("SHA256", $stringToSign, $secretSigning);
echo $signature.PHP_EOL;

// step 4: build authorization
$authorization = $algorithm
." Credential=".$secretId."/".$credentialScope
.", SignedHeaders=".$signedHeaders.", Signature=".$signature;
echo $authorization.PHP_EOL;

$curl = "curl -X POST https://".$host
.' -H "Authorization: '.$authorization.'"'
.' -H "Content-Type: application/json; charset=utf-8"'
.' -H "Host: '.$host.'"'
.' -H "X-TC-Action: '.$action.'"'
.' -H "X-TC-Timestamp: ' .$timestamp.'"'
.' -H "X-TC-Version: '.$version.'"'
.' -H "X-TC-Region: '.$region.'"'
." -d '".$payload."'";
echo $curl.PHP_EOL;
```

Ruby

```
# -*- coding: UTF-8 -*-
# require ruby>=2.3.0
require 'digest'
require 'json'
require 'time'
require 'openssl'

# 密钥参数
# 需要设置环境变量 TENCENTCLOUD_SECRET_ID，值为示例的 AKIDz8krbsJ5yKBZQpn74WFkmLPx3******
secret_id = ENV["TENCENTCLOUD_SECRET_ID"]
# 需要设置环境变量 TENCENTCLOUD_SECRET_KEY，值为示例的 Gu5t9xGARNpq86cd98joQYCN3******
secret_key = ENV["TENCENTCLOUD_SECRET_KEY"]

service = 'cvm'
host = 'cvm.tencentcloudapi.com'
endpoint = 'https://' + host
region = 'ap-guangzhou'
action = 'DescribeInstances'
```

```
version = '2017-03-12'
algorithm = 'TC3-HMAC-SHA256'
# timestamp = Time.now.to_i
timestamp = 1551113065
date = Time.at(timestamp).utc.strftime('%Y-%m-%d')

# ************* 步骤 1：拼接规范请求串 *************
http_request_method = 'POST'
canonical_uri = '/'
canonical_querystring = ''
canonical_headers = "content-type:application/json; charset=utf-8\nhost:#{host}\nx-tc-action:#{action.downcase}\n"
signed_headers = 'content-type;host;x-tc-action'
# params = { 'Limit' => 1, 'Filters' => [{ 'Name' => 'instance-name', 'Values' => ['未命名'] }] }
# payload = JSON.generate(params, { 'ascii_only' => true, 'space' => ' ' })
# json will generate in random order, to get specified result in example, we hard-code it here.
payload = '{"Limit": 1, "Filters": [{"Values": ["\u672a\u547d\u540d"], "Name": "instance-name"}]}'
hashed_request_payload = Digest::SHA256.hexdigest(payload)
canonical_request = [
http_request_method,
canonical_uri,
canonical_querystring,
canonical_headers,
signed_headers,
hashed_request_payload,
].join("\n")

puts canonical_request

# ************* 步骤 2：拼接待签名字符串 *************
credential_scope = date + '/' + service + '/' + 'tc3_request'
hashed_request_payload = Digest::SHA256.hexdigest(canonical_request)
string_to_sign = [
algorithm,
timestamp.to_s,
credential_scope,
hashed_request_payload,
].join("\n")
puts string_to_sign

# ************* 步骤 3：计算签名 *************
digest = OpenSSL::Digest.new('sha256')
secret_date = OpenSSL::HMAC.digest(digest, 'TC3' + secret_key, date)
secret_service = OpenSSL::HMAC.digest(digest, secret_date, service)
secret_signing = OpenSSL::HMAC.digest(digest, secret_service, 'tc3_request')
signature = OpenSSL::HMAC.hexdigest(digest, secret_signing, string_to_sign)
puts signature

# ************* 步骤 4：拼接 Authorization *************
authorization = "#{algorithm} Credential=#{secret_id}/#{credential_scope}, SignedHeaders=#{signed_headers}, Signature=
#{signature}"
```

```
puts authorization

puts 'curl -X POST ' + endpoint \
+ ' -H "Authorization: ' + authorization + '"' \
+ ' -H "Content-Type: application/json; charset=utf-8"' \
+ ' -H "Host: ' + host + '"' \
+ ' -H "X-TC-Action: ' + action + '"' \
+ ' -H "X-TC-Timestamp: ' + timestamp.to_s + '"' \
+ ' -H "X-TC-Version: ' + version + '"' \
+ ' -H "X-TC-Region: ' + region + '"' \
+ " -d '" + payload + "'"
```

DotNet

```
using System;
using System.Collections.Generic;
using System.Security.Cryptography;
using System.Text;

public class Application
{
public static string SHA256Hex(string s)
{
using (SHA256 algo = SHA256.Create())
{
byte[] hashbytes = algo.ComputeHash(Encoding.UTF8.GetBytes(s));
StringBuilder builder = new StringBuilder();
for (int i = 0; i < hashbytes.Length; ++i)
{
builder.Append(hashbytes[i].ToString("x2"));
}
return builder.ToString();
}
}

public static byte[] HmacSHA256(byte[] key, byte[] msg)
{
using (HMACSHA256 mac = new HMACSHA256(key))
{
return mac.ComputeHash(msg);
}
}

public static Dictionary<String, String> BuildHeaders(string secretid,
string secretkey, string service, string endpoint, string region,
string action, string version, DateTime date, string requestPayload)
{
string datestr = date.ToString("yyyy-MM-dd");
DateTime startTime = new DateTime(1970, 1, 1, 0, 0, 0, 0, DateTimeKind.Utc);
```

```
long requestTimestamp = (long)Math.Round((date - startTime).TotalMilliseconds, MidpointRounding.AwayFromZero) / 1000;
// ************* 步骤 1：拼接规范请求串 *************
string algorithm = "TC3-HMAC-SHA256";
string httpRequestMethod = "POST";
string canonicalUri = "/";
string canonicalQueryString = "";
string contentType = "application/json";
string canonicalHeaders = "content-type:" + contentType + "; charset=utf-8\n"
+ "host:" + endpoint + "\n"
+ "x-tc-action:" + action.ToLower() + "\n";
string signedHeaders = "content-type;host;x-tc-action";
string hashedRequestPayload = SHA256Hex(requestPayload);
string canonicalRequest = httpRequestMethod + "\n"
+ canonicalUri + "\n"
+ canonicalQueryString + "\n"
+ canonicalHeaders + "\n"
+ signedHeaders + "\n"
+ hashedRequestPayload;
Console.WriteLine(canonicalRequest);

// ************* 步骤 2：拼接待签名字符串 *************
string credentialScope = datestr + "/" + service + "/" + "tc3_request";
string hashedCanonicalRequest = SHA256Hex(canonicalRequest);
string stringToSign = algorithm + "\n"
+ requestTimestamp.ToString() + "\n"
+ credentialScope + "\n"
+ hashedCanonicalRequest;
Console.WriteLine(stringToSign);

// ************* 步骤 3：计算签名 *************
byte[] tc3SecretKey = Encoding.UTF8.GetBytes("TC3" + secretkey);
byte[] secretDate = HmacSHA256(tc3SecretKey, Encoding.UTF8.GetBytes(datestr));
byte[] secretService = HmacSHA256(secretDate, Encoding.UTF8.GetBytes(service));
byte[] secretSigning = HmacSHA256(secretService, Encoding.UTF8.GetBytes("tc3_request"));
byte[] signatureBytes = HmacSHA256(secretSigning, Encoding.UTF8.GetBytes(stringToSign));
string signature = BitConverter.ToString(signatureBytes).Replace("-", "").ToLower();
Console.WriteLine(signature);

// ************* 步骤 4：拼接 Authorization *************
string authorization = algorithm + " "
+ "Credential=" + secretid + "/" + credentialScope + ", "
+ "SignedHeaders=" + signedHeaders + ", "
+ "Signature=" + signature;
Console.WriteLine(authorization);

Dictionary<string, string> headers = new Dictionary<string, string>();
headers.Add("Authorization", authorization);
headers.Add("Host", endpoint);
headers.Add("Content-Type", contentType + "; charset=utf-8");
headers.Add("X-TC-Timestamp" , requestTimestamp.ToString());
```

```
headers.Add("X-TC-Version", version);
headers.Add("X-TC-Action", action);
headers.Add("X-TC-Region", region);
return headers;
}
public static void Main(string[] args)
{
// 密钥参数
// 需要设置环境变量 TENCENTCLOUD_SECRET_ID，值为示例的 AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******
string SECRET_ID = Environment.GetEnvironmentVariable("TENCENTCLOUD_SECRET_ID");
// 需要设置环境变量 TENCENTCLOUD_SECRET_KEY，值为示例的 Gu5t9xGARNpq86cd98joQYCN3*******
string SECRET_KEY = Environment.GetEnvironmentVariable("TENCENTCLOUD_SECRET_KEY");

string service = "cvm";
string endpoint = "cvm.tencentcloudapi.com";
string region = "ap-guangzhou";
string action = "DescribeInstances";
string version = "2017-03-12";

// 此处由于示例规范的原因，采用时间戳2019-02-26 00:44:25，此参数作为示例，如果在项目中，您应当使用：
// DateTime date = DateTime.UtcNow;
// 注意时区，建议此时间统一采用UTC时间戳，否则容易出错
DateTime date = new DateTime(1970, 1, 1, 0, 0, 0, 0, DateTimeKind.Utc).AddSeconds(1551113065);
string requestPayload = "{\"Limit\": 1, \"Filters\": [{\"Values\": [\"\\u672a\\u547d\\u540d\"], \"Name\": \"instance-name\"}]}";

Dictionary<string, string> headers = BuildHeaders(SECRET_ID, SECRET_KEY, service
, endpoint, region, action, version, date, requestPayload);

Console.WriteLine("POST https://cvm.tencentcloudapi.com");
foreach (KeyValuePair<string, string> kv in headers)
{
Console.WriteLine(kv.Key + ": " + kv.Value);
}
Console.WriteLine();
Console.WriteLine(requestPayload);
}
}
```

NodeJS

```
const crypto = require('crypto');

function sha256(message, secret = '', encoding) {
const hmac = crypto.createHmac('sha256', secret)
return hmac.update(message).digest(encoding)
}

function getHash(message, encoding = 'hex') {
const hash = crypto.createHash('sha256')
```

```
return hash.update(message).digest(encoding)
}

function getDate(timestamp) {
const date = new Date(timestamp * 1000)
const year = date.getUTCFullYear()
const month = ('0' + (date.getUTCMonth() + 1)).slice(-2)
const day = ('0' + date.getUTCDate()).slice(-2)
return `${year}-${month}-${day}`
}

function main(){
// 密钥参数
// 需要设置环境变量 TENCENTCLOUD_SECRET_ID，值为示例的 AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******
const SECRET_ID = process.env.TENCENTCLOUD_SECRET_ID
// 需要设置环境变量 TENCENTCLOUD_SECRET_KEY，值为示例的 Gu5t9xGARNpq86cd98joQYCN3*******
const SECRET_KEY = process.env.TENCENTCLOUD_SECRET_KEY

const endpoint = "cvm.tencentcloudapi.com"
const service = "cvm"
const region = "ap-guangzhou"
const action = "DescribeInstances"
const version = "2017-03-12"
//const timestamp = getTime()
const timestamp = 1551113065
//时间处理, 获取世界时间日期
const date = getDate(timestamp)

// ************* 步骤 1：拼接规范请求串 *************
const payload = "{\"Limit\": 1, \"Filters\": [{\"Values\": [\"\\u672a\\u547d\\u540d\"], \"Name\": \"instance-name\"}]}"

const hashedRequestPayload = getHash(payload);
const httpRequestMethod = "POST"
const canonicalUri = "/"
const canonicalQueryString = ""
const canonicalHeaders = "content-type:application/json; charset=utf-8\n"
+ "host:" + endpoint + "\n"
+ "x-tc-action:" + action.toLowerCase() + "\n"
const signedHeaders = "content-type;host;x-tc-action"

const canonicalRequest = httpRequestMethod + "\n"
+ canonicalUri + "\n"
+ canonicalQueryString + "\n"
+ canonicalHeaders + "\n"
+ signedHeaders + "\n"
+ hashedRequestPayload
console.log(canonicalRequest)

// ************* 步骤 2：拼接待签名字符串 *************
const algorithm = "TC3-HMAC-SHA256"
```

```
const hashedCanonicalRequest = getHash(canonicalRequest);
const credentialScope = date + "/" + service + "/" + "tc3_request"
const stringToSign = algorithm + "\n" +
timestamp + "\n" +
credentialScope + "\n" +
hashedCanonicalRequest
console.log(stringToSign)

// ************* 步骤 3：计算签名 *************
const kDate = sha256(date, 'TC3' + SECRET_KEY)
const kService = sha256(service, kDate)
const kSigning = sha256('tc3_request', kService)
const signature = sha256(stringToSign, kSigning, 'hex')
console.log(signature)

// ************* 步骤 4：拼接 Authorization *************
const authorization = algorithm + " " +
"Credential=" + SECRET_ID + "/" + credentialScope + ", " +
"SignedHeaders=" + signedHeaders + ", " +
"Signature=" + signature
console.log(authorization)

const curlcmd = 'curl -X POST ' + "https://" + endpoint
+ ' -H "Authorization: ' + authorization + '"'
+ ' -H "Content-Type: application/json; charset=utf-8"'
+ ' -H "Host: ' + endpoint + '"'
+ ' -H "X-TC-Action: ' + action + '"'
+ ' -H "X-TC-Timestamp: ' + timestamp.toString() + '"'
+ ' -H "X-TC-Version: ' + version + '"'
+ ' -H "X-TC-Region: ' + region + '"'
+ " -d '" + payload + "'"
console.log(curlcmd)
}
main()
```

C++

```
#include <algorithm>
#include <cstdlib>
#include <iostream>
#include <iomanip>
#include <sstream>
#include <string>
#include <stdio.h>
#include <time.h>
#include <openssl/sha.h>
#include <openssl/hmac.h>

using namespace std;
```

```
string get_data(int64_t &timestamp)
{
string utcDate;
char buff[20] = {0};
// time_t timenow;
struct tm sttime;
sttime = *gmtime(&timestamp);
strftime(buff, sizeof(buff), "%Y-%m-%d", &sttime);
utcDate = string(buff);
return utcDate;
}

string int2str(int64_t n)
{
std::stringstream ss;
ss << n;
return ss.str();
}

string sha256Hex(const string &str)
{
char buf[3];
unsigned char hash[SHA256_DIGEST_LENGTH];
SHA256_CTX sha256;
SHA256_Init(&sha256);
SHA256_Update(&sha256, str.c_str(), str.size());
SHA256_Final(hash, &sha256);
std::string NewString = "";
for(int i = 0; i < SHA256_DIGEST_LENGTH; i++)
{
snprintf(buf, sizeof(buf), "%02x", hash[i]);
NewString = NewString + buf;
}
return NewString;
}

string HmacSha256(const string &key, const string &input)
{
unsigned char hash[32];

HMAC_CTX *h;
#if OPENSSL_VERSION_NUMBER < 0x10100000L
HMAC_CTX hmac;
HMAC_CTX_init(&hmac);
h = &hmac;
#else
h = HMAC_CTX_new();
#endif
```

```
HMAC_Init_ex(h, &key[0], key.length(), EVP_sha256(), NULL);
HMAC_Update(h, ( unsigned char* )&input[0], input.length());
unsigned int len = 32;
HMAC_Final(h, hash, &len);

#if OPENSSL_VERSION_NUMBER < 0x10100000L
HMAC_CTX_cleanup(h);
#else
HMAC_CTX_free(h);
#endif

std::stringstream ss;
ss << std::setfill('0');
for (int i = 0; i < len; i++)
{
ss << hash[i];
}

return (ss.str());
}

string HexEncode(const string &input)
{
static const char* const lut = "0123456789abcdef";
size_t len = input.length();

string output;
output.reserve(2 * len);
for (size_t i = 0; i < len; ++i)
{
const unsigned char c = input[i];
output.push_back(lut[c >> 4]);
output.push_back(lut[c & 15]);
}
return output;
}

int main()
{
// 密钥参数
// 需要设置环境变量 TENCENTCLOUD_SECRET_ID，值为示例的 AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******
string SECRET_ID = getenv("TENCENTCLOUD_SECRET_ID");
// 需要设置环境变量 TENCENTCLOUD_SECRET_KEY，值为示例的 Gu5t9xGARNpq86cd98joQYCN3*******
string SECRET_KEY = getenv("TENCENTCLOUD_SECRET_KEY");

string service = "cvm";
string host = "cvm.tencentcloudapi.com";
string region = "ap-guangzhou";
string action = "DescribeInstances";
string version = "2017-03-12";
```

```
int64_t timestamp = 1551113065;
string date = get_data(timestamp);

// ************* 步骤 1：拼接规范请求串 *************
string httpRequestMethod = "POST";
string canonicalUri = "/";
string canonicalQueryString = "";
string lower = action;
std::transform(action.begin(), action.end(), lower.begin(), ::tolower);
string canonicalHeaders = string("content-type:application/json; charset=utf-8\n")
+ "host:" + host + "\n"
+ "x-tc-action:" + lower + "\n";
string signedHeaders = "content-type;host;x-tc-action";
string payload = "{\"Limit\": 1, \"Filters\": [{\"Values\": [\"\\u672a\\u547d\\u540d\"], \"Name\": \"instance-name\"}]}";
string hashedRequestPayload = sha256Hex(payload);
string canonicalRequest = httpRequestMethod + "\n"
+ canonicalUri + "\n"
+ canonicalQueryString + "\n"
+ canonicalHeaders + "\n"
+ signedHeaders + "\n"
+ hashedRequestPayload;
cout << canonicalRequest << endl;

// ************* 步骤 2：拼接待签名字符串 *************
string algorithm = "TC3-HMAC-SHA256";
string RequestTimestamp = int2str(timestamp);
string credentialScope = date + "/" + service + "/" + "tc3_request";
string hashedCanonicalRequest = sha256Hex(canonicalRequest);
string stringToSign = algorithm + "\n" + RequestTimestamp + "\n" + credentialScope + "\n" + hashedCanonicalRequest;
cout << stringToSign << endl;

// ************* 步骤 3：计算签名 ***************
string kKey = "TC3" + SECRET_KEY;
string kDate = HmacSha256(kKey, date);
string kService = HmacSha256(kDate, service);
string kSigning = HmacSha256(kService, "tc3_request");
string signature = HexEncode(HmacSha256(kSigning, stringToSign));
cout << signature << endl;

// ************* 步骤 4：拼接 Authorization *************
string authorization = algorithm + " " + "Credential=" + SECRET_ID + "/" + credentialScope + ", "
+ "SignedHeaders=" + signedHeaders + ", " + "Signature=" + signature;
cout << authorization << endl;

string curlcmd = "curl -X POST https://" + host + "\n"
+ " -H \"Authorization: " + authorization + "\"\n"
+ " -H \"Content-Type: application/json; charset=utf-8\"" + "\n"
+ " -H \"Host: " + host + "\"\n"
+ " -H \"X-TC-Action: " + action + "\"\n"
+ " -H \"X-TC-Timestamp: "  + RequestTimestamp + "\"\n"
```

```
+ " -H \"X-TC-Version: " + version + "\"\n"
+ " -H \"X-TC-Region: " + region + "\"\n"
+ " -d '" + payload + "\'";
cout << curlcmd << endl;
return 0;
};
```

C

```
#include <ctype.h>
#include <string.h>
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <stdint.h>
#include <openssl/sha.h>
#include <openssl/hmac.h>

void get_utc_date(int64_t timestamp, char* utc, int len)
{
// time_t timenow;
struct tm sttime;
sttime = *gmtime(&timestamp);
strftime(utc, len, "%Y-%m-%d", &sttime);
}

void sha256_hex(const char* str, char* result)
{
char buf[3];
unsigned char hash[SHA256_DIGEST_LENGTH];
SHA256_CTX sha256;
SHA256_Init(&sha256);
SHA256_Update(&sha256, str, strlen(str));
SHA256_Final(hash, &sha256);
for(int i = 0; i < SHA256_DIGEST_LENGTH; i++)
{
snprintf(buf, sizeof(buf), "%02x", hash[i]);
strcat(result, buf);
}
}

void hmac_sha256(char* key, const char* input, char* result)
{
unsigned char hash[32];

HMAC_CTX *h;
#if OPENSSL_VERSION_NUMBER < 0x10100000L
HMAC_CTX hmac;
HMAC_CTX_init(&hmac);
```

```
h = &hmac;
#else
h = HMAC_CTX_new();
#endif

HMAC_Init_ex(h, key, strlen(key), EVP_sha256(), NULL);
HMAC_Update(h, ( unsigned char* )input, strlen(input));
unsigned int len = 32;
HMAC_Final(h, hash, &len);

#if OPENSSL_VERSION_NUMBER < 0x10100000L
HMAC_CTX_cleanup(h);
#else
HMAC_CTX_free(h);
#endif
strncpy(result, (const char*)hash, len);
}

void hex_encode(const char* input, char* output)
{
static const char* const lut = "0123456789abcdef";
size_t len = strlen(input);
char add_out[128] = {0};
char temp[2] = {0};
for (size_t i = 0; i < len; ++i)
{
const unsigned char c = input[i];
temp[0] = lut[c >> 4];
strcat(add_out, temp);
temp[0] = lut[c & 15];
strcat(add_out, temp);
}
strncpy(output, add_out, 128);
}

void lowercase(const char * src, char * dst)
{
for (int i = 0; src[i]; i++)
{
dst[i] = tolower(src[i]);
}
}

int main()
{
// 密钥参数
// 需要设置环境变量 TENCENTCLOUD_SECRET_ID，值为示例的 AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******
const char* SECRET_ID = getenv("TENCENTCLOUD_SECRET_ID");
// 需要设置环境变量 TENCENTCLOUD_SECRET_KEY，值为示例的 Gu5t9xGARNpq86cd98joQYCN3*******
const char* SECRET_KEY = getenv("TENCENTCLOUD_SECRET_KEY");
```

