

多网聚合加速 API 文档



腾讯云

【版权声明】

©2013–2025 腾讯云版权所有

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

【商标声明】



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

【服务声明】

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

【联系我们】

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

文档目录

API 文档

更新历史

简介

API 概览

调用方式

请求结构

公共参数

签名方法 v3

签名方法

返回结果

参数类型

多网聚合加速相关接口

批量激活硬件设备

新建互通规则

分组添加设备

新建设备

分组删除设备

新建分组

添加硬件设备

设置或更新密钥

删除设备

删除互通规则

删除分组

获取分组详细信息

获取设备详细信息

获取互通规则列表

获取分组列表

设备信息列表

根据设备组获取数据流量统计数据

获取数据流量统计数据

根据区域获取数据流量统计数据

获取厂商硬件列表

批量获取设备流量统计

获取流量监控信息

获取公钥

下载用量统计数据

获取厂商硬件设备列表

更新设备

更新互通规则备注

更新分组信息

更新互通规则CIDR

修改互通规则开关

更新硬件信息

订购一次性授权License

订购流量包

查询流量告警信息

修改流量包自动续费标识

设置用户流量告警信息

获取流量包列表

活跃设备数量统计

下载活跃设备数量统计

获取设备付费模式

数据结构

错误码

API 文档

更新历史

最近更新时间：2025-04-08 01:20:58

第 28 次发布

发布时间：2025-04-08 01:20:55

本次发布包含了以下内容：

改善已有的文档。

删除接口：

- CreateQos
- DeleteQos
- DescribeQos

删除数据结构：

- Capacity
- Context
- DestAddressInfo
- DeviceInfo
- ExpectedThreshold
- NetworkData
- SrcAddressInfo

第 27 次发布

发布时间：2025-02-19 01:50:33

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- [DownloadActiveDeviceCount](#)
- [GetActiveDeviceCount](#)

新增数据结构：

- [ActiveDeviceList](#)

第 26 次发布

发布时间：2025-02-11 01:46:01

本次发布包含了以下内容：

改善已有的文档。

修改数据结构：

- [FlowDetails](#)
 - 修改成员：NetDetails, DeviceId, MaxValue, AvgValue, TotalValue

第 25 次发布

发布时间：2025-01-01 01:23:28

本次发布包含了以下内容：

改善已有的文档。

修改数据结构：

- [DeviceNetInfo](#)
 - 修改成员：DownRate, UpRate

第 24 次发布

发布时间：2024-12-30 01:46:00

本次发布包含了以下内容：

改善已有的文档。

预下线接口：

- CreateQos
- DeleteQos
- DescribeQos

第 23 次发布

发布时间：2024-08-23 01:55:12

本次发布包含了以下内容：

改善已有的文档。

修改数据结构：

- [DeviceBaseInfo](#)
 - 新增成员：Sn, Vendor

第 22 次发布

发布时间：2024-08-15 02:00:04

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- [OrderPerLicense](#)

第 21 次发布

发布时间：2024-07-22 01:55:24

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- [AddL3Conn](#)

- [DeleteL3Conn](#)
- [GetL3ConnList](#)
- [UpdateL3Cidr](#)
- [UpdateL3Conn](#)
- [UpdateL3Switch](#)

新增数据结构：

- [L3ConnInfo](#)

第 20 次发布

发布时间：2024-07-11 01:20:22

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- [AddGroup](#)
- [DeleteGroup](#)
- [GetFlowStatisticByRegion](#)
- [GetGroupDetail](#)
- [GetGroupList](#)
- [GroupAddDevice](#)
- [GroupDeleteDevice](#)
- [SetNotifyUrl](#)
- [UpdateGroup](#)

修改接口：

- [AddDevice](#)
 - 新增入参：FlowTrunc
- [UpdateDevice](#)
 - 新增入参：FlowTrunc

新增数据结构：

- [GroupInfo](#)

修改数据结构：

- [ActivateHardware](#)
 - 新增成员：FlowTrunc, DeviceId
- [DeviceBaseInfo](#)
 - 新增成员：FlowTrunc
- [HardwareInfo](#)
 - 新增成员：FlowTrunc

第 19 次发布

发布时间：2024-05-28 01:54:57

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- [GetFlowAlarmInfo](#)
- [GetFlowStatisticByGroup](#)