```
const char* service = "cvm";
const char* host = "cvm.tencentcloudapi.com";
const char* region = "ap-guangzhou";
const char* action = "DescribeInstances";
const char* version = "2017-03-12";
int64_t timestamp = 1551113065;
char date[20] = {0};
get_utc_date(timestamp, date, sizeof(date));

// ************ 步骤 1：拼接规范请求串 ************
const char* http_request_method = "POST";
const char* canonical_uri = "/";
const char* canonical_query_string = "";
char canonical_headers[100] = {"content-type:application/json; charset=utf-8\nhost:"};
strcat(canonical_headers, host);
strcat(canonical_headers, "\nx-tc-action:");
char value[100] = {0};
lowercase(action, value);
strcat(canonical_headers, value);
strcat(canonical_headers, "\n");
const char* signed_headers = "content-type;host;x-tc-action";
const char* payload = "{\"Limit\": 1, \"Filters\": [{\"Values\": [\"\\u672a\\u547d\\u540d\"], \"Name\": \"instance-name\"}]}";
char hashed_request_payload[100] = {0};
sha256_hex(payload, hashed_request_payload);

char canonical_request[256] = {0};
sprintf(canonical_request, "%s\n%s\n%s\n%s\n%s\n%s", http_request_method,
canonical_uri, canonical_query_string, canonical_headers,
signed_headers, hashed_request_payload);
printf("%s\n", canonical_request);


// ************ 步骤 2：拼接待签名字符串 ************
const char* algorithm = "TC3-HMAC-SHA256";
char request_timestamp[16] = {0};
sprintf(request_timestamp, "%d", timestamp);
char credential_scope[64] = {0};
strcat(credential_scope, date);
sprintf(credential_scope, "%s/%s/tc3_request", date, service);
char hashed_canonical_request[100] = {0};
sha256_hex(canonical_request, hashed_canonical_request);
char string_to_sign[256] = {0};
sprintf(string_to_sign, "%s\n%s\n%s\n%s", algorithm, request_timestamp,
credential_scope, hashed_canonical_request);
printf("%s\n", string_to_sign);

// ************ 步骤 3：计算签名 ***************
char k_key[64] = {0};
sprintf(k_key, "%s%s", "TC3", SECRET_KEY);
char k_date[64] = {0};
```

```
hmac_sha256(k_key, date, k_date);
char k_service[64] = {0};
hmac_sha256(k_date, service, k_service);
char k_signing[64] = {0};
hmac_sha256(k_service, "tc3_request", k_signing);
char k_hmac_sha_sign[64] = {0};
hmac_sha256(k_signing, string_to_sign, k_hmac_sha_sign);

char signature[128] = {0};
hex_encode(k_hmac_sha_sign, signature);
printf("%s\n", signature);

// ************* 步骤 4：拼接 Authorization *************
char authorization[512] = {0};
sprintf(authorization, "%s Credential=%s/%s, SignedHeaders=%s, Signature=%s",
algorithm, SECRET_ID, credential_scope, signed_headers, signature);
printf("%s\n", authorization);

char curlcmd[10240] = {0};
sprintf(curlcmd, "curl -X POST https://%s\n \
-H \"Authorization: %s\"\n \
-H \"Content-Type: application/json; charset=utf-8\"\n \
-H \"Host: %s\"\n \
-H \"X-TC-Action: %s\"\n \
-H \"X-TC-Timestamp: %s\"\n \
-H \"X-TC-Version: %s\"\n \
-H \"X-TC-Region: %s\"\n \
-d \'%s\'",
host, authorization, host, action, request_timestamp, version, region, payload);
printf("%s\n", curlcmd);
return 0;
}
```

其他语言

- Lua: GitHub
- Swift: GitHub
- Dart: GitHub
- Shell(Bash): GitHub

签名失败

存在以下签名失败的错误码，请根据实际情况处理。

错误码	错误描述
AuthFailure.SignatureExpire	签名过期。Timestamp 与服务器接收到请求的时间相差不得超过五分钟。
AuthFailure.SecretIdNotFound	密钥不存在。请到控制台查看密钥是否被禁用，是否少复制了字符或者多了字符。
AuthFailure.SignatureFailure	签名错误。可能是签名计算错误，或者签名与实际发送的内容不相符合，也有可能是密钥 SecretKey 错误导致的。

错误码	错误描述
AuthFailure.TokenFailure	临时证书 Token 错误。
AuthFailure.InvalidSecretId	密钥非法（不是云 API 密钥类型）。

签名方法

最近更新时间：2024-01-05 01:17:52

签名方法 v1 简单易用，但是功能和安全性都不如签名方法 v3，推荐使用签名方法 v3。

首次接触，建议使用 API Explorer 中的“签名串生成”功能，选择签名版本为“API 3.0 签名 v1”，可以生成签名过程进行验证，并提供了部分编程语言的签名示例，也可直接生成 SDK 代码。推荐使用腾讯云 API 配套的 8 种常见的编程语言 SDK，已经封装了签名和请求过程，均已开源，支持 Python、Java、PHP、Go、NodeJS、.NET、C++、Ruby。

推荐使用 API Explorer　点击调试

您可以通过 API Explorer 的【签名串生成】模块查看每个接口签名的生成过程。

腾讯云 API 会对每个访问请求进行身份验证，即每个请求都需要在公共请求参数中包含签名信息（Signature）以验证请求者身份。
签名信息由安全凭证生成，安全凭证包括 SecretId 和 SecretKey；若用户还没有安全凭证，请前往 云API密钥页面 申请，否则无法调用云 API 接口。

1. 申请安全凭证

在第一次使用云 API 之前，请前往 云 API 密钥页面 申请安全凭证。
安全凭证包括 SecretId 和 SecretKey：

- SecretId 用于标识 API 调用者身份
- SecretKey 用于加密签名字符串和服务器端验证签名字符串的密钥。
- **用户必须严格保管安全凭证，避免泄露。**

申请安全凭证的具体步骤如下：

1. 登录 腾讯云管理中心控制台 。
2. 前往 云 API 密钥 的控制台页面
3. 在 云 API 密钥 页面，单击【新建密钥】即可以创建一对 SecretId/SecretKey。

注意：每个账号最多可以拥有两对 SecretId/SecretKey。

2. 生成签名串

有了安全凭证 SecretId 和 SecretKey 后，就可以生成签名串了。以下是使用签名方法 v1 生成签名串的详细过程：

假设用户的 SecretId 和 SecretKey 分别是：

- SecretId: AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******
- SecretKey: Gu5t9xGARNpq86cd98joQYCN3*******

注意：这里只是示例，请根据用户实际申请的 SecretId 和 SecretKey 进行后续操作！

以云服务器查看实例列表（DescribeInstances）请求为例，当用户调用这一接口时，其请求参数可能如下：

参数名称	中文	参数值
Action	方法名	DescribeInstances
SecretId	密钥 ID	AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******
Timestamp	当前时间戳	1465185768
Nonce	随机正整数	11886

参数名称	中文	参数值
Region	实例所在区域	ap-guangzhou
InstanceIds.0	待查询的实例 ID	ins-09dx96dg
Offset	偏移量	0
Limit	最大允许输出	20
Version	接口版本号	2017-03-12

这里只展示了部分公共参数和接口输入参数，用户可以根据实际需要添加其他参数，例如 Language 和 Token 公共参数。

2.1. 对参数排序

首先对所有请求参数按参数名的字典序（ ASCII 码）升序排序。注意：1）只按参数名进行排序，参数值保持对应即可，不参与比大小；2）按 ASCII 码比大小，如 InstanceIds.2 要排在 InstanceIds.12 后面，不是按字母表，也不是按数值。用户可以借助编程语言中的相关排序函数来实现这一功能，如 PHP 中的 ksort 函数。上述示例参数的排序结果如下:

```
{
'Action' : 'DescribeInstances',
'InstanceIds.0' : 'ins-09dx96dg',
'Limit' : 20,
'Nonce' : 11886,
'Offset' : 0,
'Region' : 'ap-guangzhou',
'SecretId' : 'AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******',
'Timestamp' : 1465185768,
'Version': '2017-03-12',
}
```

使用其它程序设计语言开发时，可对上面示例中的参数进行排序，得到的结果一致即可。

2.2. 拼接请求字符串

此步骤生成请求字符串。
将把上一步排序好的请求参数格式化成“参数名称=参数值”的形式，如对 Action 参数，其参数名称为 "Action" ，参数值为 "DescribeInstances" ，因此格式化后就为 Action=DescribeInstances 。
注意：“参数值”为原始值而非 url 编码后的值。

然后将格式化后的各个参数用"&"拼接在一起，最终生成的请求字符串为:

```
Action=DescribeInstances&InstanceIds.0=ins-09dx96dg&Limit=20&Nonce=11886&Offset=0&Region=ap-guangzhou&SecretId=AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******&Timestamp=1465185768&Version=2017-03-12
```

2.3. 拼接签名原文字符串

此步骤生成签名原文字符串。
签名原文字符串由以下几个参数构成:

1. 请求方法: 支持 POST 和 GET 方式，这里使用 GET 请求，注意方法为全大写。
2. 请求主机:查看实例列表(DescribeInstances)的请求域名为：cvm.tencentcloudapi.com。实际的请求域名根据接口所属模块的不同而不同，详见各接口说明。
3. 请求路径: 当前版本云API的请求路径固定为 / 。

4. 请求字符串: 即上一步生成的请求字符串。

签名原文串的拼接规则为： 请求方法 + 请求主机 +请求路径 + ? + 请求字符串 。

示例的拼接结果为：

```
GETcvm.tencentcloudapi.com/?Action=DescribeInstances&InstanceIds.0=ins-09dx96dg&Limit=20&Nonce=11886&Offset=0&Region=ap-guangzhou&SecretId=AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******&Timestamp=1465185768&Version=2017-03-12
```

2.4. 生成签名串

此步骤生成签名串。
首先使用 HMAC-SHA1 算法对上一步中获得的**签名原文字符串**进行签名，然后将生成的签名串使用 Base64 进行编码，即可获得最终的签名串。

具体代码如下，以 PHP 语言为例：

```
$secretKey = 'Gu5t9xGARNpq86cd98joQYCN3*******';
$srcStr = 'GETcvm.tencentcloudapi.com/?Action=DescribeInstances&InstanceIds.0=ins-09dx96dg&Limit=20&Nonce=11886&Offset=0&Region=ap-guangzhou&SecretId=AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******&Timestamp=1465185768&Version=2017-03-12';
$signStr = base64_encode(hash_hmac('sha1', $srcStr, $secretKey, true));
echo $signStr;
```

最终得到的签名串为：

```
zmmjn35mikh6pM3V7sUEuX4wyYM=
```

使用其它程序设计语言开发时，可用上面示例中的原文进行签名验证，得到的签名串与例子中的一致即可。

3. 签名串编码

生成的签名串并不能直接作为请求参数，需要对其进行 URL 编码。

如上一步生成的签名串为 zmmjn35mikh6pM3V7sUEuX4wyYM= ，最终得到的签名串请求参数（Signature）为：zmmjn35mikh6pM3V7sUEuX4wyYM%3D，它将用于生成最终的请求 URL。

注意：如果用户的请求方法是 GET，或者请求方法为 POST 同时 Content-Type 为 application/x-www-form-urlencoded，则发送请求时所有请求参数的值均需要做 URL 编码，参数键和=符号不需要编码。非 ASCII 字符在 URL 编码前需要先以 UTF-8 进行编码。

注意：有些编程语言的网络库会自动为所有参数进行 urlencode，在这种情况下，就不需要对签名串进行 URL 编码了，否则两次 URL 编码会导致签名失败。

注意：其他参数值也需要进行编码，编码采用 RFC 3986。使用 %XY 对特殊字符例如汉字进行百分比编码，其中“X”和“Y”为十六进制字符（0-9 和大写字母 A-F），使用小写将引发错误。

4. 签名失败

根据实际情况，存在以下签名失败的错误码，请根据实际情况处理。

错误代码	错误描述
AuthFailure.SignatureExpire	签名过期
AuthFailure.SecretIdNotFound	密钥不存在

错误代码	错误描述
AuthFailure.SignatureFailure	签名错误
AuthFailure.TokenFailure	token 错误
AuthFailure.InvalidSecretId	密钥非法（不是云 API 密钥类型）

5. 签名演示

在实际调用 API 3.0 时，推荐使用配套的腾讯云 SDK 3.0，SDK 封装了签名的过程，开发时只关注产品提供的具体接口即可。详细信息参见 SDK 中心。当前支持的编程语言有：

- Python
- Java
- PHP
- Go
- NodeJS
- .NET
- C++
- Ruby

下面提供了不同产品的生成签名 demo，您可以找到对应的产品参考签名的生成：

- Signature Demo

为了更清楚的解释签名过程，下面以实际编程语言为例，将上述的签名过程具体实现。请求的域名、调用的接口和参数的取值都以上述签名过程为准，代码只为解释签名过程，并不具备通用性，实际开发请尽量使用 SDK 。

最终输出的 url 可能为：https://cvm.tencentcloudapi.com/?Action=DescribeInstances&InstanceIds.0=ins-09dx96dg&Limit=20&Nonce=11886&Offset=0&Region=ap-guangzhou&SecretId=AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******&Signature=zmmjn35mikh6pM3V7sUEuX4wyYM%3D&Timestamp=1465185768&Version=2017-03-12 。

注意：由于示例中的密钥是虚构的，时间戳也不是系统当前时间，因此如果将此 url 在浏览器中打开或者用 curl 等命令调用时会返回鉴权错误：签名过期。为了得到一个可以正常返回的 url，需要修改示例中的 SecretId 和 SecretKey 为真实的密钥，并使用系统当前时间戳作为 Timestamp 。

注意：在下面的示例中，不同编程语言，甚至同一语言每次执行得到的 url 可能都有所不同，表现为参数的顺序不同，但这并不影响正确性。只要所有参数都在，且签名计算正确即可。

注意：以下代码仅适用于 API 3.0，不能直接用于其他的签名流程，请以对应的实际文档为准。

Java

```
import java.io.UnsupportedEncodingException;
import java.net.URLEncoder;
import java.util.Random;
import java.util.TreeMap;
import javax.crypto.Mac;
import javax.crypto.spec.SecretKeySpec;
import javax.xml.bind.DatatypeConverter;

public class TencentCloudAPIDemo {
private final static String CHARSET = "UTF-8";
```

```
public static String sign(String s, String key, String method) throws Exception {
Mac mac = Mac.getInstance(method);
SecretKeySpec secretKeySpec = new SecretKeySpec(key.getBytes(CHARSET), mac.getAlgorithm());
mac.init(secretKeySpec);
byte[] hash = mac.doFinal(s.getBytes(CHARSET));
return DatatypeConverter.printBase64Binary(hash);
}

public static String getStringToSign(TreeMap<String, Object> params) {
StringBuilder s2s = new StringBuilder("GETcvm.tencentcloudapi.com/?");
// 签名时要求对参数进行字典排序，此处用TreeMap保证顺序
for (String k : params.keySet()) {
s2s.append(k).append("=").append(params.get(k).toString()).append("&");
}
return s2s.toString().substring(0, s2s.length() - 1);
}

public static String getUrl(TreeMap<String, Object> params) throws UnsupportedEncodingException {
StringBuilder url = new StringBuilder("https://cvm.tencentcloudapi.com/?");
// 实际请求的url中对参数顺序没有要求
for (String k : params.keySet()) {
// 需要对请求串进行urlencode，由于key都是英文字母，故此处仅对其value进行urlencode
url.append(k).append("=").append(URLEncoder.encode(params.get(k).toString(), CHARSET)).append("&");
}
return url.toString().substring(0, url.length() - 1);
}

public static void main(String[] args) throws Exception {
TreeMap<String, Object> params = new TreeMap<String, Object>(); // TreeMap可以自动排序
// 实际调用时应当使用随机数，例如：params.put("Nonce", new Random().nextInt(java.lang.Integer.MAX_VALUE));
params.put("Nonce", 11886); // 公共参数
// 实际调用时应当使用系统当前时间，例如： params.put("Timestamp", System.currentTimeMillis() / 1000);
params.put("Timestamp", 1465185768); // 公共参数
// 需要设置环境变量 TENCENTCLOUD_SECRET_ID，值为示例的 AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******
params.put("SecretId", System.getenv("TENCENTCLOUD_SECRET_ID")); // 公共参数
params.put("Action", "DescribeInstances"); // 公共参数
params.put("Version", "2017-03-12"); // 公共参数
params.put("Region", "ap-guangzhou"); // 公共参数
params.put("Limit", 20); // 业务参数
params.put("Offset", 0); // 业务参数
params.put("InstanceIds.0", "ins-09dx96dg"); // 业务参数
// 需要设置环境变量 TENCENTCLOUD_SECRET_KEY，值为示例的 Gu5t9xGARNpq86cd98joQYCN3*******
params.put("Signature", sign(getStringToSign(params), System.getenv("TENCENTCLOUD_SECRET_KEY"), "HmacSHA1")); // 公共
参数
System.out.println(getUrl(params));
}
}
```

Python

注意：如果是在 Python 2 环境中运行，需要先安装 requests 依赖包： pip install requests 。

```
# -*- coding: utf8 -*-
import base64
import hashlib
import hmac
import os
import time

import requests

# 需要设置环境变量 TENCENTCLOUD_SECRET_ID，值为示例的 AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******
secret_id = os.environ.get("TENCENTCLOUD_SECRET_ID")
# 需要设置环境变量 TENCENTCLOUD_SECRET_KEY，值为示例的 Gu5t9xGARNpq86cd98joQYCN3*******
secret_key = os.environ.get("TENCENTCLOUD_SECRET_KEY")

def get_string_to_sign(method, endpoint, params):
s = method + endpoint + "/?"
query_str = "&".join("%s=%s" % (k, params[k]) for k in sorted(params))
return s + query_str

def sign_str(key, s, method):
hmac_str = hmac.new(key.encode("utf8"), s.encode("utf8"), method).digest()
return base64.b64encode(hmac_str)

if __name__ == '__main__':
endpoint = "cvm.tencentcloudapi.com"
data = {
'Action' : 'DescribeInstances',
'InstanceIds.0' : 'ins-09dx96dg',
'Limit' : 20,
'Nonce' : 11886,
'Offset' : 0,
'Region' : 'ap-guangzhou',
'SecretId' : secret_id,
'Timestamp' : 1465185768, # int(time.time())
'Version': '2017-03-12'
}
s = get_string_to_sign("GET", endpoint, data)
data["Signature"] = sign_str(secret_key, s, hashlib.sha1)
print(data["Signature"])
# 此处会实际调用，成功后可能产生计费
# resp = requests.get("https://" + endpoint, params=data)
# print(resp.url)
```

Golang

```
package main
```

```
import (
"bytes"
"crypto/hmac"
"crypto/sha1"
"encoding/base64"
"fmt"
"os"
"sort"
"strconv"
)

func main() {
// 需要设置环境变量 TENCENTCLOUD_SECRET_ID，值为示例的 AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******
secretId := os.Getenv("TENCENTCLOUD_SECRET_ID")
// 需要设置环境变量 TENCENTCLOUD_SECRET_KEY，值为示例的 Gu5t9xGARNpq86cd98joQYCN3*******
secretKey := os.Getenv("TENCENTCLOUD_SECRET_KEY")
params := map[string]string{
"Nonce": "11886",
"Timestamp": strconv.Itoa(1465185768),
"Region": "ap-guangzhou",
"SecretId": secretId,
"Version": "2017-03-12",
"Action": "DescribeInstances",
"InstanceIds.0": "ins-09dx96dg",
"Limit": strconv.Itoa(20),
"Offset": strconv.Itoa(0),
}

var buf bytes.Buffer
buf.WriteString("GET")
buf.WriteString("cvm.tencentcloudapi.com")
buf.WriteString("/")
buf.WriteString("?")

// sort keys by ascii asc order
keys := make([]string, 0, len(params))
for k, _ := range params {
keys = append(keys, k)
}
sort.Strings(keys)

for i := range keys {
k := keys[i]
buf.WriteString(k)
buf.WriteString("=")
buf.WriteString(params[k])
buf.WriteString("&")
}
buf.Truncate(buf.Len() - 1)
```

```
hashed := hmac.New(sha1.New, []byte(secretKey))
hashed.Write(buf.Bytes())

fmt.Println(base64.StdEncoding.EncodeToString(hashed.Sum(nil)))
}
```

PHP

```
<?php
// 需要设置环境变量 TENCENTCLOUD_SECRET_ID，值为示例的 AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******
$secretId = getenv("TENCENTCLOUD_SECRET_ID");
// 需要设置环境变量 TENCENTCLOUD_SECRET_KEY，值为示例的 Gu5t9xGARNpq86cd98joQYCN3*******
$secretKey = getenv("TENCENTCLOUD_SECRET_KEY");
$param["Nonce"] = 11886;//rand();
$param["Timestamp"] = 1465185768;//time();
$param["Region"] = "ap-guangzhou";
$param["SecretId"] = $secretId;
$param["Version"] = "2017-03-12";
$param["Action"] = "DescribeInstances";
$param["InstanceIds.0"] = "ins-09dx96dg";
$param["Limit"] = 20;
$param["Offset"] = 0;

ksort($param);

$signStr = "GETcvm.tencentcloudapi.com/?";
foreach ( $param as $key => $value ) {
$signStr = $signStr . $key . "=" . $value . "&";
}
$signStr = substr($signStr, 0, -1);

$signature = base64_encode(hash_hmac("sha1", $signStr, $secretKey, true));
echo $signature.PHP_EOL;
// need to install and enable curl extension in php.ini
// $param["Signature"] = $signature;
// $url = "https://cvm.tencentcloudapi.com/?".http_build_query($param);
// echo $url.PHP_EOL;
// $ch = curl_init();
// curl_setopt($ch, CURLOPT_URL, $url);
// $output = curl_exec($ch);
// curl_close($ch);
// echo json_decode($output);
```

Ruby

```
# -*- coding: UTF-8 -*-
# require ruby>=2.3.0
require 'time'
require 'openssl'
```

```
require 'base64'

# 需要设置环境变量 TENCENTCLOUD_SECRET_ID，值为示例的 AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******
secret_id = ENV["TENCENTCLOUD_SECRET_ID"]
# 需要设置环境变量 TENCENTCLOUD_SECRET_KEY，值为示例的 Gu5t9xGARNpq86cd98joQYCN3*******
secret_key = ENV["TENCENTCLOUD_SECRET_KEY"]

method = 'GET'
endpoint = 'cvm.tencentcloudapi.com'
data = {
'Action' => 'DescribeInstances',
'InstanceIds.0' => 'ins-09dx96dg',
'Limit' => 20,
'Nonce' => 11886,
'Offset' => 0,
'Region' => 'ap-guangzhou',
'SecretId' => secret_id,
'Timestamp' => 1465185768, # Time.now.to_i
'Version' => '2017-03-12',
}
sign = method + endpoint + '/?'
params = []
data.sort.each do |item|
params << "#{item[0]}=#{item[1]}"
end
sign += params.join('&')
digest = OpenSSL::Digest.new('sha1')
data['Signature'] = Base64.encode64(OpenSSL::HMAC.digest(digest, secret_key, sign))
puts data['Signature']

# require 'net/http'
# uri = URI('https://' + endpoint)
# uri.query = URI.encode_www_form(data)
# p uri
# res = Net::HTTP.get_response(uri)
# puts res.body
```

DotNet

```
using System;
using System.Collections.Generic;
using System.Net;
using System.Security.Cryptography;
using System.Text;

public class Application {
public static string Sign(string signKey, string secret)
{
string signRet = string.Empty;
```

```
using (HMACSHA1 mac = new HMACSHA1(Encoding.UTF8.GetBytes(signKey)))
{
byte[] hash = mac.ComputeHash(Encoding.UTF8.GetBytes(secret));
signRet = Convert.ToBase64String(hash);
}
return signRet;
}
public static string MakeSignPlainText(SortedDictionary<string, string> requestParams, string requestMethod, string requestHos
t, string requestPath)
{
string retStr = "";
retStr += requestMethod;
retStr += requestHost;
retStr += requestPath;
retStr += "?";
string v = "";
foreach (string key in requestParams.Keys)
{
v += string.Format("{0}={1}&", key, requestParams[key]);
}
retStr += v.TrimEnd('&');
return retStr;
}

public static void Main(string[] args)
{
// 密钥参数
// 需要设置环境变量 TENCENTCLOUD_SECRET_ID，值为示例的 AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******
string SECRET_ID = Environment.GetEnvironmentVariable("TENCENTCLOUD_SECRET_ID");
// 需要设置环境变量 TENCENTCLOUD_SECRET_KEY，值为示例的 Gu5t9xGARNpq86cd98joQYCN3*******
string SECRET_KEY = Environment.GetEnvironmentVariable("TENCENTCLOUD_SECRET_KEY");

string endpoint = "cvm.tencentcloudapi.com";
string region = "ap-guangzhou";
string action = "DescribeInstances";
string version = "2017-03-12";
double RequestTimestamp = 1465185768; // 时间戳 2019-02-26 00:44:25,此参数作为示例，以实际为准
// long timestamp = ToTimestamp() / 1000;
// string requestTimestamp = timestamp.ToString();
Dictionary<string, string> param = new Dictionary<string, string>();
param.Add("Limit", "20");
param.Add("Offset", "0");
param.Add("InstanceIds.0", "ins-09dx96dg");
param.Add("Action", action);
param.Add("Nonce", "11886");
// param.Add("Nonce", Math.Abs(new Random().Next()).ToString());

param.Add("Timestamp", RequestTimestamp.ToString());
param.Add("Version", version);
```

```
param.Add("SecretId", SECRET_ID);
param.Add("Region", region);
SortedDictionary<string, string> headers = new SortedDictionary<string, string>(param, StringComparer.Ordinal);
string sigInParam = MakeSignPlainText(headers, "GET", endpoint, "/");
string sigOutParam = Sign(SECRET_KEY, sigInParam);
Console.WriteLine(sigOutParam);
}
}
```

NodeJS

```
const crypto = require('crypto');

function get_req_url(params, endpoint){
params['Signature'] = encodeURIComponent(params['Signature']);
const url_strParam = sort_params(params)
return "https://" + endpoint + "/?" + url_strParam.slice(1);
}

function formatSignString(reqMethod, endpoint, path, strParam){
let strSign = reqMethod + endpoint + path + "?" + strParam.slice(1);
return strSign;
}
function sha1(secretKey, strsign){
let signMethodMap = {'HmacSHA1': "sha1"};
let hmac = crypto.createHmac(signMethodMap['HmacSHA1'], secretKey || "");
return hmac.update(Buffer.from(strsign, 'utf8')).digest('base64')
}

function sort_params(params){
let strParam = "";
let keys = Object.keys(params);
keys.sort();
for (let k in keys) {
//k = k.replace(/_/g, '.');
strParam += ("&" + keys[k] + "=" + params[keys[k]]);
}
return strParam
}

function main(){
// 密钥参数
// 需要设置环境变量 TENCENTCLOUD_SECRET_ID，值为示例的 AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******
const SECRET_ID = process.env.TENCENTCLOUD_SECRET_ID
// 需要设置环境变量 TENCENTCLOUD_SECRET_KEY，值为示例的 Gu5t9xGARNpq86cd98joQYCN3*******
const SECRET_KEY = process.env.TENCENTCLOUD_SECRET_KEY

const endpoint = "cvm.tencentcloudapi.com"
const Region = "ap-guangzhou"
```

```
const Version = "2017-03-12"
const Action = "DescribeInstances"
const Timestamp = 1465185768 // 时间戳 2016-06-06 12:02:48, 此参数作为示例，以实际为准
// const Timestamp = Math.round(Date.now() / 1000)
const Nonce = 11886 // 随机正整数
//const nonce = Math.round(Math.random() * 65535)

let params = {};
params['Action'] = Action;
params['InstanceIds.0'] = 'ins-09dx96dg';
params['Limit'] = 20;
params['Offset'] = 0;
params['Nonce'] = Nonce;
params['Region'] = Region;
params['SecretId'] = SECRET_ID;
params['Timestamp'] = Timestamp;
params['Version'] = Version;

// 1. 对参数排序,并拼接请求字符串
strParam = sort_params(params)

// 2. 拼接签名原文字符串
const reqMethod = "GET";
const path = "/";
strSign = formatSignString(reqMethod, endpoint, path, strParam)
// console.log(strSign)

// 3. 生成签名串
params['Signature'] = sha1(SECRET_KEY, strSign)
console.log(params['Signature'])

// 4. 进行url编码并拼接请求url
// const req_url = get_req_url(params, endpoint)
// console.log(params['Signature'])
// console.log(req_url)
}
main()
```

返回结果

最近更新时间：2024-03-12 01:33:55

云 API 3.0 接口默认返回 JSON 数据，返回非 JSON 格式的接口会在文档中做出说明。返回 JSON 数据时最大限制为 50 MB，如果返回的数据超过最大限制，请求会失败并返回内部错误。请根据接口文档中给出的过滤功能（例如时间范围）或者分页功能，控制返回数据不要过大。

注意：目前只要请求被服务端正常处理了，响应的 HTTP 状态码均为200。例如返回的消息体里的错误码是签名失败，但 HTTP 状态码是200，而不是401。

正确返回结果

以云服务器的接口查看实例状态列表 (DescribeInstancesStatus) 2017-03-12 版本为例，若调用成功，其可能的返回如下为：

```
{
"Response": {
"TotalCount": 0,
"InstanceStatusSet": [],
"RequestId": "b5b41468-520d-4192-b42f-595cc34b6c1c"
}
}
```

- Response 及其内部的 RequestId 是固定的字段，无论请求成功与否，只要 API 处理了，则必定会返回。
- RequestId 用于一个 API 请求的唯一标识，如果 API 出现异常，可以联系 腾讯云客服 或 提交工单，并提供该 ID 来解决问题。
- 除了固定的字段外，其余均为具体接口定义的字段，不同的接口所返回的字段参见接口文档中的定义。此例中的 TotalCount 和 InstanceStatusSet 均为 DescribeInstancesStatus 接口定义的字段，由于调用请求的用户暂时还没有云服务器实例，因此 TotalCount 在此情况下的返回值为 0， InstanceStatusSet 列表为空。

错误返回结果

若调用失败，其返回值示例如下为：

```
{
"Response": {
"Error": {
"Code": "AuthFailure.SignatureFailure",
"Message": "The provided credentials could not be validated. Please check your signature is correct."
},
"RequestId": "ed93f3cb-f35e-473f-b9f3-0d451b8b79c6"
}
}
```

- Error 的出现代表着该请求调用失败。Error 字段连同其内部的 Code 和 Message 字段在调用失败时是必定返回的。
- Code 表示具体出错的错误码，当请求出错时可以先根据该错误码在公共错误码和当前接口对应的错误码列表里面查找对应原因和解决方案。
- Message 显示出了这个错误发生的具体原因，随着业务发展或体验优化，此文本可能会经常保持变更或更新，用户不应依赖这个返回值。
- RequestId 用于一个 API 请求的唯一标识，如果 API 出现异常，可以联系 腾讯云客服 或 提交工单，并提供该 ID 来解决问题。

公共错误码

返回结果中如果存在 Error 字段，则表示调用 API 接口失败。 Error 中的 Code 字段表示错误码，所有业务都可能出现的错误码为公共错误码。完整的错误码列表请参考本产品“API 文档”目录下的“错误码”页面。

参数类型

最近更新时间：2022-08-10 06:35:23

目前腾讯云 API 3.0 输入参数和输出参数支持如下几种数据格式：

- String: 字符串。
- Integer：整型，上限为无符号64位整数。SDK 3.0 不同编程语言支持的类型有所差异，建议以所使用编程语言的最大整型定义，例如 Golang 的 uint64。
- Boolean：布尔型。
- Float：浮点型。
- Double：双精度浮点型。
- Date：字符串，日期格式。例如：2022-01-01。
- Timestamp：字符串，时间格式。例如：2022-01-01 00:00:00。
- Timestamp ISO8601：ISO 8601 是由国际标准化组织（International Organization for Standardization，ISO）发布的关于日期和时间格式的国际标准，对应国标《GB/T 7408-2005数据元和交换格式信息交换日期和时间表示法》。建议以所使用编程语言的标准库进行格式解析。例如：2022-01-01T00:00:00+08:00。
- Binary：二进制内容，需要以特定协议请求和解析。

固件升级相关接口
更新固件升级任务状态

最近更新时间：2024-03-12 01:33:49

1. 接口描述

接口请求域名： iotcloud.tencentcloudapi.com 。

本接口（UpdateOtaTask）当固件升级大任务处于没有在全部成功的状态时，可修改为取消状态，取消部分或全部设备的升级;或其它允许的可修改的状态

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer

点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：UpdateOtaTaskStatus。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	是	String	公共参数，详见产品支持的 地域列表。
ProductId	是	String	产品ID 示例值：ABCDE12345
TaskId	是	Integer	固件升级任务ID 示例值：11236
Status	是	Integer	固件任务取消状态 示例值：6

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 修改固件升级任务状态为取消状态

当固件任务在升级过程需要取消时可使用。或修改为其它允许的状态

输入示例

```
https://iotcloud.tencentcloudapi.com/?Action=CancelDeviceFirmwareTask
&ProductId=ABCDE12345
&TaskId=10000
&Status=6
&<公共请求参数>
```

输出示例

```
{
"Response": {
"RequestId": "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx"
}
}
```

5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python: GitHub Gitee
- Tencent Cloud SDK 3.0 for Java: GitHub Gitee
- Tencent Cloud SDK 3.0 for PHP: GitHub Gitee
- Tencent Cloud SDK 3.0 for Go: GitHub Gitee
- Tencent Cloud SDK 3.0 for Node.js: GitHub Gitee
- Tencent Cloud SDK 3.0 for .NET: GitHub Gitee
- Tencent Cloud SDK 3.0 for C++: GitHub Gitee
- Tencent Cloud SDK 3.0 for Ruby: GitHub Gitee

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

该接口暂无业务逻辑相关的错误码，其他错误码详见 公共错误码。

设备影子相关接口
删除设备影子

最近更新时间：2024-03-12 01:33:48

1. 接口描述

接口请求域名： iotcloud.tencentcloudapi.com 。

本接口（DeleteDeviceShadow）用于删除设备影子

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer

点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：DeleteDeviceShadow。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	是	String	公共参数，详见产品支持的 地域列表。
ProductId	是	String	产品ID 示例值：ABCDE12345
DeviceName	是	String	设备名称 示例值：abc

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 删除设备影子接口

输入示例

```
https://iotcloud.tencentcloudapi.com/?Action=DeleteDeviceShadow
&ProductId=ABCDE12345
&DeviceName=abc
&<公共请求参数>
```

输出示例

```
{
"Response": {
"RequestId": "xxxxxxxxxxxxxxxxxxxxxxxx"
}
}
```

5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python: GitHub Gitee
- Tencent Cloud SDK 3.0 for Java: GitHub Gitee
- Tencent Cloud SDK 3.0 for PHP: GitHub Gitee
- Tencent Cloud SDK 3.0 for Go: GitHub Gitee
- Tencent Cloud SDK 3.0 for Node.js: GitHub Gitee
- Tencent Cloud SDK 3.0 for .NET: GitHub Gitee
- Tencent Cloud SDK 3.0 for C++: GitHub Gitee
- Tencent Cloud SDK 3.0 for Ruby: GitHub Gitee

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
InternalError	内部错误。
InvalidParameterValue	参数取值错误。
ResourceNotFound.DeviceShadowNotExist	设备影子不存在。

获取设备影子

最近更新时间：2024-03-12 01:33:48

1. 接口描述

接口请求域名： iotcloud.tencentcloudapi.com 。

本接口（DescribeDeviceShadow）用于查询虚拟设备信息。

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer　点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：DescribeDeviceShadow。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	是	String	公共参数，详见产品支持的 地域列表。
ProductId	是	String	产品 ID 示例值：ABCDE12345
DeviceName	是	String	设备名称。命名规则：[a-zA-Z0-9:_-]{1,60} 示例值：abc

3. 输出参数

参数名称	类型	描述
Data	String	设备影子数据 示例值：{"payload":{"state": {"reported": { "color": "red"}}, "timestamp
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 获取设备影子

输入示例

```
https://iotcloud.tencentcloudapi.com/?Action=DescribeDeviceShadow
&ProductId=ABCDE12345
&DeviceName=abc
&<公共请求参数>
```

输出示例


```
{
"Response": {
"Data": "{\"payload\":{\"state\": {\"reported\": { \"color\": \"red\"}}, \"metadata\": {\"reported\": {\"color\": {\"timestamp\": 150
9092895971}}}, \"timestamp\": 1509443636326, \"version\": 5},\"result\":0,\"timestamp\":1509440846582,\"type\":\"get\"}",
"RequestId": "xxxxxxxxxxxxxxxxxxxxxx"
}
}
```


5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python: GitHub Gitee
- Tencent Cloud SDK 3.0 for Java: GitHub Gitee
- Tencent Cloud SDK 3.0 for PHP: GitHub Gitee
- Tencent Cloud SDK 3.0 for Go: GitHub Gitee
- Tencent Cloud SDK 3.0 for Node.js: GitHub Gitee
- Tencent Cloud SDK 3.0 for .NET: GitHub Gitee
- Tencent Cloud SDK 3.0 for C++: GitHub Gitee
- Tencent Cloud SDK 3.0 for Ruby: GitHub Gitee

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
InternalError	内部错误。
InvalidParameterValue	参数取值错误。
InvalidParameterValue.ParamIncomplete	请求中缺少关键字段信息。
ResourceNotFound.DeviceShadowNotExist	设备影子不存在。
ResourceNotFound.ProductOrDeviceNotExist	用户不存在此产品或设备。

更新设备影子

最近更新时间：2024-03-12 01:33:47

1. 接口描述

接口请求域名： iotcloud.tencentcloudapi.com 。

本接口（UpdateDeviceShadow）用于更新虚拟设备信息。

默认接口请求频率限制：100次/秒。

推荐使用 API Explorer　点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：UpdateDeviceShadow。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	是	String	公共参数，详见产品支持的 地域列表。
ProductId	是	String	产品ID 示例值：ABCDE12345
DeviceName	是	String	设备名称 示例值：ABC
State	是	String	虚拟设备的状态，JSON字符串格式，由desired结构组成 示例值：xxx
ShadowVersion	是	Integer	当前版本号，需要和后台的version保持一致，才能更新成功 示例值：0

3. 输出参数

参数名称	类型	描述
Data	String	设备影子数据，JSON字符串格式 示例值：{"payload":{"state": {"desired": { "color": "red"}}, "timestamp"
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 更新设备影子

输入示例

```
https://iotcloud.tencentcloudapi.com/?Action=UpdateDeviceShadow
&ProductId=ABCDE12345
&DeviceName=abc
&State={"desired":{"color":"red"}}
&ShadowVersion=1
&<公共请求参数>
```

输出示例


```
{
"Response": {
"Data": "{\"payload\":{\"state\": {\"desired\": { \"color\": \"red\"}}, \"metadata\": {\"desired\": {\"color\": {\"timestamp\": 1509092895971}}}, \"timestamp\": 1509443636326, \"version\": 5},\"result\":0,\"timestamp\":1509440846582,\"type\":\"update\"}",
"RequestId": "9e574269-093f-4a7f-bf90-24ef80b6528a"
}
}
```


5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python: GitHub Gitee
- Tencent Cloud SDK 3.0 for Java: GitHub Gitee
- Tencent Cloud SDK 3.0 for PHP: GitHub Gitee
- Tencent Cloud SDK 3.0 for Go: GitHub Gitee
- Tencent Cloud SDK 3.0 for Node.js: GitHub Gitee
- Tencent Cloud SDK 3.0 for .NET: GitHub Gitee
- Tencent Cloud SDK 3.0 for C++: GitHub Gitee
- Tencent Cloud SDK 3.0 for Ruby: GitHub Gitee

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
FailedOperation.UpdateVersionNotMatch	更新版本不匹配。

错误码	描述
InternalError	内部错误。
InvalidParameterValue	参数取值错误。
InvalidParameterValue.InvalidJSON	JSON参数非法。
InvalidParameterValue.JSONHasInvalidNode	State JSON对象中包含非法节点。
InvalidParameterValue.JSONSizeExceedLimit	State JSON对象超过大小限制，最大为 8k。
InvalidParameterValue.NotMergeAble	不可合并。
InvalidParameterValue.PrefixInvalid	prefix不合法。
ResourceNotFound.DeviceShadowNotExist	设备影子不存在。
ResourceNotFound.ProductOrDeviceNotExist	用户不存在此产品或设备。

设备相关接口
批量绑定子设备

最近更新时间：2024-03-12 01:33:47

1. 接口描述

接口请求域名： iotcloud.tencentcloudapi.com 。

本接口（BindDevices）用于网关设备批量绑定子设备

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer　点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：BindDevices。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	是	String	公共参数，详见产品支持的 地域列表。
GatewayProductId	是	String	网关设备的产品ID 示例值：12345ABCDE
GatewayDeviceName	是	String	网关设备的设备名 示例值：test
ProductId	是	String	被绑定设备的产品ID 示例值：ABCDE12345
DeviceNames.N	是	Array of String	被绑定的多个设备名 示例值：["test0\n","test1\n"]
Skey	否	String	中兴CLAA设备的绑定需要skey，普通的设备不需要 示例值：123

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 批量绑定子设备

输入示例

```
https://iotcloud.tencentcloudapi.com/?Action=BindDevices
&GatewayProductId=12345ABCDE
&GatewayDeviceName=test
&ProductId=ABCDE12345
&DeviceNames.0=test0
&DeviceNames.1=test1
&<公共请求参数>
```

输出示例