第 18 次发布

发布时间：2024-04-12 01:19:07

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- [AddDevice](#)
 - 新增入参：LicensePayMode, GroupName, GroupId
- [GetFlowStatistic](#)
 - 新增入参：DeviceList
- [GetStatisticData](#)
 - 新增入参：DeviceList, GroupId
- [OrderFlowPackage](#)
 - 新增入参：FlowTruncFlag

修改数据结构：

- [ActivateHardware](#)
 - 新增成员：LicensePayMode, GroupId, GroupName
- [DeviceBaseInfo](#)
 - 新增成员：LicensePayMode, Payer, GroupId, GroupName
- [DeviceDetails](#)
 - **修改成员：** GatewaySite, BusinessDownRate, BusinessUpRate
- [FlowPackageInfo](#)
 - 新增成员：TruncFlag, CapacityRemainPrecise
- [HardwareInfo](#)
 - 新增成员：LicensePayMode, Payer, GroupId, GroupName
 - **修改成员：** DeviceId, DeviceName, ActiveTime, LastOnlineTime, Description, VendorDescription, LicenseChargingMode, CreateTime, SN
- [VendorHardware](#)
 - 新增成员：LicensePayMode, Payer
 - **修改成员：** HardwareId, SN, CreateTime, Status, ActiveTime, Description, DeviceId, LicenseChargingMode, LastOnlineTime

第 17 次发布

发布时间：2024-03-01 01:20:25

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- [AddDevice](#)
 - 新增入参：AccessScope
- [GetFlowStatistic](#)

- 新增入参：AccessRegion, GatewayType
- [GetMultiFlowStatistic](#)
 - 新增入参：AccessRegion, GatewayType
- [GetNetMonitor](#)
 - 新增入参：GatewayType
 - 新增出参：AccessRegion
- [GetStatisticData](#)
 - 新增入参：AccessRegion, GatewayType

修改数据结构：

- [ActivateHardware](#)
 - 新增成员：AccessScope
- [DeviceBaseInfo](#)
 - 新增成员：AccessScope
- [FlowPackageInfo](#)
 - 新增成员：CreateTime, ModifyStatus

第 16 次发布

发布时间：2024-02-26 01:19:32

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- [OrderFlowPackage](#)
 - 新增入参：AutoVoucher, VoucherIds

第 15 次发布

发布时间：2023-12-01 01:19:47

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- [GetDevicePayMode](#)
- [GetFlowPackages](#)
- [ModifyPackageRenewFlag](#)
- [OrderFlowPackage](#)

新增数据结构：

- [DevicePayModelInfo](#)
- [FlowPackageInfo](#)

第 14 次发布

发布时间：2023-11-28 01:19:46

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- [ActivateHardware](#)
- [AddHardware](#)
- [GetHardwareList](#)
- [GetVendorHardware](#)
- [UpdateHardware](#)

新增数据结构：

- [ActivateHardware](#)
- [Hardware](#)
- [HardwareInfo](#)
- [VendorHardware](#)

第 13 次发布

发布时间：2023-11-08 01:53:44

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- [GetNetMonitor](#)

新增数据结构：

- [MonitorData](#)
- [SlotNetInfo](#)

第 12 次发布

发布时间：2023-08-01 01:32:11

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- [GetDevices](#)
 - 新增入参：DeviceType

第 11 次发布

发布时间：2023-02-24 01:20:12

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- [GetMultiFlowStatistic](#)

新增数据结构：

- [FlowDetails](#)

第 10 次发布

发布时间：2023-02-23 01:43:45

本次发布包含了以下内容：

改善已有的文档。

修改数据结构：

- [DeviceDetails](#)
 - 新增成员：GatewaySite, BusinessDownRate, BusinessUpRate
- [DeviceNetInfo](#)
 - 新增成员：DownRate, UpRate

第 9 次发布

发布时间：2022-11-17 11:16:37

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- [CreateEncryptedKey](#)
- [GetPublicKey](#)

第 8 次发布

发布时间：2022-11-17 06:34:55

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- [AddDevice](#)
 - 新增入参：Encrypted
 - 新增出参：Signature

第 7 次发布

发布时间：2022-09-14 20:21:28

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- [GetFlowStatistic](#)

新增数据结构：

- [NetDetails](#)

第 6 次发布

发布时间：2022-09-14 15:06:36

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- [AddDevice](#)
- [DeleteDevice](#)
- [GetDevice](#)
- [GetDevices](#)
- [GetStatisticData](#)
- [UpdateDevice](#)