```
{
"Response": {
"RequestId": "69f65618-600b-4ac4-b8e3-4528a6819078"
}
}
```

5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python: GitHub Gitee
- Tencent Cloud SDK 3.0 for Java: GitHub Gitee
- Tencent Cloud SDK 3.0 for PHP: GitHub Gitee
- Tencent Cloud SDK 3.0 for Go: GitHub Gitee
- Tencent Cloud SDK 3.0 for Node.js: GitHub Gitee
- Tencent Cloud SDK 3.0 for .NET: GitHub Gitee
- Tencent Cloud SDK 3.0 for C++: GitHub Gitee
- Tencent Cloud SDK 3.0 for Ruby: GitHub Gitee

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
FailedOperation.BindDeviceOverLimit	绑定设备超过限制。

错误码	描述
FailedOperation.BindDevicePerOnceOverLimit	单次绑定的设备数量超过限制。
InternalError	内部错误。
InternalError.DBOperationError	数据库内部错误。
InvalidParameterValue	参数取值错误。
InvalidParameterValue.DeviceIsNotGateway	设备不是网关类型。
ResourceNotFound.DeviceNotExist	设备不存在。
ResourceNotFound.ProductNotExist	产品不存在。
UnauthorizedOperation.DeviceHasAlreadyBindGateway	该设备绑定了网关设备，无法删除。
UnauthorizedOperation.DeviceIsNotEnabled	设备未启用。

创建设备

最近更新时间：2024-03-12 01:33:47

1. 接口描述

接口请求域名： iotcloud.tencentcloudapi.com 。

本接口（CreateDevice）用于新建一个物联网通信设备。

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer　点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：CreateDevice。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	是	String	公共参数，详见产品支持的 地域列表。
ProductId	是	String	产品 ID 。创建产品时腾讯云为用户分配全局唯一的 ID 示例值：ABCDE12345
DeviceName	是	String	设备名称。命名规则：[a-zA-Z0-9:-]{1,48}。 *示例值：test*device
Attribute	否	Attribute	设备属性
DefinedPsk	否	String	是否使用自定义PSK，默认不使用 示例值：ABC
Isp	否	Integer	运营商类型，当产品是NB-IoT产品时，此字段必填。1表示中国电信，2表示中国移动，3表示中国联通 示例值：Isp
Imei	否	String	IMEI，当产品是NB-IoT产品时，此字段必填 示例值：Imsi
LoraDevEui	否	String	LoRa设备的DevEui，当创建LoRa时，此字段必填 示例值：Eui
LoraMoteType	否	Integer	LoRa设备的MoteType 示例值：Type
Skey	否	String	创建LoRa设备需要skey 示例值：skey
LoraAppKey	否	String	LoRa设备的AppKey 示例值：key

参数名称	必选	类型	描述
TlsCrt	否	String	私有CA创建的设备证书 示例值：-----BEGIN CERTIFICATE----- MIIFGjCCBAKgAwIBAgIQCgRw0Ja8ihLIkKbf

3. 输出参数

参数名称	类型	描述
DeviceName	String	设备名称 示例值：test_device
DevicePsk	String	对称加密密钥，base64编码。采用对称加密时返回该参数 示例值：xxxxxxxxxxxxx
DeviceCert	String	设备证书，用于 TLS 建立链接时校验客户端身份。采用非对称加密时返回该参数 示例值：-----BEGIN CERTIFICATE----- MIIFGjCCBAKgAwIBAgIQCgRw0Ja8ihLIkKbf
DevicePrivateKey	String	设备私钥，用于 TLS 建立链接时校验客户端身份，腾讯云后台不保存，请妥善保管。采用非对称加密时返回该参数 示例值：-----BEGIN PRIVATE KEY----- MIIEvgIBADANBgkqhkiG9w0BAQEFAASCBKgw
LoraDevEui	String	LoRa设备的DevEui，当设备是LoRa设备时，会返回该字段 示例值：xxx
LoraMoteType	Integer	LoRa设备的MoteType，当设备是LoRa设备时，会返回该字段 示例值：xxx
LoraAppKey	String	LoRa设备的AppKey，当设备是LoRa设备时，会返回该字段 示例值：xxx
LoraNwkKey	String	LoRa设备的NwkKey，当设备是LoRa设备时，会返回该字段 示例值：xxx
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 创建设备（采用对称加密）

输入示例

```
https://iotcloud.tencentcloudapi.com/?Action=CreateDevice
&ProductId=ABCDE12345
&DeviceName=test_device
&Attribute.Tags.0.Tag=note
&Attribute.Tags.0.Type=2
&Attribute.Tags.0.Value=test_note
&<公共请求参数>
```

输出示例

```
{
"Response": {
"DeviceName": "test_device",
"DevicePsk": "xxxxxxxxxxxx",
"DeviceCert": "",
"DevicePrivateKey": "",
"LoraDevEui": "",
"LoraMoteType": 1,
"LoraNwkKey": "",
"LoraAppKey": "xx",
"RequestId": "54f75f05-a87c-45fc-9520-6b59e251e91c"
}
}
```


示例2 创建设备（采用非对称加密）

输入示例

```
https://iotcloud.tencentcloudapi.com/?Action=CreateDevice
&ProductId=ABCDE12345
&DeviceName=test_device
&<公共请求参数>
```

输出示例


```
{
"Response": {
"DeviceName": "test_device",
"DevicePsk": "",
"DeviceCert": "xxxxxxxxxxxxxxxxxxx",
"DevicePrivateKey": "xxxxxxxxxxxxxxxxxxxxxx",
"LoraDevEui": "",
"LoraMoteType": 1,
"LoraNwkKey": "",
"LoraAppKey": "xx",
"RequestId": "54f75f05-a87c-45fc-9520-6b59e251e91c"
}
}
```


5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python: GitHub Gitee
- Tencent Cloud SDK 3.0 for Java: GitHub Gitee
- Tencent Cloud SDK 3.0 for PHP: GitHub Gitee
- Tencent Cloud SDK 3.0 for Go: GitHub Gitee
- Tencent Cloud SDK 3.0 for Node.js: GitHub Gitee
- Tencent Cloud SDK 3.0 for .NET: GitHub Gitee
- Tencent Cloud SDK 3.0 for C++: GitHub Gitee
- Tencent Cloud SDK 3.0 for Ruby: GitHub Gitee

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
FailedOperation.AlreadyDistributionDevice	已分发设备，不能再次创建。
FailedOperation.TidWhiteListNotOpen	白名单校验未开启，用户不可创建设备，平台会根据设备认证时携带的设备名称自动创建设备。
InternalError	内部错误。
InternalError.DBOperationError	数据库内部错误。
InvalidParameterValue	参数取值错误。
InvalidParameterValue.DefinedPskNotBase64	格式错误，DefinedPsk需为Base64格式的字符串。
InvalidParameterValue.DeviceAlreadyExist	创建的设备名已存在。
InvalidParameterValue.ProductTypeNotSupport	产品类型不支持。
LimitExceeded.DeviceExceedLimit	设备数量超过限制。
ResourceNotFound.ProductNotExist	产品不存在。
UnauthorizedOperation.ProductCantHaveLoRaDevice	该产品类型不能创建LoRa设备。
UnauthorizedOperation.ProductCantHaveNormalDevice	NB-IoT产品不允许创建普通设备。
UnauthorizedOperation.ProductCantHaveNotLoRaDevice	该产品类型只能创建LoRa设备。
UnauthorizedOperation.ProductIsForbidden	产品禁用了该功能。
UnauthorizedOperation.ProductNotSupportPSK	产品不支持密钥认证。
UnsupportedOperation.SuiteTokenNoCreate	产品为Suite token类型，无法创建新设备。

删除设备

最近更新时间：2024-03-12 01:33:47

1. 接口描述

接口请求域名： iotcloud.tencentcloudapi.com 。

本接口（DeleteDevice）用于删除物联网通信设备。

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer　点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：DeleteDevice。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	是	String	公共参数，详见产品支持的 地域列表。
ProductId	是	String	设备所属的产品 ID 示例值：ABCDE12345
DeviceName	是	String	需要删除的设备名称 示例值：abc
Skey	否	String	删除LoRa设备以及LoRa网关设备需要skey 示例值：xxx

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 删除设备

输入示例

```
https://iotcloud.tencentcloudapi.com/?Action=DeleteDevice
&ProductId=ABCDE12345
&DeviceName=abc
&<公共请求参数>
```

输出示例

```
{
"Response": {
"RequestId": "xxxxxxxxxxxxxxxxxxxxxxxx"
}
}
```

5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python: GitHub Gitee
- Tencent Cloud SDK 3.0 for Java: GitHub Gitee
- Tencent Cloud SDK 3.0 for PHP: GitHub Gitee
- Tencent Cloud SDK 3.0 for Go: GitHub Gitee
- Tencent Cloud SDK 3.0 for Node.js: GitHub Gitee
- Tencent Cloud SDK 3.0 for .NET: GitHub Gitee
- Tencent Cloud SDK 3.0 for C++: GitHub Gitee
- Tencent Cloud SDK 3.0 for Ruby: GitHub Gitee

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
InternalError	内部错误。
InternalError.DBOperationError	数据库内部错误。
InvalidParameterValue	参数取值错误。
ResourceNotFound.DeviceNotExist	设备不存在。
ResourceNotFound.ProductNotExist	产品不存在。
UnauthorizedOperation.DeviceHasAlreadyBindGateway	该设备绑定了网关设备，无法删除。
UnauthorizedOperation.GatewayHasBindedDevices	该设备下仍有绑定的设备。

错误码	描述
UnsupportedOperation.DeviceOtaTaskInProgress	设备ota升级中。

查看设备详情

最近更新时间：2024-03-12 01:33:47

1. 接口描述

接口请求域名： iotcloud.tencentcloudapi.com 。

本接口（DescribeDevice）用于查看设备信息

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer　点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：DescribeDevice。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	是	String	公共参数，详见产品支持的 地域列表。
ProductId	是	String	产品ID 示例值：ABCDE12345
DeviceName	是	String	设备名 示例值：abc

3. 输出参数

参数名称	类型	描述
DeviceName	String	设备名 示例值：ABCD
Online	Integer	设备是否在线，0不在线，1在线，3未激活 示例值：1
LoginTime	Integer	设备登录时间 示例值：1
Version	String	设备固件版本 示例值：1.0.0
LastUpdateTime	Integer	设备最后更新时间 示例值：1
DeviceCert	String	设备证书 示例值：-----BEGIN CERTIFICATE-----IIDS...Fw==-----END CERTIFICATE-----

参数名称	类型	描述
DevicePsk	String	设备密钥 示例值：PSK
Tags	Array of DeviceTag	设备属性
DeviceType	Integer	设备类型 示例值：3
Imei	String	国际移动设备识别码 IMEI 示例值：imei
Isp	Integer	运营商类型 示例值：1
ConnIP	Integer	IP地址 示例值：1
LoraDevEui	String	Lora设备的dev eui 示例值：eui
LoraMoteType	Integer	Lora设备的mote type 示例值：1
LogLevel	Integer	设备的sdk日志等级 注意：此字段可能返回 null，表示取不到有效值。 示例值：1
FirstOnlineTime	Integer	首次上线时间 注意：此字段可能返回 null，表示取不到有效值。 示例值：1617801604
LastOfflineTime	Integer	最近下线时间 注意：此字段可能返回 null，表示取不到有效值。 示例值：1617801604
CreateTime	Integer	设备创建时间 注意：此字段可能返回 null，表示取不到有效值。 示例值：1617801604
CertState	Integer	设备证书获取状态，0 未获取过设备密钥, 1 已获取过设备密钥 注意：此字段可能返回 null，表示取不到有效值。 示例值：-----BEGIN CERTIFICATE----- MIIFGjCCBAKgAwIBAgIQCgRw0Ja8ihLIkKbf
EnableState	Integer	设备启用状态 注意：此字段可能返回 null，表示取不到有效值。 示例值：1
Labels	Array of DeviceLabel	设备标签 注意：此字段可能返回 null，表示取不到有效值。
ClientIP	String	MQTT客户端IP地址 注意：此字段可能返回 null，表示取不到有效值。 示例值：127.0.0.1
FirmwareUpdateTime	Integer	设备固件更新时间 注意：此字段可能返回 null，表示取不到有效值。 示例值：1617801604

参数名称	类型	描述
CreateUserId	Integer	创建者账号ID 注意：此字段可能返回 null，表示取不到有效值。 示例值：0
NBIoTDeviceID	String	NB IoT运营商处的DeviceID 示例值：1233445
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 查看设备详情示例

输入示例

```
https://iotcloud.tencentcloudapi.com/?Action=DescribeDevice
&ProductId=ABCDE12345
&DeviceName=abc
&<公共请求参数>
```

输出示例


```
{
"Response": {
"EnableState": 1,
"LastOfflineTime": 1,
"Version": "2021-04-08",
"CertState": 1,
"Online": 1,
"FirmwareUpdateTime": 1,
"DeviceName": "Device001",
"Tags": [
{
"Tag": "Key",
"Type": 1,
"Name": "Key",
"Value": "Key"
},
{
"Tag": "tag-a",
"Type": 1,
"Name": "tag-name",
"Value": "tag-value"
}
],
"LogLevel": 1,
"FirstOnlineTime": 1,
"DeviceCert": "******",
```

```
"Imei": "Imei",
"ClientIP": "127.0.0.1",
"DevicePsk": "DevicePsk",
"Isp": 1,
"NBIoTDeviceID": "123124",
"LoraDevEui": "your eui",
"DeviceType": 1,
"RequestId": "2abji99uojssd88",
"LoginTime": 1,
"ConnIP": 1,
"LastUpdateTime": 1,
"Labels": [
{
"Value": "value-a",
"Key": "key-a"
}
],
"CreateTime": 1,
"LoraMoteType": 1,
"CreateUserId": 0
}
}
```


5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python: GitHub Gitee
- Tencent Cloud SDK 3.0 for Java: GitHub Gitee
- Tencent Cloud SDK 3.0 for PHP: GitHub Gitee
- Tencent Cloud SDK 3.0 for Go: GitHub Gitee
- Tencent Cloud SDK 3.0 for Node.js: GitHub Gitee
- Tencent Cloud SDK 3.0 for .NET: GitHub Gitee
- Tencent Cloud SDK 3.0 for C++: GitHub Gitee
- Tencent Cloud SDK 3.0 for Ruby: GitHub Gitee

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
InternalError	内部错误。
InternalError.DBOperationError	数据库内部错误。
InvalidParameterValue	参数取值错误。
ResourceNotFound.DeviceNotExist	设备不存在。
ResourceNotFound.ProductNotExist	产品不存在。

获取设备私钥

最近更新时间：2024-03-12 01:33:47

1. 接口描述

接口请求域名： iotcloud.tencentcloudapi.com 。

获取证书认证类型设备的私钥，刚生成或者重置设备后仅可调用一次

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer 点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：DescribeDeviceClientKey。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	是	String	公共参数，详见产品支持的 地域列表。
ProductId	是	String	所属产品的Id 示例值：ABCDE12345
DeviceName	是	String	设备名称 示例值：abc

3. 输出参数

参数名称	类型	描述
ClientKey	String	设备的私钥 示例值：----------BEGIN PRIVATE KEY----------END PRIVATE KEY-----
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 获取设备密钥

输入示例

```
https://iotcloud.tencentcloudapi.com/?Action=DescribeDeviceClientKey
&ProductId=ABCDE12345
&DeviceName=abc
&<公共请求参数>
```

输出示例

```
{
"Response": {
"ClientKey": "----------BEGIN PRIVATE KEY-----\nIIDS...Fw==\n-----END PRIVATE KEY-----\n",
"RequestId": "54f75f05-a87c-45fc-9520-6b59e251e91c"
}
}
```

5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python: GitHub Gitee
- Tencent Cloud SDK 3.0 for Java: GitHub Gitee
- Tencent Cloud SDK 3.0 for PHP: GitHub Gitee
- Tencent Cloud SDK 3.0 for Go: GitHub Gitee
- Tencent Cloud SDK 3.0 for Node.js: GitHub Gitee
- Tencent Cloud SDK 3.0 for .NET: GitHub Gitee
- Tencent Cloud SDK 3.0 for C++: GitHub Gitee
- Tencent Cloud SDK 3.0 for Ruby: GitHub Gitee

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
InternalError	内部错误。
InvalidParameterValue.ProductTypeNotSupport	产品类型不支持。
ResourceNotFound.DeviceNotExist	设备不存在。
ResourceNotFound.ProductNotExist	产品不存在。
UnsupportedOperation.ClientCertAlreadyGot	设备私钥已被获取。
UnsupportedOperation.WrongProductAuthType	不支持的认证类型。

查询设备资源详情

最近更新时间：2024-03-12 01:33:47

1. 接口描述

接口请求域名： iotcloud.tencentcloudapi.com 。

本接口（DescribeDeviceResource）用于查询设备资源详情。

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer　点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：DescribeDeviceResource。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	是	String	公共参数，详见产品支持的 地域列表。
DeviceName	是	String	设备名称 示例值：test
ProductID	否	String	产品ID 示例值：AAAAAAAAAA
Name	否	String	具体的设备资源名称 示例值：test

3. 输出参数

参数名称	类型	描述
Result	DeviceResourceInfo	设备资源详情
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 查询设备资源详情

输入示例

```
https://iotcloud.tencentcloudapi.com/?Action=DescribeDeviceResource
&ProductID=ABCDE12345
&Name=test
```

```
&DeviceName=test
&<公共请求参数>
```

输出示例


```
{
"Response": {
"Result": {
"Status": 1,
"DeviceName": "xx",
"UpdateTime": "xx",
"Name": "xx",
"Percent": 1,
"ProductName": "xx",
"Md5": "xx",
"ProductID": "xx",
"Size": 1
},
"RequestId": "xx"
}
}
```


5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python: GitHub Gitee
- Tencent Cloud SDK 3.0 for Java: GitHub Gitee
- Tencent Cloud SDK 3.0 for PHP: GitHub Gitee
- Tencent Cloud SDK 3.0 for Go: GitHub Gitee
- Tencent Cloud SDK 3.0 for Node.js: GitHub Gitee
- Tencent Cloud SDK 3.0 for .NET: GitHub Gitee
- Tencent Cloud SDK 3.0 for C++: GitHub Gitee
- Tencent Cloud SDK 3.0 for Ruby: GitHub Gitee

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
InternalError	内部错误。
InvalidParameterValue	参数取值错误。
ResourceNotFound.DeviceResourceNotExist	设备资源不存在。
ResourceNotFound.ProductNotExist	产品不存在。

获取设备资源列表

最近更新时间：2024-03-12 01:33:47

1. 接口描述

接口请求域名： iotcloud.tencentcloudapi.com 。

本接口（DescribeDeviceResources）用于查询设备资源列表。

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer　点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：DescribeDeviceResources。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	是	String	公共参数，详见产品支持的 地域列表。
Offset	是	Integer	偏移量，Offset从0开始 示例值：0
Limit	是	Integer	分页的大小，数值范围 10-250 示例值：10
ProductID	否	String	产品ID 示例值：AAAAAAAAAA
DeviceName	否	String	需要过滤的设备名称 示例值：test
StartTime	否	String	资源搜索开始时间 示例值：2006-01-02 15:04:05
EndTime	否	String	资源搜索结束时间 示例值：2006-01-02 15:04:05

3. 输出参数

参数名称	类型	描述
TotalCount	Integer	资源总数 示例值：10
Result	Array of DeviceResourceInfo	资源列表 注意：此字段可能返回 null，表示取不到有效值。

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 获取设备资源列表

输入示例

```
https://iotcloud.tencentcloudapi.com/?Action=DescribeDeviceResources
&ProductID=ABCDE12345
&Offset=0
&Limit=10
&DeviceName=test
&StartTime=xxx
&EndTime=xxx
&<公共请求参数>
```

输出示例

```
{
"Response": {
"TotalCount": 1,
"Result": [
{
"Name": "xx",
"ProductName": "xx",
"Md5": "xx",
"DeviceName": "xx",
"ProductID": "xx",
"UpdateTime": "xx",
"Size": 1
}
],
"RequestId": "xx"
}
}
```

5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python: GitHub Gitee
- Tencent Cloud SDK 3.0 for Java: GitHub Gitee
- Tencent Cloud SDK 3.0 for PHP: GitHub Gitee
- Tencent Cloud SDK 3.0 for Go: GitHub Gitee
- Tencent Cloud SDK 3.0 for Node.js: GitHub Gitee
- Tencent Cloud SDK 3.0 for .NET: GitHub Gitee
- Tencent Cloud SDK 3.0 for C++: GitHub Gitee
- Tencent Cloud SDK 3.0 for Ruby: GitHub Gitee

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
InternalError	内部错误。
InternalError.DBOperationError	数据库内部错误。
InvalidParameterValue	参数取值错误。
ResourceNotFound.ProductNotExist	产品不存在。

获取设备列表

最近更新时间：2024-03-12 01:33:47

1. 接口描述

接口请求域名： iotcloud.tencentcloudapi.com 。

本接口（DescribeDevices）用于查询物联网通信设备的设备列表。

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer 点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：DescribeDevices。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	是	String	公共参数，详见产品支持的 地域列表。
ProductId	是	String	需要查看设备列表的产品 ID 示例值：ABCDE12345
Offset	是	Integer	偏移量，Offset从0开始 示例值：0
Limit	是	Integer	分页的大小，数值范围 10-250 示例值：10
FirmwareVersion	否	String	设备固件版本号，若不带此参数会返回所有固件版本的设备。传"None-FirmwareVersion"查询无版本号的设备 示例值：1.0.0
DeviceName	否	String	需要过滤的设备名称 示例值：ABCD
EnableState	否	Integer	设备是否启用，0禁用状态1启用状态，默认不区分 示例值：1

3. 输出参数

参数名称	类型	描述
TotalCount	Integer	设备总数 示例值：1
Devices	Array of DeviceInfo	设备详细信息列表

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 获取设备列表

输入示例

```
https://iotcloud.tencentcloudapi.com/?Action=DescribeDevices
&ProductId=ABCDE12345
&Offset=0
&Limit=10
&FirmwareVersion=1.0.0
&<公共请求参数>
```

输出示例

```
{
"Response": {
"TotalCount": 1,
"RequestId": "xx",
"Devices": [
{
"EnableState": 1,
"LastOfflineTime": 1,
"Version": "xx",
"CertState": 1,
"Online": 1,
"FirmwareUpdateTime": 1,
"DeviceName": "xx",
"Tags": [
{
"Tag": "xx",
"Type": 1,
"Name": "xx",
"Value": "xx"
},
{
"Tag": "xx",
"Type": 1,
"Name": "xx",
"Value": "xx"
}
],
"LogLevel": 1,
"FirstOnlineTime": 1,
"DeviceCert": "xx",
```

```
"Imei": "xx",
"ClientIP": "xx",
"DevicePsk": "xx",
"Isp": 1,
"NbiotDeviceID": "xx",
"LoraDevEui": "xx",
"DeviceType": 1,
"LoginTime": 1,
"ConnIP": 1,
"LastUpdateTime": 1,
"Labels": [
{
"Value": "xx",
"Key": "xx"
}
],
"CreateTime": 1,
"LoraMoteType": 1
}
]
}
}
```


5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python: GitHub Gitee
- Tencent Cloud SDK 3.0 for Java: GitHub Gitee
- Tencent Cloud SDK 3.0 for PHP: GitHub Gitee
- Tencent Cloud SDK 3.0 for Go: GitHub Gitee
- Tencent Cloud SDK 3.0 for Node.js: GitHub Gitee
- Tencent Cloud SDK 3.0 for .NET: GitHub Gitee
- Tencent Cloud SDK 3.0 for C++: GitHub Gitee
- Tencent Cloud SDK 3.0 for Ruby: GitHub Gitee

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
InternalError	内部错误。
InternalError.DBOperationError	数据库内部错误。
InvalidParameterValue	参数取值错误。
ResourceNotFound.ProductNotExist	产品不存在。

获取网关绑定的子设备列表

最近更新时间：2024-03-12 01:33:46

1. 接口描述

接口请求域名： iotcloud.tencentcloudapi.com 。

本接口（DescribeGatewayBindDevices）用于获取网关绑定的子设备列表

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer　点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：DescribeGatewayBindDevices。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	是	String	公共参数，详见产品支持的 地域列表。
GatewayProductId	是	String	网关设备的产品ID 示例值：12345ABCDE
GatewayDeviceName	是	String	网关设备的设备名 示例值：test
Offset	是	Integer	偏移量，Offset从0开始 示例值：0
Limit	是	Integer	分页的页大小 示例值：10
ProductId	否	String	LoRa产品的ID 示例值：ABCDE12345

3. 输出参数

参数名称	类型	描述
TotalCount	Integer	子设备总数 示例值：1
Devices	Array of BindDeviceInfo	子设备信息
ProductName	String	子设备所属的产品名 示例值：test

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 获取网关绑定的子设备列表

输入示例

```
https://iotcloud.tencentcloudapi.com/?Action=DescribeGatewayBindDevices
&GatewayProductId=12345ABCDE
&GatewayDeviceName=test
&Offset=0
&Limit=10
&ProductId=ABCDE12345
&<公共请求参数>
```

输出示例

```
{
"Response": {
"TotalCount": 1,
"Devices": [
{
"ProductId": "ABCDE12345",
"DeviceName": "test",
"BindTime": 1,
"Tags": []
}
],
"ProductName": "test",
"RequestId": "69f65618-600b-4ac4-b8e3-4528a6819078"
}
}
```

5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python: GitHub Gitee
- Tencent Cloud SDK 3.0 for Java: GitHub Gitee
- Tencent Cloud SDK 3.0 for PHP: GitHub Gitee
- Tencent Cloud SDK 3.0 for Go: GitHub Gitee
- Tencent Cloud SDK 3.0 for Node.js: GitHub Gitee
- Tencent Cloud SDK 3.0 for .NET: GitHub Gitee
- Tencent Cloud SDK 3.0 for C++: GitHub Gitee
- Tencent Cloud SDK 3.0 for Ruby: GitHub Gitee

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
InternalError	内部错误。
InternalError.DBOperationError	数据库内部错误。
InvalidParameterValue	参数取值错误。
ResourceNotFound.ProductNotExist	产品不存在。

查询产品资源详情

最近更新时间：2024-03-12 01:33:46

1. 接口描述

接口请求域名： iotcloud.tencentcloudapi.com 。

本接口（DescribeProductResource）用于查询产品资源详情。

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer　点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：DescribeProductResource。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	是	String	公共参数，详见产品支持的 地域列表。
ProductID	否	String	需要查看资源列表的产品 ID 示例值：AAAAAAAAAA
Name	否	String	需要过滤的资源名称 示例值：test

3. 输出参数

参数名称	类型	描述
Result	ProductResourceInfo	资源详情 注意：此字段可能返回 null，表示取不到有效值。
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 查询产品资源详情

输入示例

```
https://iotcloud.tencentcloudapi.com/?Action=DescribeProductResource
&ProductID=ABCDE12345
&Name=test
&<公共请求参数>
```

输出示例

```
{
"Response": {
"Result": {
"Name": "xx",
"ProductName": "xx",
"Md5": "xx",
"Description": "xx",
"ProductID": "xx",
"CreateTime": "xx",
"Size": 1
},
"RequestId": "xx"
}
}
```

5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python: GitHub Gitee
- Tencent Cloud SDK 3.0 for Java: GitHub Gitee
- Tencent Cloud SDK 3.0 for PHP: GitHub Gitee
- Tencent Cloud SDK 3.0 for Go: GitHub Gitee
- Tencent Cloud SDK 3.0 for Node.js: GitHub Gitee
- Tencent Cloud SDK 3.0 for .NET: GitHub Gitee
- Tencent Cloud SDK 3.0 for C++: GitHub Gitee
- Tencent Cloud SDK 3.0 for Ruby: GitHub Gitee

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
InternalError	内部错误。

错误码	描述
InvalidParameterValue	参数取值错误。
ResourceNotFound.ProductNotExist	产品不存在。
ResourceNotFound.ProductResourceNotExist	产品资源不存在。

获取产品资源列表

最近更新时间：2024-03-12 01:33:46

1. 接口描述

接口请求域名： iotcloud.tencentcloudapi.com 。

本接口（DescribeProductResources）用于查询产品资源列表。

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer　点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：DescribeProductResources。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	是	String	公共参数，详见产品支持的 地域列表。
Offset	是	Integer	偏移量，Offset从0开始 示例值：0
Limit	是	Integer	分页的大小，数值范围 10-250 示例值：10
ProductID	否	String	需要查看资源列表的产品 ID 示例值：AAAAAAAAAA
Name	否	String	需要过滤的资源名称 示例值：test

3. 输出参数

参数名称	类型	描述
TotalCount	Integer	资源总数 示例值：10
Result	Array of ProductResourceInfo	资源详情 注意：此字段可能返回 null，表示取不到有效值。
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 获取产品资源列表

输入示例

```
https://iotcloud.tencentcloudapi.com/?Action=DescribeProductResources
&ProductID=ABCDE12345
&Offset=0
&Limit=10
&Name=test
&<公共请求参数>
```

输出示例

```
{
"Response": {
"TotalCount": 1,
"Result": [
{
"Name": "xx",
"ProductName": "xx",
"Md5": "xx",
"Description": "xx",
"ProductID": "xx",
"CreateTime": "xx",
"Size": 1
}
],
"RequestId": "xx"
}
}
```

5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python: GitHub Gitee
- Tencent Cloud SDK 3.0 for Java: GitHub Gitee
- Tencent Cloud SDK 3.0 for PHP: GitHub Gitee
- Tencent Cloud SDK 3.0 for Go: GitHub Gitee
- Tencent Cloud SDK 3.0 for Node.js: GitHub Gitee
- Tencent Cloud SDK 3.0 for .NET: GitHub Gitee
- Tencent Cloud SDK 3.0 for C++: GitHub Gitee

- Tencent Cloud SDK 3.0 for Ruby: GitHub Gitee

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
FailedOperation.ProductResourceDuplicate	同名产品资源已存在。
InternalError	内部错误。
InvalidParameterValue	参数取值错误。
ResourceNotFound.ProductNotExist	产品不存在。
ResourceNotFound.ProductResourceNotExist	产品资源不存在。

重置设备状态

最近更新时间：2024-03-12 01:33:46

1. 接口描述

接口请求域名： iotcloud.tencentcloudapi.com 。

重置设备的连接状态

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer 点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：ResetDeviceState。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	是	String	公共参数，详见产品支持的 地域列表。
ProductId	是	String	产品ID 示例值：ABCDE12345
DeviceNames.N	是	Array of String	设备名称 示例值：["test_device1\n","test_device2\n"]

3. 输出参数

参数名称	类型	描述
SuccessCount	Integer	批量重置设备成功数 示例值：1
ResetDeviceResults	Array of ResetDeviceResult	批量重置设备结果
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 重置设备连接状态

输入示例

```
https://iotcloud.tencentcloudapi.com/?Action=ResetDeviceState
&ProductId=ABCDE12345
```

```
&DeviceNames.0=test_device1
&DeviceNames.1=test_device2
&<公共请求参数>
```

输出示例

```
{
"Response": {
"RequestId": "54f75f05-a87c-45fc-9520-6b59e251e91c",
"SuccessCount": 1,
"ResetDeviceResults": [
{
"DeviceName": "test_device1",
"Success": true,
"Reason": ""
},
{
"DeviceName": "test_device2",
"Success": false,
"Reason": "device not exists"
}
]
}
}
```

5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python: GitHub Gitee
- Tencent Cloud SDK 3.0 for Java: GitHub Gitee
- Tencent Cloud SDK 3.0 for PHP: GitHub Gitee
- Tencent Cloud SDK 3.0 for Go: GitHub Gitee
- Tencent Cloud SDK 3.0 for Node.js: GitHub Gitee
- Tencent Cloud SDK 3.0 for .NET: GitHub Gitee
- Tencent Cloud SDK 3.0 for C++: GitHub Gitee
- Tencent Cloud SDK 3.0 for Ruby: GitHub Gitee

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
InternalError	内部错误。
InvalidParameterValue.ProductTypeNotSupport	产品类型不支持。
ResourceNotFound.ProductNotExist	产品不存在。
UnauthorizedOperation.DeviceIsNotEnabled	设备未启用。

批量解绑子设备

最近更新时间：2024-03-12 01:33:46

1. 接口描述

接口请求域名： iotcloud.tencentcloudapi.com 。

本接口（UnbindDevices）用于网关设备批量解绑子设备

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer　点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：UnbindDevices。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	是	String	公共参数，详见产品支持的 地域列表。
GatewayProductId	是	String	网关设备的产品ID 示例值：12345ABCDE
GatewayDeviceName	是	String	网关设备的设备名 示例值：test
ProductId	是	String	产品ID 示例值：ABCDE12345
DeviceNames.N	是	Array of String	多个设备名 示例值：["test0\n","test1\n"]
Skey	否	String	中兴CLAA设备的解绑需要Skey，普通设备不需要 示例值：123

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 批量解绑子设备

输入示例

```
https://iotcloud.tencentcloudapi.com/?Action=UnbindDevices
&GatewayProductId=12345ABCDE
&GatewayDeviceName=test
&ProductId=ABCDE12345
&DeviceNames.0=test0
&DeviceNames.1=test1
&<公共请求参数>
```

输出示例

```
{
"Response": {
"RequestId": "69f65618-600b-4ac4-b8e3-4528a6819078"
}
}
```

5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python: GitHub Gitee
- Tencent Cloud SDK 3.0 for Java: GitHub Gitee
- Tencent Cloud SDK 3.0 for PHP: GitHub Gitee
- Tencent Cloud SDK 3.0 for Go: GitHub Gitee
- Tencent Cloud SDK 3.0 for Node.js: GitHub Gitee
- Tencent Cloud SDK 3.0 for .NET: GitHub Gitee
- Tencent Cloud SDK 3.0 for C++: GitHub Gitee
- Tencent Cloud SDK 3.0 for Ruby: GitHub Gitee

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
InternalError	内部错误。
InternalError.DBOperationError	数据库内部错误。

错误码	描述
InvalidParameterValue	参数取值错误。
ResourceNotFound.DeviceNotExist	设备不存在。

切换设备可用状态

最近更新时间：2024-03-12 01:33:46

1. 接口描述

接口请求域名： iotcloud.tencentcloudapi.com 。

启用或者禁用设备

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer

点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：UpdateDeviceAvailableState。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	是	String	公共参数，详见产品支持的 地域列表。
ProductId	是	String	设备所属产品id 示例值：SB9OJFCJ1C
DeviceName	是	String	设备名称 示例值：test123
EnableState	是	Integer	要设置的设备状态，1为启用，0为禁用 示例值：0

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 UpdateDeviceAvailableState

输入示例

```
https://iotcloud.tencentcloudapi.com/?Action=UpdateDeviceAvailableState
&ProductId=SB9OJFCJ1C
&DeviceName=test123
```

```
&EnableState=0
&<公共请求参数>
```

输出示例

```
{
"Response": {
"RequestId": "69f65618-600b-4ac4-b8e3-4528a6819078"
}
}
```

5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python: GitHub Gitee
- Tencent Cloud SDK 3.0 for Java: GitHub Gitee
- Tencent Cloud SDK 3.0 for PHP: GitHub Gitee
- Tencent Cloud SDK 3.0 for Go: GitHub Gitee
- Tencent Cloud SDK 3.0 for Node.js: GitHub Gitee
- Tencent Cloud SDK 3.0 for .NET: GitHub Gitee
- Tencent Cloud SDK 3.0 for C++: GitHub Gitee
- Tencent Cloud SDK 3.0 for Ruby: GitHub Gitee

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
InternalError	内部错误。
ResourceNotFound.DeviceNotExist	设备不存在。
ResourceNotFound.ProductNotExist	产品不存在。
UnauthorizedOperation.DeviceHasAlreadyBindGateway	该设备绑定了网关设备，无法删除。
UnauthorizedOperation.ProductIsForbidden	产品禁用了该功能。

更新设备日志级别

最近更新时间：2024-03-12 01:33:46

1. 接口描述

接口请求域名： iotcloud.tencentcloudapi.com 。

设置设备上报的日志级别

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer 点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：UpdateDeviceLogLevel。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	是	String	公共参数，详见产品支持的 地域列表。
ProductId	是	String	产品ID 示例值：ABCDE12345
DeviceName	是	String	设备名称 示例值：abc
LogLevel	是	Integer	日志级别，0：关闭，1：错误，2：告警，3：信息，4：调试 示例值：1

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 更新设备日志级别

输入示例

```
https://iotcloud.tencentcloudapi.com/?Action=UpdateDeviceLogLevel
&ProductId=ABCDE12345
&DeviceName=abc
```

```
&LogLevel=1
&<公共请求参数>
```

输出示例

```
{
"Response": {
"RequestId": "9e574269-093f-4a7f-bf90-24ef80b6528a"
}
}
```

5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python: GitHub Gitee
- Tencent Cloud SDK 3.0 for Java: GitHub Gitee
- Tencent Cloud SDK 3.0 for PHP: GitHub Gitee
- Tencent Cloud SDK 3.0 for Go: GitHub Gitee
- Tencent Cloud SDK 3.0 for Node.js: GitHub Gitee
- Tencent Cloud SDK 3.0 for .NET: GitHub Gitee
- Tencent Cloud SDK 3.0 for C++: GitHub Gitee
- Tencent Cloud SDK 3.0 for Ruby: GitHub Gitee

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
InternalError	内部错误。
InternalError.DBOperationError	数据库内部错误。
ResourceNotFound.DeviceNotExist	设备不存在。
UnauthorizedOperation.DeviceIsNotEnabled	设备未启用。

更新设备PSK

最近更新时间：2024-03-12 01:33:45

1. 接口描述

接口请求域名： iotcloud.tencentcloudapi.com 。

本接口（UpdateDevicePSK）用于更新设备的PSK

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer 点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：UpdateDevicePSK。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	是	String	公共参数，详见产品支持的 地域列表。
ProductId	是	String	产品名 示例值：ABCDE12345
DeviceName	是	String	设备名 示例值：test_device
Psk	是	String	设备的psk 示例值：VOQmAku7T1haYaiV7LJfmg==

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 更新设备PSK

输入示例

```
https://iotcloud.tencentcloudapi.com/?Action=UpdateDevicePSK
&ProductId=ABCDE12345
&DeviceName=test_device
```

```
&Psk=xxxxxxxxxxxxxxxxx
&<公共请求参数>
```

输出示例

```
{
"Response": {
"RequestId": "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx"
}
}
```

5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python: GitHub Gitee
- Tencent Cloud SDK 3.0 for Java: GitHub Gitee
- Tencent Cloud SDK 3.0 for PHP: GitHub Gitee
- Tencent Cloud SDK 3.0 for Go: GitHub Gitee
- Tencent Cloud SDK 3.0 for Node.js: GitHub Gitee
- Tencent Cloud SDK 3.0 for .NET: GitHub Gitee
- Tencent Cloud SDK 3.0 for C++: GitHub Gitee
- Tencent Cloud SDK 3.0 for Ruby: GitHub Gitee

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
InternalError	内部错误。
InternalError.DBOperationError	数据库内部错误。
InvalidParameterValue	参数取值错误。
ResourceNotFound.DeviceNotExist	设备不存在。
UnauthorizedOperation.ProductNotSupportPSK	产品不支持密钥认证。

批量切换设备可用状态

最近更新时间：2024-03-12 01:33:45

1. 接口描述

接口请求域名： iotcloud.tencentcloudapi.com 。

批量启用或者禁用设备

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer

点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：UpdateDevicesEnableState。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	是	String	公共参数，详见产品支持的 地域列表。
ProductId	是	String	设备所属产品id 示例值：SB9OJFCJ1C
DeviceNames.N	是	Array of String	设备名称集合 示例值：["test123\n"]
Status	是	Integer	要设置的设备状态，1为启用，0为禁用 示例值：1

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 UpdateDevicesEnableState

输入示例

```
https://iotcloud.tencentcloudapi.com/?Action=UpdateDevicesEnableState
&ProductId=SB9OJFCJ1C
&DeviceNames.0=test123
```

```
&Status=1
&<公共请求参数>
```

输出示例

```
{
"Response": {
"RequestId": "69f65618-600b-4ac4-b8e3-4528a6819078"
}
}
```

5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python: GitHub Gitee
- Tencent Cloud SDK 3.0 for Java: GitHub Gitee
- Tencent Cloud SDK 3.0 for PHP: GitHub Gitee
- Tencent Cloud SDK 3.0 for Go: GitHub Gitee
- Tencent Cloud SDK 3.0 for Node.js: GitHub Gitee
- Tencent Cloud SDK 3.0 for .NET: GitHub Gitee
- Tencent Cloud SDK 3.0 for C++: GitHub Gitee
- Tencent Cloud SDK 3.0 for Ruby: GitHub Gitee

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
InternalError	内部错误。
InternalError.DBOperationError	数据库内部错误。
InvalidParameterValue.ProductTypeNotSupport	产品类型不支持。
ResourceNotFound.DeviceNotExist	设备不存在。
ResourceNotFound.ProductNotExist	产品不存在。

错误码	描述
UnauthorizedOperation.DeviceHasAlreadyBindGateway	该设备绑定了网关设备，无法删除。
UnauthorizedOperation.ProductIsForbidden	产品禁用了该功能。

产品相关接口
创建产品

最近更新时间：2024-03-12 01:33:55

1. 接口描述

接口请求域名： iotcloud.tencentcloudapi.com 。

本接口（CreateProduct）用于创建一个新的物联网通信产品

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer　点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：CreateProduct。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	是	String	公共参数，详见产品支持的 地域列表。
ProductName	是	String	产品名称，名称不能和已经存在的产品名称重复。命名规则：[a-zA-Z0-9:_-]{1,32} 示例值：fruit
ProductProperties	否	ProductProperties	产品属性
Skey	否	String	创建CLAA产品时，需要Skey 示例值：xxx

3. 输出参数

参数名称	类型	描述
ProductName	String	产品名称 示例值：fruit
ProductId	String	产品 ID，腾讯云生成全局唯一 ID 示例值：ABCDE12345
ProductProperties	ProductProperties	产品属性
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 创建产品

输入示例

```
https://iotcloud.tencentcloudapi.com/?Action=CreateProduct
&ProductName=fruit
&ProductProperties.ProductDescription=test
&ProductProperties.EncryptionType=1
&ProductProperties.Region=gz
&ProductProperties.ProductType=0
&ProductProperties.Format=json
&<公共请求参数>
```

输出示例

```
{
"Response": {
"ProductId": "ABCDE12345",
"ProductName": "fruit",
"ProductProperties": {
"ProductDescription": "test",
"EncryptionType": 1,
"Region": "gz",
"ProductType": 0,
"Format": "json",
"Platform": "DEFAULT",
"Appeui": ""
},
"RequestId": "xxxxxxxxxxxxxxxxxxxxxx"
}
}
```

5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python: GitHub Gitee
- Tencent Cloud SDK 3.0 for Java: GitHub Gitee
- Tencent Cloud SDK 3.0 for PHP: GitHub Gitee
- Tencent Cloud SDK 3.0 for Go: GitHub Gitee

- Tencent Cloud SDK 3.0 for Node.js: GitHub Gitee
- Tencent Cloud SDK 3.0 for .NET: GitHub Gitee
- Tencent Cloud SDK 3.0 for C++: GitHub Gitee
- Tencent Cloud SDK 3.0 for Ruby: GitHub Gitee

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
FailedOperation.AccountIsolated	操作失败，账号已欠费隔离。
InternalError	内部错误。
InternalError.DBOperationError	数据库内部错误。
InvalidParameterValue	参数取值错误。
InvalidParameterValue.ProductAlreadyExist	创建的产品名已存在。
InvalidParameterValue.ProductTypeNotSupport	产品类型不支持。
InvalidParameterValue.TidProductAlreadyExist	该TID产品已存在。
LimitExceeded.ProductExceedLimit	超过产品数量限制。
ResourceNotFound.ThingModelNotExist	物模型不存在。
UnauthorizedOperation.UserNotAuthenticaed	用户未通过实名认证。

创建Topic

最近更新时间：2024-03-12 01:33:54

1. 接口描述

接口请求域名： iotcloud.tencentcloudapi.com 。

本接口（CreateTopicPolicy）用于创建一个Topic

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer　点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：CreateTopicPolicy。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	是	String	公共参数，详见产品支持的 地域列表。
ProductId	是	String	产品自身ID 示例值：ABCDE12345
TopicName	是	String	Topic名称 示例值：abc
Privilege	是	Integer	Topic权限，1发布，2订阅，3订阅和发布 示例值：2
BrokerSubscribe	否	BrokerSubscribe	代理订阅信息，网关产品为绑定的子产品创建topic时需要填写，内容为子产品的ID和设备信息。

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 创建Topic示例

输入示例

```
https://iotcloud.tencentcloudapi.com/?Action=CreateTopicPolicy
&ProductId=ABCDE12345
```