新增数据结构：

- [DeviceBaseInfo](#)
- [DeviceDetails](#)
- [DeviceNetInfo](#)
- [UpdateNetInfo](#)

第 5 次发布

发布时间：2022-02-21 08:09:33

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- [CreateQos](#)
 - 新增入参：Context, Extern

新增数据结构：

- [Context](#)
- [ExpectedThreshold](#)
- [NetworkData](#)

第 4 次发布

发布时间：2021-09-29 08:07:33

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- [DescribeQos](#)

第 3 次发布

发布时间：2021-09-27 08:14:04

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- CreateQos
 - 新增入参：Protocol

第 2 次发布

发布时间：2021-07-22 08:00:32

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- CreateQos
 - 新增入参：Templateld

第 1 次发布

发布时间：2021-05-17 08:00:05

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- CreateQos
- DeleteQos

新增数据结构：

- Capacity
- DestAddressInfo
- DeviceInfo
- SrcAddressInfo

简介

最近更新时间：2025-04-25 01:45:05

多网聚合加速（Multiple Network Acceleration）应用多通道网络聚合技术，可将多个网络线路智能聚合，将它们视为一个大的网络通道，提供更加稳定、快速、带宽更大的网络服务。产品支持5G/4G/WAN/Wi-Fi/卫星等多类型网络，为移动端手机和行业端各类硬件提供服务。

API 概览

最近更新时间：2025-04-08 01:20:58

多网聚合加速相关接口

接口名称	接口功能	频率限制（次/秒）
ActivateHardware	批量激活硬件设备	20
AddDevice	新建设备	20
AddGroup	新建分组	20
AddL3Conn	新建互通规则	20
GroupAddDevice	分组添加设备	20
GroupDeleteDevice	分组删除设备	20
AddHardware	添加硬件设备	20
CreateEncryptedKey	设置或更新密钥	20
DeleteDevice	删除设备	20
DeleteGroup	删除分组	20
DeleteL3Conn	删除互通规则	20
GetDevice	获取设备详细信息	20
GetGroupDetail	获取分组详细信息	20
GetDevices	设备信息列表	20
GetGroupList	获取分组列表	20
GetL3ConnList	获取互通规则列表	20
GetFlowStatistic	获取数据流量统计数据	20
GetFlowStatisticByGroup	根据设备组获取数据流量统计数据	20
GetFlowStatisticByRegion	根据区域获取数据流量统计数据	20
GetHardwareList	获取厂商硬件列表	20
GetMultiFlowStatistic	批量获取设备流量统计	20
GetNetMonitor	获取流量监控信息	20
GetPublicKey	获取公钥	20
GetStatisticData	下载用量统计数据	20
GetVendorHardware	获取厂商硬件设备列表	20
UpdateDevice	更新设备	20
UpdateGroup	更新分组信息	20
UpdateL3Cidr	更新互通规则CIDR	20

接口名称	接口功能	频率限制（次/秒）
UpdateL3Conn	更新互通规则备注	20
UpdateL3Switch	修改互通规则开关	20
UpdateHardware	更新硬件信息	20
OrderFlowPackage	订购流量包	20
OrderPerLicense	订购一次性授权License	20
GetFlowAlarmInfo	查询流量告警信息	20
ModifyPackageRenewFlag	修改流量包自动续费标识	20
SetNotifyUrl	设置用户流量告警信息	20
GetFlowPackages	获取流量包列表	20
DownloadActiveDeviceCount	下载活跃设备数量统计	20
GetActiveDeviceCount	活跃设备数量统计	20
GetDevicePayMode	获取设备付费模式	20

 注意：

以上给出的接口频率限制维度为 API + 接入地域 + 子账号，有关限频更多说明参考：[API 频率限制说明](#)