```
&TopicName=abc
&Privilege=2
&BrokerSubscribe.ProductId=11LAWZ3J2D
&BrokerSubscribe.DeviceName=device1
&<公共请求参数>
```

输出示例

```
{
"Response": {
"RequestId": "be69a7a3-7315-40a7-9532-3316e4a3e97e"
}
}
```

5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python: GitHub Gitee
- Tencent Cloud SDK 3.0 for Java: GitHub Gitee
- Tencent Cloud SDK 3.0 for PHP: GitHub Gitee
- Tencent Cloud SDK 3.0 for Go: GitHub Gitee
- Tencent Cloud SDK 3.0 for Node.js: GitHub Gitee
- Tencent Cloud SDK 3.0 for .NET: GitHub Gitee
- Tencent Cloud SDK 3.0 for C++: GitHub Gitee
- Tencent Cloud SDK 3.0 for Ruby: GitHub Gitee

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
FailedOperation.ProductNotBind	产品未绑定，无法代理订阅。
InternalError	内部错误。
InvalidParameterValue	参数取值错误。

错误码	描述
InvalidParameterValue.TopicPolicyAlreadyExist	Topic已存在。
LimitExceeded.TopicPolicyExceedLimit	Topic数量超出限制。
ResourceNotFound.ProductNotExist	产品不存在。

删除产品

最近更新时间：2024-03-12 01:33:54

1. 接口描述

接口请求域名： iotcloud.tencentcloudapi.com 。

本接口（DeleteProduct）用于删除一个物联网通信产品

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer 点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：DeleteProduct。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	是	String	公共参数，详见产品支持的 地域列表。
ProductId	是	String	需要删除的产品 ID 示例值：ABCDE12345
Skey	否	String	删除LoRa产品需要skey 示例值：xxx

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 删除产品

输入示例

```
https://iotcloud.tencentcloudapi.com/?Action=DeleteProduct
&ProductId=ABCDE12345
&<公共请求参数>
```

输出示例

```
{
"Response": {
"RequestId": "xxxxxxxxxxxxxxxxxxxxxxx"
}
}
```

5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python: GitHub Gitee
- Tencent Cloud SDK 3.0 for Java: GitHub Gitee
- Tencent Cloud SDK 3.0 for PHP: GitHub Gitee
- Tencent Cloud SDK 3.0 for Go: GitHub Gitee
- Tencent Cloud SDK 3.0 for Node.js: GitHub Gitee
- Tencent Cloud SDK 3.0 for .NET: GitHub Gitee
- Tencent Cloud SDK 3.0 for C++: GitHub Gitee
- Tencent Cloud SDK 3.0 for Ruby: GitHub Gitee

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
InternalError	内部错误。
InternalError.DBOperationError	数据库内部错误。
InvalidParameterValue	参数取值错误。
ResourceNotFound.ProductNotExist	产品不存在。
UnauthorizedOperation.DeleteTidFail	该产品已存在TID申请，禁止删除。
UnauthorizedOperation.DevicesExistUnderProduct	删除的产品下还包括未删除的设备。
UnsupportedOperation.GatewayProductHasBindedProduct	网关产品下存在绑定的子产品，无法删除。
UnsupportedOperation.ProductHasBindGateway	存在网关设备绑定当前产品，无法删除。
UnsupportedOperation.ProductHasBindedGatewayProduct	产品存在绑定的网关产品，无法删除。

删除产品的私有CA证书

最近更新时间：2024-03-12 01:33:54

1. 接口描述

接口请求域名： iotcloud.tencentcloudapi.com 。

删除产品的私有CA证书

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer 点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：DeleteProductPrivateCA。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	是	String	公共参数，详见产品支持的 地域列表。
ProductId	是	String	产品ID 示例值：ABCDE2345

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 删除产品的私有CA证书

删除产品的私有CA证书

输入示例

```
https://iotcloud.tencentcloudapi.com/?Action=DeleteProductPrivateCA
&ProductId=ABCDE12345
&<公共请求参数>
```

输出示例

```
{
"Response": {
"RequestId": "xxxxxxxxxx"
}
}
```

5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python: GitHub Gitee
- Tencent Cloud SDK 3.0 for Java: GitHub Gitee
- Tencent Cloud SDK 3.0 for PHP: GitHub Gitee
- Tencent Cloud SDK 3.0 for Go: GitHub Gitee
- Tencent Cloud SDK 3.0 for Node.js: GitHub Gitee
- Tencent Cloud SDK 3.0 for .NET: GitHub Gitee
- Tencent Cloud SDK 3.0 for C++: GitHub Gitee
- Tencent Cloud SDK 3.0 for Ruby: GitHub Gitee

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
InvalidParameterValue.CACertInvalid	CA证书内容错误。

查看产品详情

最近更新时间：2024-03-12 01:33:54

1. 接口描述

接口请求域名： iotcloud.tencentcloudapi.com 。

本接口（DescribeProduct）用于查看产品详情

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer 点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：DescribeProduct。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	是	String	公共参数，详见产品支持的 地域列表。
ProductId	是	String	产品ID 示例值：ABCDE12345

3. 输出参数

参数名称	类型	描述
ProductId	String	产品ID 示例值：ABCDE12345
ProductName	String	产品名 示例值：Test_1
ProductMetadata	ProductMetadata	产品元数据
ProductProperties	ProductProperties	产品属性
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 查看产品详情示例

输入示例

```
https://iotcloud.tencentcloudapi.com/?Action=DescribeProduct
&ProductId=ABCDE12345
&<公共请求参数>
```

输出示例


```
{
"Response": {
"ProductMetadata": {
"CreationDate": 1509453755000
},
"ProductProperties": {
"ProductDescription": "description1"
},
"ProductName": "Test_1",
"ProductId": "ABCDE12345",
"RequestId": "8e0b3665-cfb5-4077-a535-0ed7f970cf3b"
}
}
```


5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python: GitHub Gitee
- Tencent Cloud SDK 3.0 for Java: GitHub Gitee
- Tencent Cloud SDK 3.0 for PHP: GitHub Gitee
- Tencent Cloud SDK 3.0 for Go: GitHub Gitee
- Tencent Cloud SDK 3.0 for Node.js: GitHub Gitee
- Tencent Cloud SDK 3.0 for .NET: GitHub Gitee
- Tencent Cloud SDK 3.0 for C++: GitHub Gitee
- Tencent Cloud SDK 3.0 for Ruby: GitHub Gitee

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
InternalError	内部错误。
InternalError.DBOperationError	数据库内部错误。
InvalidParameterValue	参数取值错误。
ResourceNotFound.ProductNotExist	产品不存在。

查询产品绑定的CA证书

最近更新时间：2024-03-12 01:33:54

1. 接口描述

接口请求域名： iotcloud.tencentcloudapi.com 。

查询产品绑定的CA证书

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer

点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：DescribeProductCA。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	是	String	公共参数，详见产品支持的 地域列表。
ProductId	是	String	产品ID 示例值：ABCDE2345

3. 输出参数

参数名称	类型	描述
CAs	Array of CertInfo	CA证书列表
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 查询产品的CA证书

输入示例

```
https://iotcloud.tencentcloudapi.com/?Action=DescribeProductCA
&ProductId=ABCDE12345
&<公共请求参数>
```

输出示例

```
{
"Response": {
"RequestId": "xxxxxxxxxxxxxxxx",
"CAs": [
{
"CreateTime": 1622619674,
"EffectiveTime": 1622448592,
"ExpireTime": 1653984592,
"CertText": "-----BEGIN CERTIFICATE-----\nMIIDgTCCAmmgAwIBAgIUX/aeTIr85dbejTlbNGcpRPW0dIowDQYJKoZIhvcNAQEL\nBQAwU
DELMAkGA1UEBhMCQ04xEjAQBgNVBAgMCWd1YW5nZG9uZzERMA8GA1UEBwwIc2hlbnpoZW4xDDAKBgNVBAoMA0FBQTEMMA
oGA1UEAwwDQUFBMB4XDTIxMDUzMTA4\nMDk1MloXDTIyMDUzMTA4MDk1MlowUDELMAkGA1UEBhMCQ04xEjAQBgNVBAgMCWd
1\nYW5nZG9uZzERMA8GA1UEBwwIc2hlbnpoZW4xDDAKBgNVBAoMA0FBQTEMMAoGA1UE\nAwwDQUFBMIIBIjANBgkqhkiG9w0BA
QEFAAOCAQ8AMIIBCgKCAQEA65Rt1X/RBdG6\n24JYIJrsYOC8NALr/i7jA13v3EHEOjA2505G3YM7RL/NeHIMD7m3XYH2lqHv9XVU\nHz
d+stK5MXXmVSIazt2BYaTdIzWOl0sPQGU9BaIloZyb1nSR/UAZ7sJpon5+nTvV\nNUUnkVC36BgWUKfLOTlmKkzlHOLecD/WAzgIJ55N
C6QsjKWiKXBm5mtlz4uoZ7Bh\nhQGM/8Zax1YLYnMwkQrMB/o8ma/o5/wRpdqKT0ixm2yZMxW3c6XHGpEioowbJnHa\nH4AZm8LAVI
H+TtGZoeKRppTfEArlszocuxZHTfk5XJZh0NsofmwKm4BPmMzW+9tF\nRvEwnnES1QIDAQABo1MwUTAdBgNVHQ4EFgQU0wRl/eNy
5y9eF1xcJlosQoyTAHsw\nHwYDVR0jBBgwFoAU0wRl/eNy5y9eF1xcJlosQoyTAHswDwYDVR0TAQH/BAUwAwEB\n/zANBgkqhkiG9w0B
AQsFAAOCAQEAO9P5UH8lf1Qb/Za4M1gwCyIlVtexON7qOk5Y\nqWIPvhG+fpqeA/fJQq/3LnKbL2b8Dm/SUFEUAsZs/MptXC5d7E++M
wDaiVVQ5rNy\nemHpNrgHXoPZ9JdB4plFWF4K8CvIcLEmlyG6tj9mBbQ/toBqHpGdkaGTQMP/UjxQ\nbZFrV9YiRodEQHfEXD5ZXwvt3
VZsfIbz8gf+flAanx8Ce1EeaZDbQZuqbRHt4FKS\nJoasx3KICfdGocM6PGA7smAYc7MFszAS4tGS9H75EZqNZdseKsct9vP3TCb4hE5x
\nupDam8V9w/SQ9vMGzeW1FM91BfWuRXgsv/Bz4FQdeV1+Xyf+Eg==\n-----END CERTIFICATE-----",
"CertName": "XKFAWDE6LX",
"CertSN": "5ff69e4c8afce5d6de8d395b34672944f5b4765a",
"IssuerName": "CN=AAA,O=AAA,L=shenzhen,ST=guangdong,C=CN",
"Subject": "CN=AAA,O=AAA,L=shenzhen,ST=guangdong,C=CN"
}
]
}
}
```


5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python: GitHub Gitee
- Tencent Cloud SDK 3.0 for Java: GitHub Gitee
- Tencent Cloud SDK 3.0 for PHP: GitHub Gitee
- Tencent Cloud SDK 3.0 for Go: GitHub Gitee
- Tencent Cloud SDK 3.0 for Node.js: GitHub Gitee
- Tencent Cloud SDK 3.0 for .NET: GitHub Gitee

- Tencent Cloud SDK 3.0 for C++: GitHub Gitee
- Tencent Cloud SDK 3.0 for Ruby: GitHub Gitee

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

该接口暂无业务逻辑相关的错误码，其他错误码详见 公共错误码。

获取产品列表

最近更新时间：2024-03-12 01:33:53

1. 接口描述

接口请求域名： iotcloud.tencentcloudapi.com 。

本接口（DescribeProducts）用于列出产品列表。

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer 点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：DescribeProducts。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	是	String	公共参数，详见产品支持的 地域列表。
Offset	是	Integer	偏移量，Offset从0开始 示例值：0
Limit	是	Integer	分页大小，当前页面中显示的最大数量，值范围 10-250。 示例值：10

3. 输出参数

参数名称	类型	描述
TotalCount	Integer	产品总数 示例值：1
Products	Array of ProductInfo	产品详细信息列表
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 获取产品列表

输入示例

```
https://iotcloud.tencentcloudapi.com/?Action=DescribeProducts
&Offset=0
```

```
&Limit=10
&<公共请求参数>
```

输出示例


```
{
"Response": {
"Products": [
{
"ProductId": "ABCDEF12345",
"ProductName": "hello",
"ProductMetadata": {
"CreationDate": 1529049275
},
"ProductProperties": {
"ProductDescription": "test",
"EncryptionType": "1",
"Region": "gz",
"ProductType": 0,
"Format": "json"
}
}
],
"TotalCount": 1,
"RequestId": "69f65618-600b-4ac4-b8e3-4528a6819078"
}
}
```


5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python: GitHub Gitee
- Tencent Cloud SDK 3.0 for Java: GitHub Gitee
- Tencent Cloud SDK 3.0 for PHP: GitHub Gitee
- Tencent Cloud SDK 3.0 for Go: GitHub Gitee
- Tencent Cloud SDK 3.0 for Node.js: GitHub Gitee
- Tencent Cloud SDK 3.0 for .NET: GitHub Gitee
- Tencent Cloud SDK 3.0 for C++: GitHub Gitee
- Tencent Cloud SDK 3.0 for Ruby: GitHub Gitee

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
InternalError	内部错误。
InternalError.DBOperationError	数据库内部错误。
InvalidParameterValue	参数取值错误。

批量设置产品禁用状态

最近更新时间：2024-03-12 01:33:53

1. 接口描述

接口请求域名： iotcloud.tencentcloudapi.com 。

批量设置产品禁用状态

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer 点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：SetProductsForbiddenStatus。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	是	String	公共参数，详见产品支持的 地域列表。
ProductId.N	是	Array of String	要设置禁用状态的产品列表 示例值：["productID1","productID2"]
Status	是	Integer	0启用，1禁用 示例值：1

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 批量设置产品禁用状态

输入示例

```
POST / HTTP/1.1
Host: iotcloud.tencentcloudapi.com
Content-Type: application/json
X-TC-Action: SetProductsForbiddenStatus
<公共请求参数>
```

```
{
"ProductId": [
"productID1",
"productID2"
],
"Status": 1
}
```

输出示例

```
{
"Response": {
"RequestId": "be69a7a3-7315-40a7-9532-3316e4a3e97e"
}
}
```

5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python: GitHub Gitee
- Tencent Cloud SDK 3.0 for Java: GitHub Gitee
- Tencent Cloud SDK 3.0 for PHP: GitHub Gitee
- Tencent Cloud SDK 3.0 for Go: GitHub Gitee
- Tencent Cloud SDK 3.0 for Node.js: GitHub Gitee
- Tencent Cloud SDK 3.0 for .NET: GitHub Gitee
- Tencent Cloud SDK 3.0 for C++: GitHub Gitee
- Tencent Cloud SDK 3.0 for Ruby: GitHub Gitee

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
FailedOperation.AccountIsolated	操作失败，账号已欠费隔离。
InternalError	内部错误。

错误码	描述
InvalidParameter	参数错误。
ResourceNotFound.ProductNotExist	产品不存在。

更新产品动态注册

最近更新时间：2024-03-12 01:33:53

1. 接口描述

接口请求域名： iotcloud.tencentcloudapi.com 。

更新产品动态注册的配置

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer 点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：UpdateProductDynamicRegister。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	是	String	公共参数，详见产品支持的 地域列表。
ProductId	是	String	产品Id 示例值：ABCDE12345
RegisterType	是	Integer	动态注册类型，0-关闭 1-预创建设备 2-自动创建设备 示例值：0
RegisterLimit	是	Integer	动态注册设备上限 示例值：10

3. 输出参数

参数名称	类型	描述
RegisterType	Integer	动态注册类型，0-关闭 1-预创建设备 2-自动创建设备 示例值：0
ProductSecret	String	动态注册产品密钥 示例值：xxxx
RegisterLimit	Integer	动态注册设备上限 示例值：10000
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 更新产品动态注册

更新产品动态注册

输入示例

```
https://iotcloud.tencentcloudapi.com/?Action=UpdateProductDynamicRegister
&ProductId=ABCDE12345
&RegisterType=0
&RegisterLimit=10
&<公共请求参数>
```

输出示例

```
{
"Response": {
"RegisterType": 0,
"ProductSecret": "xxxx",
"RegisterLimit": 10000,
"RequestId": "d15b72a9-ab2b-4906-9632-52f7a31932a9"
}
}
```

5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python: GitHub Gitee
- Tencent Cloud SDK 3.0 for Java: GitHub Gitee
- Tencent Cloud SDK 3.0 for PHP: GitHub Gitee
- Tencent Cloud SDK 3.0 for Go: GitHub Gitee
- Tencent Cloud SDK 3.0 for Node.js: GitHub Gitee
- Tencent Cloud SDK 3.0 for .NET: GitHub Gitee
- Tencent Cloud SDK 3.0 for C++: GitHub Gitee
- Tencent Cloud SDK 3.0 for Ruby: GitHub Gitee

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
InternalError	内部错误。
InvalidParameterValue.ProductTypeNotSupport	产品类型不支持。
ResourceNotFound.ProductNotExist	产品不存在。
UnauthorizedOperation.ProductIsForbidden	产品禁用了该功能。

更新产品的私有CA

最近更新时间：2024-03-12 01:33:53

1. 接口描述

接口请求域名： iotcloud.tencentcloudapi.com 。

更新产品的私有CA

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer 点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：UpdateProductPrivateCA。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	是	String	公共参数，详见产品支持的 地域列表。
ProductId	是	String	产品ID 示例值：ABCDE2345
CertName	是	String	私有CA证书名称 示例值：CertName

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 更新产品的私有CA证书

更新产品的私有CA证书

输入示例

```
https://iotcloud.tencentcloudapi.com/?Action=UpdateProductPrivateCA
&ProductId=ABCDE12345
&CertName=CertName
&<公共请求参数>
```

输出示例

```
{
"Response": {
"RequestId": "xxxxxxxxxx"
}
}
```

5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python: GitHub Gitee
- Tencent Cloud SDK 3.0 for Java: GitHub Gitee
- Tencent Cloud SDK 3.0 for PHP: GitHub Gitee
- Tencent Cloud SDK 3.0 for Go: GitHub Gitee
- Tencent Cloud SDK 3.0 for Node.js: GitHub Gitee
- Tencent Cloud SDK 3.0 for .NET: GitHub Gitee
- Tencent Cloud SDK 3.0 for C++: GitHub Gitee
- Tencent Cloud SDK 3.0 for Ruby: GitHub Gitee

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
LimitExceeded.CACertNotSupport	不支持私有证书操作。
ResourceNotFound.ProductNotExist	产品不存在。

更新Topic

最近更新时间：2024-03-12 01:33:53

1. 接口描述

接口请求域名： iotcloud.tencentcloudapi.com 。

本接口（UpdateTopicPolicy）用于更新Topic信息

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer 点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：UpdateTopicPolicy。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	是	String	公共参数，详见产品支持的 地域列表。
ProductId	是	String	产品ID 示例值：ABCDE12345
TopicName	是	String	更新前Topic名 示例值：abc
NewTopicName	是	String	更新后Topic名 示例值：ABC
Privilege	是	Integer	Topic权限 示例值：2
BrokerSubscribe	否	BrokerSubscribe	代理订阅信息

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 更新Topic示例

输入示例

```
https://iotcloud.tencentcloudapi.com/?Action=UpdateTopicPolicy
&ProductId=ABCDE12345
&TopicName=abc
&NewTopicName=ABC
&Privilege=2
&BrokerSubscribe.ProductId=11LAWZ3J2D
&BrokerSubscribe.DeviceName=device1
&<公共请求参数>
```

输出示例

```
{
"Response": {
"RequestId": "be69a7a3-7315-40a7-9532-3316e4a3e97e"
}
}
```

5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python: GitHub Gitee
- Tencent Cloud SDK 3.0 for Java: GitHub Gitee
- Tencent Cloud SDK 3.0 for PHP: GitHub Gitee
- Tencent Cloud SDK 3.0 for Go: GitHub Gitee
- Tencent Cloud SDK 3.0 for Node.js: GitHub Gitee
- Tencent Cloud SDK 3.0 for .NET: GitHub Gitee
- Tencent Cloud SDK 3.0 for C++: GitHub Gitee
- Tencent Cloud SDK 3.0 for Ruby: GitHub Gitee

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
InternalError	内部错误。

错误码	描述
InvalidParameterValue	参数取值错误。
InvalidParameterValue.TopicPolicyAlreadyExist	Topic已存在。
ResourceNotFound.TopicPolicyNotExist	Topic不存在。

消息相关接口
发布消息

最近更新时间：2024-03-12 01:33:49

1. 接口描述

接口请求域名： iotcloud.tencentcloudapi.com 。

本接口（PublishMessage）用于向某个主题发消息。

默认接口请求频率限制：300次/秒。

推荐使用 API Explorer　点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：PublishMessage。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	是	String	公共参数，详见产品支持的 地域列表。
Topic	是	String	消息发往的主题。命名规则：${ProductId}/${DeviceName}/[a-zA-Z0-9:_-]{1,128} 示例值：RL0BAZKZ6V/dev1/control
Payload	是	String	消息内容 示例值：hahaha
ProductId	是	String	产品ID 示例值：RL0BAZKZ6V
DeviceName	是	String	设备名称 示例值：dev1
Qos	否	Integer	服务质量等级，取值为0或1 示例值：0
PayloadEncoding	否	String	Payload内容的编码格式，取值为base64或空。base64表示云端将收到的请求数据进行base64解码后下发到设备，空则直接将原始内容下发到设备 示例值：base64

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 发布消息

输入示例