调用方式

请求结构

最近更新时间：2025-05-21 01:24:55

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 `mna.tencentcloudapi.com`，也支持指定地域域名访问，例如广州地域的域名为 `mna.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 `mna.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）	<code>mna.tencentcloudapi.com</code>
华南地区（广州）	<code>mna.ap-guangzhou.tencentcloudapi.com</code>
华东地区（上海）	<code>mna.ap-shanghai.tencentcloudapi.com</code>
华东地区（南京）	<code>mna.ap-nanjing.tencentcloudapi.com</code>
华北地区（北京）	<code>mna.ap-beijing.tencentcloudapi.com</code>
西南地区（成都）	<code>mna.ap-chengdu.tencentcloudapi.com</code>
西南地区（重庆）	<code>mna.ap-chongqing.tencentcloudapi.com</code>
港澳台地区（中国香港）	<code>mna.ap-hongkong.tencentcloudapi.com</code>
亚太东南（新加坡）	<code>mna.ap-singapore.tencentcloudapi.com</code>
亚太东南（雅加达）	<code>mna.ap-jakarta.tencentcloudapi.com</code>
亚太东南（曼谷）	<code>mna.ap-bangkok.tencentcloudapi.com</code>
亚太东北（首尔）	<code>mna.ap-seoul.tencentcloudapi.com</code>
亚太东北（东京）	<code>mna.ap-tokyo.tencentcloudapi.com</code>
美国东部（弗吉尼亚）	<code>mna.na-ashburn.tencentcloudapi.com</code>
美国西部（硅谷）	<code>mna.na-siliconvalley.tencentcloudapi.com</code>
南美地区（圣保罗）	<code>mna.sa-saopaulo.tencentcloudapi.com</code>
欧洲地区（法兰克福）	<code>mna.eu-frankfurt.tencentcloudapi.com</code>

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）	mna.ap-shanghai-fsi.tencentcloudapi.com
华南地区（深圳金融）	mna.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：2024-11-15 01:49:08

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=AKID***/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，AKID*** 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 mna；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*****/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*****/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*****/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输入公共参数 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=37ac2f4fde00b0ac9bd9eadeb459b1bbe224158d66e7ae5fcadb70b2d181d02&Region=ap-guangzhou&Nonce=23823223&SecretId=AKID*****
```

```
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=37ac2f4fde00b0ac9bd9eadeb459b1bbe224158d66e7ae5fcadb70b2d181d02&Region=ap-guangzhou&Nonce=23823223&SecretId=AKID*****
```

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
华东地区（上海）	ap-shanghai

签名方法 v3

最近更新时间：2024-12-25 01:47:46

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：AKID***** 和 *****。用户想查看广州云服务器名为“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=AKID*****/2019-02-25/cvm/tc3_request, SignedHeaders=content-type;host;x-tc-action, Signature=10b1a37a7301a02ca19a647ad722d5e43b4b3cff309d421d85b46093f6ab6c4f" \
-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。
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；

字段名称	解释
	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 <code>content-type:application/json; charset=utf-8\nhost:cvm.tencentcloudapi.com\nx-tc-action:describeinstances\n</code>。 注意：<code>content-type</code> 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 <code>charset</code> 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。<code>content-type</code> 和 <code>host</code> 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 <code>content-type;host;x-tc-action</code>
HashedRequestPayload	请求正文（payload，即 body，此示例为 <code>{"Limit": 1, "Filters": [{"Values": [{"\u672a\u547d\u540d"}], "Name": "instance-name"}]}</code>）的哈希值，计算伪代码为 <code>Lowercase(HexEncode(Hash.SHA256(RequestPayload)))</code>，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 <code>35e9c5b0e3ae67532d3c9f17ead6c90222632e5b1ff7f6e89887f1398934f064</code>。

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 <code>TC3-HMAC-SHA256</code> 。
RequestTimestamp	请求时间戳，即请求头部的公共参数 <code>X-TC-Timestamp</code> 取值，取当前时间 UNIX 时间戳，精确到秒。此示例取值为 <code>1551113065</code> 。
CredentialScope	凭证范围，格式为 <code>Date/service/tc3_request</code> ，包含日期、所请求的服务和终止字符串（ <code>tc3_request</code> ）。 <code>Date</code> 为 UTC 标准时间的日期，取值需要和公共参数 <code>X-TC-Timestamp</code> 换算的

字段名称	解释
	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 = "*****"
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，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：da98fb70dcf6b112dc21038d1eeeb3a95c74b4dcb12c1131f864f6066bd02be0，8d70cbefb03939f929db64d32dc2ba89b1095620119fe3e050e2b18c5bd2752f，b596b923aad85185e2d1f6659d2a062e0a86731226e021e61bfe06f7ed05f5af。

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 *****。
Date	即 Credential 中的 Date 字段信息。此示例取值为 2019-02-25。
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 10b1a37a7301a02ca19a647ad722d5e43b4b3cff309d421d85b46093f6ab6c4f。

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 AKID*****。
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 10b1a37a7301a02ca19a647ad722d5e43b4b3cff309d421d85b46093f6ab6c4f。

根据以上规则，示例中得到的值为：

```
TC3-HMAC-SHA256 Credential=AKID*****/2019-02-25/cvm/tc3_request, SignedHeaders=c  
ontent-type;host;x-tc-action, Signature=10b1a37a7301a02ca19a647ad722d5e43b4b3cff309d421d85b46093f6ab6c4f
```

最终完整的调用信息如下：

```
POST https://cvm.tencentcloudapi.com/  
Authorization: TC3-HMAC-SHA256 Credential=AKID*****/2019-02-25/cvm/tc3_request,  
SignedHeaders=content-type;host;x-tc-action, Signature=10b1a37a7301a02ca19a647ad722d5e43b4b3cff309d421d85b4  
6093f6ab6c4f  
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 AKID*****
    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 *****
    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 AKID*****
secret_id = os.environ.get("TENCENTCLOUD_SECRET_ID")
# 需要设置环境变量 TENCENTCLOUD_SECRET_KEY, 值为示例的 *****
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.utcnow().timestamp(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 AKID*****  
    secretId := os.Getenv("TENCENTCLOUD_SECRET_ID")  
    // 需要设置环境变量 TENCENTCLOUD_SECRET_KEY, 值为示例的 *****  
    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 AKID*****
$secretId = getenv("TENCENTCLOUD_SECRET_ID");
// 需要设置环境变量 TENCENTCLOUD_SECRET_KEY, 值为示例的 *****
$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"
.$shashedCanonicalRequest;
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 AKID*****
secret_id = ENV["TENCENTCLOUD_SECRET_ID"]
# 需要设置环境变量 TENCENTCLOUD_SECRET_KEY, 值为示例的 *****
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 AKID*****
    string SECRET_ID = Environment.GetEnvironmentVariable("TENCENTCLOUD_SECRET_ID");
    // 需要设置环境变量 TENCENTCLOUD_SECRET_KEY, 值为示例的 *****
    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 AKID*****
  const SECRET_ID = process.env.TENCENTCLOUD_SECRET_ID
  // 需要设置环境变量 TENCENTCLOUD_SECRET_KEY, 值为示例的 *****
  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 AKID*****
string SECRET_ID = getenv("TENCENTCLOUD_SECRET_ID");
// 需要设置环境变量 TENCENTCLOUD_SECRET_KEY, 值为示例的 *****
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++)
    {
        sprintf(buf, sizeof(buf), "%02x", hash[i]);
        strcat(result, buf);
    }
}

void hmac_sha256(const char* key, int key_len,
const char* input, int input_len,
```

```
unsigned char* output, unsigned int* output_len)
{
    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, key_len, EVP_sha256(), NULL);
    HMAC_Update(h, ( unsigned char* )input, input_len);
    HMAC_Final(h, output, output_len);

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

}

void hex_encode(const char* input, int input_len, char* output)
{
    static const char* const lut = "0123456789abcdef";

    char add_out[128] = {0};
    char temp[2] = {0};
    for (size_t i = 0; i < input_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 AKID*****
const char* SECRET_ID = getenv("TENCENTCLOUD_SECRET_ID");
// 需要设置环境变量 TENCENTCLOUD_SECRET_KEY, 值为示例的 *****
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, "\n\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\": \"i\nstance-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);
unsigned char k_date[64] = {0};
unsigned int output_len = 0;
hmac_sha256(k_key, strlen(k_key), date, strlen(date), k_date, &output_len);
unsigned char k_service[64] = {0};
hmac_sha256(k_date, output_len, service, strlen(service), k_service, &output_len);
unsigned char k_signing[64] = {0};
hmac_sha256(k_service, output_len, "tc3_request", strlen("tc3_request"), k_signing, &output_len);
unsigned char k_hmac_sha_sign[64] = {0};
hmac_sha256(k_signing, output_len, string_to_sign, strlen(string_to_sign), k_hmac_sha_sign, &output_len);

char signature[128] = {0};
hex_encode(k_hmac_sha_sign, output_len, 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符合，也有可能是密钥 SecretKey 错误导致的。
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：2024-12-25 01:47:46

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: AKID*****
- SecretKey: *****