```
https://iotcloud.tencentcloudapi.com/?Action=PublishMessage
&Topic=RL0BAZKZ6V/dev1/control
&Payload=hahaha
&ProductId=RL0BAZKZ6V
&DeviceName=dev1
&Qos=0
&<公共请求参数>
```

输出示例

```
{
"Response": {
"RequestId": "be69a7a3-7315-40a7-9532-3316e4a3e97e"
}
}
```

5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python: GitHub Gitee
- Tencent Cloud SDK 3.0 for Java: GitHub Gitee
- Tencent Cloud SDK 3.0 for PHP: GitHub Gitee
- Tencent Cloud SDK 3.0 for Go: GitHub Gitee
- Tencent Cloud SDK 3.0 for Node.js: GitHub Gitee
- Tencent Cloud SDK 3.0 for .NET: GitHub Gitee
- Tencent Cloud SDK 3.0 for C++: GitHub Gitee
- Tencent Cloud SDK 3.0 for Ruby: GitHub Gitee

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
FailedOperation	操作失败。
FailedOperation.DeviceAlreadyDisabled	设备已经被禁用。
FailedOperation.DeviceNoSubscription	设备没有订阅相应的topic。
FailedOperation.DeviceOffline	设备离线。
FailedOperation.InvalidMsgLen	消息长度非法。
FailedOperation.InvalidTopicName	消息topic非法。
InternalError	内部错误。
InvalidParameterValue	参数取值错误。
InvalidParameterValue.PayloadOverLimit	消息Payload超出限制。
LimitExceeded.MessageSaved	消息已经保存到离线队列。
LimitExceeded.OfflineMessageExceedLimit	qos为1的离线消息超过数量限制。
ResourceNotFound.DeviceNotExist	设备不存在。
ResourceNotFound.ProductOrDeviceNotExist	用户不存在此产品或设备。
UnauthorizedOperation.DeviceIsNotEnabled	设备未启用。

发布RRPC消息

最近更新时间：2024-03-12 01:33:49

1. 接口描述

接口请求域名： iotcloud.tencentcloudapi.com 。

发布RRPC消息

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer 点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：PublishRRPCMessage。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	是	String	公共参数，详见产品支持的 地域列表。
ProductId	是	String	产品ID 示例值：ASBHKN121
DeviceName	是	String	设备名称 示例值：dev
Payload	是	String	消息内容，utf8编码 示例值：1234561

3. 输出参数

参数名称	类型	描述
MessageId	Integer	RRPC消息ID 示例值：74
PayloadBase64	String	设备回复的消息内容，采用base64编码 示例值：QUJDRA==
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 发布RRPC消息

输入示例

```
https://iotcloud.tencentcloudapi.com/?Action=PublishRRPCMessage
&ProductId=ASBHKN121
&DeviceName=dev
&Payload=1234561
&<公共请求参数>
```

输出示例

```
{
"Response": {
"MessageId": 74,
"PayloadBase64": "QUJDRA==",
"RequestId": "be69a7a3-7315-40a7-9532-3316e4a3e97e"
}
}
```

5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python: GitHub Gitee
- Tencent Cloud SDK 3.0 for Java: GitHub Gitee
- Tencent Cloud SDK 3.0 for PHP: GitHub Gitee
- Tencent Cloud SDK 3.0 for Go: GitHub Gitee
- Tencent Cloud SDK 3.0 for Node.js: GitHub Gitee
- Tencent Cloud SDK 3.0 for .NET: GitHub Gitee
- Tencent Cloud SDK 3.0 for C++: GitHub Gitee
- Tencent Cloud SDK 3.0 for Ruby: GitHub Gitee

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
FailedOperation.DeviceAlreadyDisabled	设备已经被禁用。
FailedOperation.DeviceNoSubscription	设备没有订阅相应的topic。

错误码	描述
FailedOperation.DeviceOffline	设备离线。
FailedOperation.RRPCTimeout	RRPC接口未收到设备端响应。
InvalidParameterValue.PayloadOverLimit	消息Payload超出限制。
LimitExceeded.OfflineMessageExceedLimit	qos为1的离线消息超过数量限制。
ResourceNotFound.DeviceNotExist	设备不存在。

规则引擎相关接口
创建规则

最近更新时间：2024-03-12 01:33:48

1. 接口描述

接口请求域名： iotcloud.tencentcloudapi.com 。

本接口（CreateTopicRule）用于创建一个规则

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer

点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：CreateTopicRule。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	是	String	公共参数，详见产品支持的 地域列表。
RuleName	是	String	规则名称 示例值：testrulename
TopicRulePayload	是	TopicRulePayload	规则内容

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 创建规则示例

输入示例

```
POST / HTTP/1.1
Host: iotcloud.tencentcloudapi.com
Content-Type: application/json
X-TC-Action: CreateTopicRule
<公共请求参数>
```

```
{
"TopicRulePayload": {
"Sql": "U0VMRUNUIGZpZWxkMSwgZmllbGQyIEZST00gJ3NyY1Byb2R1Y3RJZC9zcmNEZXZpY2VOYW1lL2V2ZW50Jw==",
"Description": "xx",
"Actions": "[]",
"RuleDisabled": true
},
"RuleName": "testrulename"
}
```


输出示例


```
{
"Response": {
"RequestId": "be69a7a3-7315-40a7-9532-3316e4a3e97e"
}
}
```


5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python: GitHub Gitee
- Tencent Cloud SDK 3.0 for Java: GitHub Gitee
- Tencent Cloud SDK 3.0 for PHP: GitHub Gitee
- Tencent Cloud SDK 3.0 for Go: GitHub Gitee
- Tencent Cloud SDK 3.0 for Node.js: GitHub Gitee
- Tencent Cloud SDK 3.0 for .NET: GitHub Gitee
- Tencent Cloud SDK 3.0 for C++: GitHub Gitee
- Tencent Cloud SDK 3.0 for Ruby: GitHub Gitee

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述

错误码	描述
InternalError	内部错误。
InvalidParameterValue	参数取值错误。
InvalidParameterValue.ActionNil	规则行为未配置。
InvalidParameterValue.CheckForwardURLFail	检查第三方URL超时或失败。
InvalidParameterValue.CloudComponentAlreadyExist	保存失败，行为操作和转发错误行为数据目标不可一致。
InvalidParameterValue.FailActionHasSameDevice	存失败，行为操作和转发错误行为数据目标不可为同一设备。
InvalidParameterValue.InvalidSQL	SQL语句含有非法字符。
InvalidParameterValue.RuleNumberBeyondLimit	规则数量超过限制。
InvalidParameterValue.TopicRuleAlreadyExist	规则已存在。
UnauthorizedOperation.PermissionDenied	没有权限。

删除规则

最近更新时间：2024-03-12 01:33:48

1. 接口描述

接口请求域名： iotcloud.tencentcloudapi.com 。

本接口（DeleteTopicRule）用于删除规则

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer

点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：DeleteTopicRule。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	是	String	公共参数，详见产品支持的 地域列表。
RuleName	是	String	规则名 示例值：test

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 删除规则示例

输入示例

```
https://iotcloud.tencentcloudapi.com/?Action=DeleteTopicRule
&RuleName=test
&<公共请求参数>
```

输出示例

```
{
"Response": {
```

```
"RequestId": "be69a7a3-7315-40a7-9532-3316e4a3e97e"
}
}
```

5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python: GitHub Gitee
- Tencent Cloud SDK 3.0 for Java: GitHub Gitee
- Tencent Cloud SDK 3.0 for PHP: GitHub Gitee
- Tencent Cloud SDK 3.0 for Go: GitHub Gitee
- Tencent Cloud SDK 3.0 for Node.js: GitHub Gitee
- Tencent Cloud SDK 3.0 for .NET: GitHub Gitee
- Tencent Cloud SDK 3.0 for C++: GitHub Gitee
- Tencent Cloud SDK 3.0 for Ruby: GitHub Gitee

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
InternalError	内部错误。
InvalidParameterValue	参数取值错误。
ResourceNotFound.TopicRuleNotExist	规则不存在。

禁用规则

最近更新时间：2024-03-12 01:33:48

1. 接口描述

接口请求域名： iotcloud.tencentcloudapi.com 。

本接口（DisableTopicRule）用于禁用规则

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer 点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：DisableTopicRule。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	是	String	公共参数，详见产品支持的 地域列表。
RuleName	是	String	规则名称 示例值：test

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 禁用接口示例

输入示例

```
https://iotcloud.tencentcloudapi.com/?Action=DisableTopicRule
&RuleName=test
&<公共请求参数>
```

输出示例

```
{
"Response": {
```

```
"RequestId": "be69a7a3-7315-40a7-9532-3316e4a3e97e"
}
}
```

5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python: GitHub Gitee
- Tencent Cloud SDK 3.0 for Java: GitHub Gitee
- Tencent Cloud SDK 3.0 for PHP: GitHub Gitee
- Tencent Cloud SDK 3.0 for Go: GitHub Gitee
- Tencent Cloud SDK 3.0 for Node.js: GitHub Gitee
- Tencent Cloud SDK 3.0 for .NET: GitHub Gitee
- Tencent Cloud SDK 3.0 for C++: GitHub Gitee
- Tencent Cloud SDK 3.0 for Ruby: GitHub Gitee

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
FailedOperation.RuleAlreadyDisabled	该规则引擎已经是禁用状态，不需要再被禁用。
InternalError	内部错误。
InvalidParameterValue	参数取值错误。
InvalidParameterValue.ActionNil	规则行为未配置。
InvalidParameterValue.RuleNumberBeyondLimit	规则数量超过限制。
InvalidParameterValue.TopicRuleSqlNotEdited	规则sql未编辑。
ResourceNotFound.TopicRuleNotExist	规则不存在。

启用规则

最近更新时间：2024-03-12 01:33:48

1. 接口描述

接口请求域名： iotcloud.tencentcloudapi.com 。

本接口（EnableTopicRule）用于启用规则

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer　点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：EnableTopicRule。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	是	String	公共参数，详见产品支持的 地域列表。
RuleName	是	String	规则名称 示例值：test

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 启用规则示例

输入示例

```
https://iotcloud.tencentcloudapi.com/?Action=EnableTopicRule
&RuleName=test
&<公共请求参数>
```

输出示例

```
{
"Response": {
```

```
"RequestId": "be69a7a3-7315-40a7-9532-3316e4a3e97e"
}
}
```


5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python: GitHub Gitee
- Tencent Cloud SDK 3.0 for Java: GitHub Gitee
- Tencent Cloud SDK 3.0 for PHP: GitHub Gitee
- Tencent Cloud SDK 3.0 for Go: GitHub Gitee
- Tencent Cloud SDK 3.0 for Node.js: GitHub Gitee
- Tencent Cloud SDK 3.0 for .NET: GitHub Gitee
- Tencent Cloud SDK 3.0 for C++: GitHub Gitee
- Tencent Cloud SDK 3.0 for Ruby: GitHub Gitee

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
FailedOperation.DuplicationOfFunctionItem	不能创建重复的函数。
FailedOperation.FunctionFileNotExist	函数文件不存在。
FailedOperation.ProxyIPIsNotEnough	代理ip或端口资源不足。
FailedOperation.RuleAlreadyEnabled	规则已经是启用状态。
InternalError	内部错误。
InvalidParameterValue	参数取值错误。
InvalidParameterValue.ActionNil	规则行为未配置。
InvalidParameterValue.CheckForwardURLFail	检查第三方URL超时或失败。
InvalidParameterValue.CloudComponentAlreadyExist	保存失败，行为操作和转发错误行为数据目标不可一致。
InvalidParameterValue.FailActionHasSameDevice	存失败，行为操作和转发错误行为数据目标不可为同一设备。

错误码	描述
InvalidParameterValue.ForwardRedirectDenied	不允许转发重定向。
InvalidParameterValue.InvalidSQL	SQL语句含有非法字符。
InvalidParameterValue.RuleNumberBeyondLimit	规则数量超过限制。
InvalidParameterValue.TopicRuleSqlNotEdited	规则sql未编辑。
OperationDenied.GetTDMQProInternalEndpointFail	TDMQ缺少内网接入点，无法转发，请咨询TDMQ产品
ResourceNotFound.TopicRuleNotExist	规则不存在。

替换规则

最近更新时间：2024-03-12 01:33:48

1. 接口描述

接口请求域名： iotcloud.tencentcloudapi.com 。

本接口（ReplaceTopicRule）用于修改替换规则

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer　点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：ReplaceTopicRule。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	是	String	公共参数，详见产品支持的 地域列表。
RuleName	是	String	规则名称 示例值：testrulename
TopicRulePayload	是	TopicRulePayload	替换的规则包体

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 替换规则示例

输入示例

```
POST / HTTP/1.1
Host: iotcloud.tencentcloudapi.com
Content-Type: application/json
X-TC-Action: ReplaceTopicRule
<公共请求参数>

{
```

```
"TopicRulePayload": {
"Sql": "U0VMRUNUIGZpZWxkMSwgZmllbGQyIEZST00gJ3NyY1Byb2R1Y3RJZC9zcmNEZXZpY2VOYW1lL2V2ZW50Jw==",
"Description": "xx",
"Actions": "xx",
"RuleDisabled": true
},
"RuleName": "testrulename"
}
```


输出示例