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	AKID*****
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' : 'AKID*****',
  '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"，因此格式化后即为 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=AKID*****&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=AKID*****&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 = '*****';
$srcStr = 'GETcvm.tencentcloudapi.com/?Action=DescribeInstances&InstanceIds.0=ins-09dx96dg&Limit=20&Nonce=11886&Offset=0&Region=ap-guangzhou&SecretId=AKID*****&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:

```
7RAM2xfNMO9EiVTNmPg06MRnCvQ=
```

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 7RAM2xfNMO9EiVTNmPg06MRnCvQ=, 最终得到的签名串请求参数 (Signature) 为: 7RAM2xfNMO9EiVTNmPg06MRnCvQ%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 编码前需要先用 UTF-8 进行编码。

注意: 有些编程语言的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

错误代码	错误描述
AuthFailure.SecretIdNotFound	密钥不存在
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=AKID*****&Signature=7RAM2xfNM09EiVTNmPg06MRnCvQ%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 AKID*****
    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 *****
    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 AKID*****
secret_id = os.environ.get("TENCENTCLOUD_SECRET_ID")
# 需要设置环境变量 TENCENTCLOUD_SECRET_KEY, 值为示例的 *****
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 AKID*****
    secretId := os.Getenv("TENCENTCLOUD_SECRET_ID")
    // 需要设置环境变量 TENCENTCLOUD_SECRET_KEY, 值为示例的 *****
    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 AKID*****
$secretId = getenv("TENCENTCLOUD_SECRET_ID");
// 需要设置环境变量 TENCENTCLOUD_SECRET_KEY, 值为示例的 *****
$secretKey = getenv("TENCENTCLOUD_SECRET_KEY");
$params["Nonce"] = 11886;//rand();
$params["Timestamp"] = 1465185768;//time();
$params["Region"] = "ap-guangzhou";
$params["SecretId"] = $secretId;
$params["Version"] = "2017-03-12";
$params["Action"] = "DescribeInstances";
$params["InstanceIds.0"] = "ins-09dx96dg";
$params["Limit"] = 20;
$params["Offset"] = 0;

ksort($params);

$signStr = "GETcvm.tencentcloudapi.com/?";
foreach ( $params 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
// $params["Signature"] = $signature;
// $url = "https://cvm.tencentcloudapi.com/?".http_build_query($params);
// 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 AKID*****
secret_id = ENV["TENCENTCLOUD_SECRET_ID"]
# 需要设置环境变量 TENCENTCLOUD_SECRET_KEY, 值为示例的 *****
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 requestHost, 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 AKID*****
        string SECRET_ID = Environment.GetEnvironmentVariable("TENCENTCLOUD_SECRET_ID");
        // 需要设置环境变量 TENCENTCLOUD_SECRET_KEY, 值为示例的 *****
        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 AKID*****
  const SECRET_ID = process.env.TENCENTCLOUD_SECRET_ID
```

```
// 需要设置环境变量 TENCENTCLOUD_SECRET_KEY, 值为示例的 *****
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 01:40:02

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 06:40:25

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。

多网聚合加速相关接口

批量激活硬件设备

最近更新时间：2025-04-25 01:45:05

1. 接口描述

接口请求域名：mna.tencentcloudapi.com。

激活硬件设备

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：ActivateHardware。
Version	是	String	公共参数 ，本接口取值：2021-01-19。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
Hardware.N	是	Array of ActivateHardware	待激活的设备列表

3. 输出参数

参数名称	类型	描述
HardwareInfo	Array of ActivateHardware	完成激活的设备信息
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 批量激活设备

批量激活设备

输入示例

```
POST / HTTP/1.1
Host: mna.tencentcloudapi.com
Content-Type: application/json
X-TC-Action: ActivateHardware
```

<公共请求参数>

```
{
  "Hardware": [
    {
      "Vendor": "vendor",
      "SN": "AN",
      "DeviceName": "name",
      "Description": "activateHardware info",
      "DataKey": "keys"
    }
  ]
}
```

输出示例

```
{
  "Response": {
    "HardwareInfo": [
      {
        "Vendor": "vendor",
        "SN": "AN",
        "DeviceName": "name",
        "Description": "activateHardware info",
        "DataKey": "keys",
        "LicensePayMode": 1,
        "AccessScope": 1
      }
    ],
    "RequestId": "edd378f7-2511-4692-a029-5ca3d22c1884"
  }
}
```