```
{
"Response": {
"RequestId": "be69a7a3-7315-40a7-9532-3316e4a3e97e"
}
}
```


5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python: GitHub Gitee
- Tencent Cloud SDK 3.0 for Java: GitHub Gitee
- Tencent Cloud SDK 3.0 for PHP: GitHub Gitee
- Tencent Cloud SDK 3.0 for Go: GitHub Gitee
- Tencent Cloud SDK 3.0 for Node.js: GitHub Gitee
- Tencent Cloud SDK 3.0 for .NET: GitHub Gitee
- Tencent Cloud SDK 3.0 for C++: GitHub Gitee
- Tencent Cloud SDK 3.0 for Ruby: GitHub Gitee

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
InternalError	内部错误。

错误码	描述
InvalidParameterValue	参数取值错误。
InvalidParameterValue.ActionNil	规则行为未配置。
InvalidParameterValue.CheckForwardURLFail	检查第三方URL超时或失败。
InvalidParameterValue.CloudComponentAlreadyExist	保存失败，行为操作和转发错误行为数据目标不可一致。
InvalidParameterValue.FailActionHasSameDevice	存失败，行为操作和转发错误行为数据目标不可为同一设备。
InvalidParameterValue.ForwardRedirectDenied	不允许转发重定向。
InvalidParameterValue.InvalidSQL	SQL语句含有非法字符。
InvalidParameterValue.OperationDenied	修改规则的操作被禁止。
InvalidParameterValue.RepublishTopicFormatError	转发的topic格式错误。
InvalidParameterValue.RuleNumberBeyondLimit	规则数量超过限制。
InvalidParameterValue.TopicRuleAlreadyExist	规则已存在。
InvalidParameterValue.TopicRuleSqlNotEdited	规则sql未编辑。
InvalidParameterValue.UpdateTopicRuleDBFail	请确认规则相关数据是否有更新。
ResourceNotFound.TopicRuleNotExist	规则不存在。
UnauthorizedOperation.PermissionDenied	没有权限。

数据结构

最近更新时间：2023-08-17 03:26:36

Attribute

设备属性

被如下接口引用：CreateDevice。

名称	类型	必选	描述
Tags	Array of DeviceTag	否	属性列表

BindDeviceInfo

子设备信息

被如下接口引用：DescribeGatewayBindDevices。

名称	类型	描述
ProductId	String	产品ID 示例值：ABCDE12345
DeviceName	String	设备名 示例值：ABCD
Tags	Array of DeviceTag	设备Tag
BindTime	Integer	子设备绑定时间 注意：此字段可能返回 null，表示取不到有效值。 示例值：1617801604

BindProductInfo

子产品信息

被如下接口引用：DescribePrivateCABindedProducts。

名称	类型	描述
ProductId	String	产品ID 示例值：ABCDE12345
ProductName	String	产品名 示例值：ABCD

BrokerSubscribe

代理订阅信息

被如下接口引用：CreateTopicPolicy, UpdateTopicPolicy。

名称	类型	必选	描述
ProductId	String	是	产品ID 示例值：ABCDE12345

名称	类型	必选	描述
DeviceName	String	是	设备名 示例值：ABCD

CLSLogItem

CLS日志

被如下接口引用：ListLog。

名称	类型	描述
Content	String	日志内容 示例值：message
DeviceName	String	设备名称 示例值：ABCD
ProductId	String	产品ID 示例值：ABCDE12345
RequestId	String	请求ID 示例值：xxxxxxxxx
Result	String	结果 示例值：SUCC
Scene	String	模块 示例值：SHADOW
Time	String	日志时间 示例值：2021-04-08 00:00:00
UserId	String	腾讯云账号 示例值：12345

CertInfo

X509证书信息

被如下接口引用：DescribePrivateCA, DescribePrivateCAs, DescribeProductCA。

名称	类型	必选	描述
CertName	String	是	证书名称 示例值：IotHub
CertSN	String	是	证书的序列号，16进制编码 示例值：5ff69e4c8afce5d6de8d395b34672944f5b4765a
IssuerName	String	是	证书颁发着名称 示例值：CN=AAA,O=AAA,L=shenzhen,ST=guangdong,C=CN
Subject	String	是	证书主题 示例值：CN=AAA,O=AAA,L=shenzhen,ST=guangdong,C=CN
CreateTime	Integer	是	证书创建时间，秒级时间戳 示例值：1622619674

名称	类型	必选	描述
EffectiveTime	Integer	是	证书生效时间，秒级时间戳 示例值：1622448592
ExpireTime	Integer	是	证书失效时间，秒级时间戳 示例值：1653984592
CertText	String	是	X509证书内容 示例值：-----BEGIN CERTIFICATE-----\nBz4FQdeV1+Xyf+Eg==\n-----END CERTIF

DeviceInfo

设备详细信息

被如下接口引用：DescribeDevices。

名称	类型	描述
DeviceName	String	设备名 示例值：ABCD
Online	Integer	设备是否在线，0不在线，1在线 示例值：1
LoginTime	Integer	设备登录时间 示例值：1617801604
Version	String	设备版本 示例值：1.0.0
DeviceCert	String	设备证书，证书加密的设备返回 示例值：-----BEGIN CERTIFICATE----- MIIFGjCCBAKgAwIBAgIQCgRw0Ja8ihLIkKbf
DevicePsk	String	设备密钥，密钥加密的设备返回 示例值：PSK
Tags	Array of DeviceTag	设备属性
DeviceType	Integer	设备类型 示例值：5
Imei	String	国际移动设备识别码 IMEI 示例值：imei
Isp	Integer	运营商类型 示例值：isp
ConnIP	Integer	IP地址 示例值：123124121
LastUpdateTime	Integer	设备最后更新时间 示例值：1617801604
LoraDevEui	String	LoRa设备的dev eui 示例值：eui
LoraMoteType	Integer	LoRa设备的Mote type 示例值：2

名称	类型	描述
FirstOnlineTime	Integer	首次上线时间 注意：此字段可能返回 null，表示取不到有效值。 示例值：1617801604
LastOfflineTime	Integer	最近下线时间 注意：此字段可能返回 null，表示取不到有效值。 示例值：1617801604
CreateTime	Integer	设备创建时间 注意：此字段可能返回 null，表示取不到有效值。 示例值：1617801604
LogLevel	Integer	设备日志级别 注意：此字段可能返回 null，表示取不到有效值。 示例值：1
CertState	Integer	设备证书获取状态, 1 已获取过设备密钥，0 未获取过设备密钥 注意：此字段可能返回 null，表示取不到有效值。 示例值：1
EnableState	Integer	设备可用状态，0禁用，1启用 注意：此字段可能返回 null，表示取不到有效值。 示例值：1
Labels	Array of DeviceLabel	设备标签 注意：此字段可能返回 null，表示取不到有效值。
ClientIP	String	MQTT客户端IP地址 注意：此字段可能返回 null，表示取不到有效值。 示例值：127.0.0.1
FirmwareUpdateTime	Integer	ota最后更新时间 注意：此字段可能返回 null，表示取不到有效值。 示例值：1617801604
CreateUserId	Integer	创建者 Uin 注意：此字段可能返回 null，表示取不到有效值。 示例值：0
NBIoTDeviceID	String	NB IOT运营商处的DeviceID 示例值：12345

DeviceLabel

设备标签

被如下接口引用：DescribeDevice, DescribeDevices。

名称	类型	必选	描述
Key	String	是	标签标识 示例值：key
Value	String	是	标签值 示例值：value

DeviceResourceInfo

设备资源详细信息

被如下接口引用：DescribeDeviceResource, DescribeDeviceResources。

名称	类型	描述
ProductID	String	产品ID 示例值：AAAAAAAAAA
ProductName	String	产品名 示例值：test
Name	String	资源名称 示例值：test
Md5	String	资源文件md5 示例值：bda2cc64487b0db7f53d689bccabde2d
Size	Integer	资源文件大小 示例值：10
UpdateTime	String	资源更新时间 示例值：2021-01-03 17:04:05
DeviceName	String	设备名称 示例值：test
Status	Integer	设备资源上传状态 示例值：0
Percent	Integer	设备资源上传百分比 示例值：70

DeviceTag

设备属性

被如下接口引用：CreateDevice, DescribeDevice, DescribeDevices, DescribeGatewayBindDevices。

名称	类型	必选	描述
Tag	String	是	属性名称 示例值：tagkey
Type	Integer	是	属性值的类型，1 int，2 string 示例值：1
Value	String	是	属性的值 示例值：123
Name	String	否	属性描述名称 注意：此字段可能返回 null，表示取不到有效值。 示例值：xyz

DeviceUpdateStatus

设备固件更新状态

被如下接口引用：DescribeFirmwareTaskDevices。

名称	类型	描述

名称	类型	描述
DeviceName	String	设备名 示例值：ABCD
LastProcessTime	Integer	最后处理时间 示例值：1617801604
Status	Integer	状态 示例值：1
ErrMsg	String	错误消息 示例值：fail
Retcode	Integer	返回码 示例值：-1
DstVersion	String	目标更新版本 示例值：1.0.1
Percent	Integer	下载中状态时的下载进度 注意：此字段可能返回 null，表示取不到有效值。 示例值：0
OriVersion	String	原版本号 注意：此字段可能返回 null，表示取不到有效值。 示例值：1.0.0
TaskId	Integer	任务ID 注意：此字段可能返回 null，表示取不到有效值。 示例值：123

FirmwareInfo

设备固件详细信息

被如下接口引用：ListFirmwares。

名称	类型	描述
Version	String	固件版本 示例值：1.0.1
Md5sum	String	固件MD5值 示例值：2f8222b4f275c4f18e69c34f66d2631b
CreateTime	Integer	固件创建时间 示例值：1617801604
ProductName	String	产品名称 注意：此字段可能返回 null，表示取不到有效值。 示例值：ABCDE12345
Name	String	固件名称 注意：此字段可能返回 null，表示取不到有效值。 示例值：固件名称
Description	String	固件描述 注意：此字段可能返回 null，表示取不到有效值。 示例值：固件信息描述

名称	类型	描述
ProductId	String	产品ID 注意：此字段可能返回 null，表示取不到有效值。 示例值：ABCDE12345
FwType	String	固件类型 注意：此字段可能返回 null，表示取不到有效值。 示例值：mcu
CreateUserId	Integer	创建者 Uin 注意：此字段可能返回 null，表示取不到有效值。 示例值：0

FirmwareTaskInfo

固件升级任务信息

被如下接口引用：DescribeFirmwareTasks, DescribeResourceTasks。

名称	类型	描述
TaskId	Integer	任务ID 注意：此字段可能返回 null，表示取不到有效值。 示例值：111
Status	Integer	任务状态 注意：此字段可能返回 null，表示取不到有效值。 示例值：1
Type	Integer	任务类型 注意：此字段可能返回 null，表示取不到有效值。 示例值：2
CreateTime	Integer	任务创建时间 注意：此字段可能返回 null，表示取不到有效值。 示例值：1617801604

PayloadLogItem

内容日志项

被如下接口引用：ListLogPayload。

名称	类型	描述
Uin	String	账号id 示例值：12345
ProductId	String	产品id 示例值：ABCDE12345
DeviceName	String	设备名称 示例值：ABCD
SrcType	String	来源类型 示例值：device:DEFAULT_DEVICE
SrcName	String	来源名称 示例值：xxx/xxxx

名称	类型	描述
Topic	String	消息topic 示例值：xxx/xxxx/data
PayloadFormatType	String	内容格式类型 示例值：JSON
Payload	String	内容信息 示例值：{"abc":123}
RequestId	String	请求ID 示例值：xxxxxxxxx
DateTime	String	日期时间 示例值：2021-04-08 00:00:00

ProductInfo

产品详细信息

被如下接口引用：DescribeProducts。

名称	类型	描述
ProductId	String	产品ID 示例值：ABCDE2345
ProductName	String	产品名 示例值：IotHub
ProductMetadata	ProductMetadata	产品元数据
ProductProperties	ProductProperties	产品属性

ProductMetadata

产品元数据

被如下接口引用：DescribeProduct, DescribeProducts。

名称	类型	描述
CreationDate	Integer	产品创建时间 示例值：12345
CreateUserId	Integer	创建者 Uin 示例值：0
UserId	Integer	账号 Uin 示例值：0

ProductProperties

产品属性

被如下接口引用：CreateProduct, DescribeProduct, DescribeProducts。

名称	类型	必选	描述

名称	类型	必选	描述
ProductDescription	String	否	产品描述 示例值：IotHub
EncryptionType	String	否	加密类型，1表示证书认证，2表示签名认证。如不填写，默认值是1 示例值：2
Region	String	否	产品所属区域，目前只支持广州（gz） 示例值：gz
ProductType	Integer	否	产品类型，各个类型值代表的节点-类型如下： 0 普通产品，2 NB-IoT产品，4 LoRa产品，3 LoRa网关产品，5 普通网关产品 默认值是0 示例值：5
Format	String	否	数据格式，取值为json或者custom，默认值是json 示例值：json
Platform	String	否	产品所属平台，默认值是0 示例值：0
ModelId	String	否	产品绑定的物模型ID，-1表示不绑定 示例值：123
ModelName	String	否	产品绑定的物模型名称 示例值：xxx
ProductKey	String	否	产品密钥，suite产品才会有 示例值：xxxxx
RegisterType	Integer	否	动态注册类型 0-关闭, 1-预定义设备名 2-动态定义设备名 示例值：0
ProductSecret	String	否	动态注册产品密钥 示例值：key
RegisterLimit	Integer	否	RegisterType为2时，设备动态创建的限制数量 示例值：1000
OriginProductId	String	否	划归的产品，展示为源产品ID，其余为空 示例值：ABCDE2345
PrivateCAName	String	否	私有CA名称 示例值：caname
OriginUserId	Integer	否	划归的产品，展示为源用户ID，其余为空 示例值：12345
DeviceLimit	Integer	否	设备限制 示例值：1000
ForbiddenStatus	Integer	否	产品禁用状态 示例值：0
AppEUI	String	否	LoRa产品运营侧APPEUI，只有LoRa产品需要填写 示例值：12345

ProductResourceInfo

产品资源详细信息

被如下接口引用：DescribeProductResource, DescribeProductResources。

名称	类型	描述
ProductID	String	产品ID 示例值：AAAAAAAAAA
ProductName	String	产品名 示例值：test
Name	String	资源名称 示例值：test
Md5	String	资源文件md5 示例值：bda2cc64487b0db7f53d689bccabde2d
Size	Integer	资源文件大小 示例值：10
Description	String	资源文件描述 注意：此字段可能返回 null，表示取不到有效值。 示例值：测试
CreateTime	String	资源创建时间 示例值：2021-01-03 17:04:05

ProductTaskInfo

产品级任务详细信息

被如下接口引用：DescribeProductTask, DescribeProductTasks。

名称	类型	描述
Id	Integer	任务ID 示例值：123
Type	Integer	任务类型 0-批量创建设备类型 示例值：0
State	Integer	任务状态 0-创建中 1-待执行 2-执行中 3-执行失败 4-子任务部分失败 5-执行成功 示例值：0
ParametersType	String	任务参数类型 cosfile-文件输入 random-随机生成 示例值：cosfile
Parameters	String	任务参数 示例值：abc
ResultType	String	任务执行结果类型 cosfile-文件输出 errmsg-错误信息 注意：此字段可能返回 null，表示取不到有效值。 示例值：cosfile
Result	String	任务执行结果 注意：此字段可能返回 null，表示取不到有效值。 示例值：succ
BatchCount	Integer	子任务总个数 注意：此字段可能返回 null，表示取不到有效值。 示例值：100
BatchOffset	Integer	子任务已执行个数 注意：此字段可能返回 null，表示取不到有效值。 示例值：50

名称	类型	描述
CreateTime	Integer	任务创建时间 示例值：1617801604
UpdateTime	Integer	任务更新时间 示例值：1617801604
CompleteTime	Integer	任务完成时间 注意：此字段可能返回 null，表示取不到有效值。 示例值：1617801604

ResetDeviceResult

重置设备状态结果

被如下接口引用：ResetDeviceState。

名称	类型	描述
DeviceName	String	设备名 示例值：ABCD
Success	Boolean	是否成功 示例值：true
Reason	String	失败原因 注意：此字段可能返回 null，表示取不到有效值。 示例值：not exist

SDKLogItem

SDK日志项

被如下接口引用：ListSDKLog。

名称	类型	描述
ProductId	String	产品ID 示例值：ABCDE12345
DeviceName	String	设备名称 示例值：ABCD
Level	String	日志等级 示例值：DEBUG
DateTime	String	日志时间 示例值：2021-04-08 00:00:00
Content	String	日志内容 示例值：12345

SearchKeyword

搜索关键词

被如下接口引用：DescribeFirmwareTaskDevices, DescribeFirmwareTasks, DescribeResourceTasks, ListFirmwares。

名称	类型	必选	描述

名称	类型	必选	描述
Key	String	是	搜索条件的Key 示例值：key
Value	String	否	搜索条件的值 示例值：value

StatusStatistic

状态统计信息

被如下接口引用：DescribeFirmwareTaskDistribution。

名称	类型	描述
Status	Integer	任务状态 注意：此字段可能返回 null，表示取不到有效值。 示例值：1
Total	Integer	统计总数 注意：此字段可能返回 null，表示取不到有效值。 示例值：123

TopicRuleInfo

规则详细信息

被如下接口引用：ListTopicRules。

名称	类型	描述
RuleName	String	规则名称 示例值：test
Description	String	规则描述 示例值：test
CreatedAt	Integer	创建时间 示例值：1000
RuleDisabled	Boolean	不生效 示例值：false
TopicPattern	String	规则模式 示例值：test

TopicRulePayload

创建规则请求包体

被如下接口引用：CreateTopicRule, ReplaceTopicRule。

名称	类型	必选	描述
Sql	String	是	规则的SQL语句，如： SELECT * FROM 'pid/dname/event'，然后对其进行base64编码，得：U0VMRUNUICogRlJPTSAncGlkL2RuYW1lL2V2ZW50Jw== 示例值：U0VMRUNUICogRlJPTSAncGlkL2RuYW1lL2V2ZW50Jw==

名称	类型	必选	描述

Actions	String	否	行为的JSON字符串，大部分种类举例如下：
			[
			{
			"republish": {
			"topic": "TEST/test"
			}
			},
			{
			"forward": {
			"api": "http://127.0.0.1:8080",
			"token":"xxx"
			}
			},
			{
			"ckafka": {
			"instance": {
			"id": "ckafka-test",
			"name": ""
			},
			"topic": {
			"id": "topic-test",
			"name": "test"
			},
			"region": "gz"
			}
			},
			{
			"cmqqueue": {
			"queuename": "queue-test-TEST",
			"region": "gz"
			}
			},
			{
			"mysql": {
			"instanceid": "cdb-test",
			"region": "gz",
			"username": "test",
			"userpwd": "*",
			"dbname": "d_mqtt",
			"tablename": "t_test",
			"fieldpairs": [
			{
			"field": "test",
			"value": "test"
			}
			],
			"devicetype": "CUSTOM"
			}
			}
			]
			示例值：[]

Description	String	否	规则描述 示例值：test
RuleDisabled	Boolean	否	是否禁用规则 示例值：false

错误码

最近更新时间：2023-12-26 01:17:18

功能说明

如果返回结果中存在 Error 字段，则表示调用 API 接口失败。例如：

```
{
"Response": {
"Error": {
"Code": "AuthFailure.SignatureFailure",
"Message": "The provided credentials could not be validated. Please check your signature is correct."
},
"RequestId": "ed93f3cb-f35e-473f-b9f3-0d451b8b79c6"
}
}
```

Error 中的 Code 表示错误码，Message 表示该错误的具体信息。

错误码列表

公共错误码

错误码	说明
ActionOffline	接口已下线。
AuthFailure.InvalidAuthorization	请求头部的 Authorization 不符合腾讯云标准。
AuthFailure.InvalidSecretId	密钥非法（不是云 API 密钥类型）。
AuthFailure.MFAFailure	MFA 错误。
AuthFailure.SecretIdNotFound	密钥不存在。请在 控制台 检查密钥是否已被删除或者禁用，如状态正常，请检查密钥是否填写正确，注意前后不得有空格。
AuthFailure.SignatureExpire	签名过期。Timestamp 和服务器时间相差不得超过五分钟，请检查本地时间是否和标准时间同步。
AuthFailure.SignatureFailure	签名错误。签名计算错误，请对照调用方式中的签名方法文档检查签名计算过程。
AuthFailure.TokenFailure	token 错误。
AuthFailure.UnauthorizedOperation	请求未授权。请参考 CAM 文档对鉴权的说明。
DryRunOperation	DryRun 操作，代表请求将会是成功的，只是多传了 DryRun 参数。
FailedOperation	操作失败。
InternalError	内部错误。
InvalidAction	接口不存在。
InvalidParameter	参数错误（包括参数格式、类型等错误）。
InvalidParameterValue	参数取值错误。

错误码	说明
InvalidRequest	请求 body 的 multipart 格式错误。
IpInBlacklist	IP地址在黑名单中。
IpNotInWhitelist	IP地址不在白名单中。
LimitExceeded	超过配额限制。
MissingParameter	缺少参数。
NoSuchProduct	产品不存在
NoSuchVersion	接口版本不存在。
RequestLimitExceeded	请求的次数超过了频率限制。
RequestLimitExceeded.GlobalRegionUinLimitExceeded	主账号超过频率限制。
RequestLimitExceeded.IPLimitExceeded	IP限频。
RequestLimitExceeded.UinLimitExceeded	主账号限频。
RequestSizeLimitExceeded	请求包超过限制大小。
ResourceInUse	资源被占用。
ResourceInsufficient	资源不足。
ResourceNotFound	资源不存在。
ResourceUnavailable	资源不可用。
ResponseSizeLimitExceeded	返回包超过限制大小。
ServiceUnavailable	当前服务暂时不可用。
UnauthorizedOperation	未授权操作。
UnknownParameter	未知参数错误，用户多传未定义的参数会导致错误。
UnsupportedOperation	操作不支持。
UnsupportedProtocol	http(s) 请求协议错误，只支持 GET 和 POST 请求。
UnsupportedRegion	接口不支持所传地域。

业务错误码

错误码	说明
FailedOperation.AccountIsolated	操作失败，账号已欠费隔离。
FailedOperation.AlreadyDistributionDevice	已分发设备，不能再次创建。
FailedOperation.BindDeviceOverLimit	绑定设备超过限制。
FailedOperation.BindDevicePerOnceOverLimit	单次绑定的设备数量超过限制。
FailedOperation.BroadcastTaskIsRunning	广播任务正在执行。
FailedOperation.DeviceAlreadyDisabled	设备已经被禁用。
FailedOperation.DeviceFirmwareTaskAlreadDone	设备固件升级任务已经完成。

错误码	说明
FailedOperation.DeviceIsUpdating	设备正在升级中。
FailedOperation.DeviceNoSubscription	设备没有订阅相应的topic。
FailedOperation.DeviceOffline	设备离线。
FailedOperation.DeviceRunningOtherOtaTask	设备已经运行其他ota升级任务。
FailedOperation.DuplicationOfFunctionItem	不能创建重复的函数。
FailedOperation.FunctionFileNotExist	函数文件不存在。
FailedOperation.InvalidMsgLen	消息长度非法。
FailedOperation.InvalidTopicName	消息topic非法。
FailedOperation.ProductNotBind	产品未绑定，无法代理订阅。
FailedOperation.ProductResourceDuplicate	同名产品资源已存在。
FailedOperation.ProxyIPIsNotEnough	代理ip或端口资源不足。
FailedOperation.RRPCTimeout	RRPC接口未收到设备端响应。
FailedOperation.ResourceFileNotMatch	资源文件MD5或者大小不一致。
FailedOperation.RuleAlreadyDisabled	该规则引擎已经是禁用状态，不需要再被禁用。
FailedOperation.RuleAlreadyEnabled	规则已经是启用状态。
FailedOperation.TidWhiteListNotOpen	白名单校验未开启，用户不可创建设备，平台会根据设备认证时携带的设备名称自动创建设备。
FailedOperation.UpdateVersionNotMatch	更新版本不匹配。
InternalError.DBOperationError	数据库内部错误。
InvalidParameterValue.ActionNil	规则行为未配置。
InvalidParameterValue.CACertInvalid	CA证书内容错误。
InvalidParameterValue.CACertNotMatch	CA验证证书不匹配。
InvalidParameterValue.CheckForwardURLFail	检查第三方URL超时或失败。
InvalidParameterValue.CloudComponentAlreadyExist	保存失败，行为操作和转发错误行为数据目标不可一致。
InvalidParameterValue.DefinedPskNotBase64	格式错误，DefinedPsk需为Base64格式的字符串。
InvalidParameterValue.DeviceAlreadyExist	创建的设备名已存在。
InvalidParameterValue.DeviceIsNotGateway	设备不是网关类型。
InvalidParameterValue.FailActionHasSameDevice	存失败，行为操作和转发错误行为数据目标不可为同一设备。
InvalidParameterValue.FirmwareAlreadyExist	固件已存在。
InvalidParameterValue.ForwardRedirectDenied	不允许转发重定向。
InvalidParameterValue.InvalidJSON	JSON参数非法。
InvalidParameterValue.InvalidSQL	SQL语句含有非法字符。

错误码	说明
InvalidParameterValue.JSONHasInvalidNode	State JSON对象中包含非法节点。
InvalidParameterValue.JSONSizeExceedLimit	State JSON对象超过大小限制，最大为 8k。
InvalidParameterValue.NotMergeAble	不可合并。
InvalidParameterValue.OperationDenied	修改规则的操作被禁止。
InvalidParameterValue.ParamIncomplete	请求中缺少关键字段信息。
InvalidParameterValue.PayloadOverLimit	消息Payload超出限制。
InvalidParameterValue.PrefixInvalid	prefix不合法。
InvalidParameterValue.ProductAlreadyExist	创建的产品名已存在。
InvalidParameterValue.ProductTypeNotSupport	产品类型不支持。
InvalidParameterValue.RepublishTopicFormatError	转发的topic格式错误。
InvalidParameterValue.RuleNumberBeyondLimit	规则数量超过限制。
InvalidParameterValue.TidProductAlreadyExist	该TID产品已存在。
InvalidParameterValue.TopicPolicyAlreadyExist	Topic已存在。
InvalidParameterValue.TopicRuleAlreadyExist	规则已存在。
InvalidParameterValue.TopicRuleSqlNotEdited	规则sql未编辑。
InvalidParameterValue.UpdateTopicRuleDBFail	请确认规则相关数据是否有更新。
LimitExceeded.CAAlreadyBindProduct	CA证书已经绑定了产品，无法操作。
LimitExceeded.CACertLimit	CA证书达到上限。
LimitExceeded.CACertNameRepeat	CA证书名称重复。
LimitExceeded.CACertNotSupport	不支持私有证书操作。
LimitExceeded.CARepeat	CA证书重复。
LimitExceeded.DeviceExceedLimit	设备数量超过限制。
LimitExceeded.FirmwareExceedLimit	固件数量超出限制。
LimitExceeded.MessageSaved	消息已经保存到离线队列。
LimitExceeded.OfflineMessageExceedLimit	qos为1的离线消息超过数量限制。
LimitExceeded.ProductExceedLimit	超过产品数量限制。
LimitExceeded.TopicPolicyExceedLimit	Topic数量超出限制。
OperationDenied.GetTDMQProInternalEndpointFail	TDMQ缺少内网接入点，无法转发，请咨询TDMQ产品
ResourceNotFound.CACertNotExist	CA证书不存在。
ResourceNotFound.CreateMultiDeviceTaskNotExist	批量创建设备任务不存在。
ResourceNotFound.DeviceFirmwareTaskNotExist	设备固件升级任务不存在。
ResourceNotFound.DeviceHasNoFirmware	设备无固件版本。

错误码	说明
ResourceNotFound.DeviceNotExist	设备不存在。
ResourceNotFound.DeviceResourceNotExist	设备资源不存在。
ResourceNotFound.DeviceShadowNotExist	设备影子不存在。
ResourceNotFound.FirmwareNotExist	固件不存在。
ResourceNotFound.FirmwareTaskNotExist	固件升级任务不存在。
ResourceNotFound.ProductNotExist	产品不存在。
ResourceNotFound.ProductOrDeviceNotExist	用户不存在此产品或设备。
ResourceNotFound.ProductResourceNotExist	产品资源不存在。
ResourceNotFound.ResourceFileNotExist	资源文件不存在。
ResourceNotFound.TaskNotExist	任务不存在。
ResourceNotFound.ThingModelNotExist	物模型不存在。
ResourceNotFound.TopicPolicyNotExist	Topic不存在。
ResourceNotFound.TopicRuleNotExist	规则不存在。
UnauthorizedOperation.DeleteTidFail	该产品已存在TID申请，禁止删除。
UnauthorizedOperation.DeviceHasAlreadyBindGateway	该设备绑定了网关设备，无法删除。
UnauthorizedOperation.DeviceIsNotEnabled	设备未启用。
UnauthorizedOperation.DevicesExistUnderProduct	删除的产品下还包括未删除的设备。
UnauthorizedOperation.GatewayHasBindedDevices	该设备下仍有绑定的设备。
UnauthorizedOperation.PermissionDenied	没有权限。
UnauthorizedOperation.ProductCantHaveLoRaDevice	该产品类型不能创建LoRa设备。
UnauthorizedOperation.ProductCantHaveNormalDevice	NB-IoT产品不允许创建普通设备。
UnauthorizedOperation.ProductCantHaveNotLoRaDevice	该产品类型只能创建LoRa设备。
UnauthorizedOperation.ProductIsForbidden	产品禁用了该功能。
UnauthorizedOperation.ProductNotSupportPSK	产品不支持密钥认证。
UnauthorizedOperation.UserNotAuthenticaed	用户未通过实名认证。
UnsupportedOperation.ClientCertAlreadyGot	设备私钥已被获取。
UnsupportedOperation.DeviceOtaTaskInProgress	设备ota升级中。
UnsupportedOperation.GatewayProductHasBindedProduct	网关产品下存在绑定的子产品，无法删除。
UnsupportedOperation.ProductHasBindGateway	存在网关设备绑定当前产品，无法删除。
UnsupportedOperation.ProductHasBindedGatewayProduct	产品存在绑定的网关产品，无法删除。
UnsupportedOperation.SuiteTokenNoCreate	产品为Suite token类型，无法创建新设备。
UnsupportedOperation.WrongProductAuthType	不支持的认证类型。