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
InternalServerError	内部错误。
InternalServerError.ControlRequestError	请求控制器发生错误。
InternalServerError.DuplicateDeviceName	设备名已存在。
InvalidParameterValue	参数取值错误。
OperationDenied.HardwareHasActivated	输入SN对应的硬件已经激活
OperationDenied.HardwareNotExist	输入SN对应的硬件不存在。
OperationDenied.VendorNotRegister	当前账号尚未注册为厂商。

新建互通规则

最近更新时间：2025-04-25 01:45:05

1. 接口描述

接口请求域名：mna.tencentcloudapi.com。

新建互通规则

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：AddL3Conn。
Version	是	String	公共参数 ，本接口取值：2021-01-19。
Region	否	String	公共参数 ，此参数为可选参数。
Cidr1	是	String	设置互通网段CIDR1，支持：10.0.0.0 – 10.255.255.255，172.16.0.0 – 172.31.255.255，192.168.0.0 – 192.168.255.255 示例值："172.12.0.0/16"
Cidr2	是	String	设置互通网段CIDR2，支持：10.0.0.0 – 10.255.255.255，172.16.0.0 – 172.31.255.255，192.168.0.0 – 192.168.255.255 示例值："172.13.0.0/16"
DeviceId1	是	String	CIDR1对应的设备ID 示例值：mna-de234ff
DeviceId2	是	String	CIDR2对应的设备ID 示例值：cde
Description	否	String	规则描述 示例值：规则描述

3. 输出参数

参数名称	类型	描述
L3ConnId	String	互通规则ID 示例值："l3conn-4413p97xxx"
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 示例1

输入示例

```
POST / HTTP/1.1
Host: mna.tencentcloudapi.com
Content-Type: application/json
X-TC-Action: AddL3Conn
<公共请求参数>

{
  "Cidr1": "173.12.0.0/16",
  "Cidr2": "173.13.0.0/16",
  "DeviceId1": "mna-der44545r",
  "DeviceId2": "cde",
  "Description": "this is a test rule"
}
```

输出示例

```
{
  "Response": {
    "L3ConnId": "l3conn-h9boibynmp",
    "RequestId": "f0367096-b039-491c-b7eb-76669a2b22cf"
  }
}
```

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.ControlRequestError	请求控制器发生错误。
InvalidParameterValue	参数取值错误。
OperationDenied.L3CidrOverLap	互通规则CIDR重叠
OperationDenied.L3ConnectionOverSize	互通规则数超过最大限制150条

分组添加设备

最近更新时间：2025-04-25 01:45:01

1. 接口描述

接口请求域名：mna.tencentcloudapi.com。

向已存在分组中添加设备

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：GroupAddDevice。
Version	是	String	公共参数 ，本接口取值：2021-01-19。
Region	否	String	公共参数 ，此参数为可选参数。
GroupId	是	String	分组ID 示例值："1a2b3c4d5e"
DeviceList.N	是	Array of String	待添加的设备列表 示例值：["mna-test1","mna-test2"]

3. 输出参数

参数名称	类型	描述
DeviceNum	Integer	分组中的设备数量 示例值：5
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 示例1

输入示例

```
POST / HTTP/1.1
Host: mna.tencentcloudapi.com
Content-Type: application/json
X-TC-Action: GroupAddDevice
```

<公共请求参数>

```
{
  "DeviceList": [
    "mna-test1",
    "mna-test2"
  ],
  "GroupId": "group-id"
}
```

输出示例

```
{
  "Response": {
    "DeviceNum": 5,
    "RequestId": "a1434e98-16e8-41de-9b9b-27805a9cffbf"
  }
}
```

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
InternalServerError	内部错误。
InternalServerError.ControlRequestError	请求控制器发生错误。
InvalidParameterValue	参数取值错误。

新建设备

最近更新时间：2025-04-25 01:45:05

1. 接口描述

接口请求域名：mna.tencentcloudapi.com。

新建设备记录

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：AddDevice。
Version	是	String	公共参数 ，本接口取值：2021-01-19。
Region	否	String	公共参数 ，本接口不需要传递此参数。
DeviceName	是	String	新建设备的名称 示例值：dev1
Remark	否	String	新建设备的备注 示例值：mobile
DataKey	否	String	新建设备的base64密钥字符串，非必选，如果不填写则由系统自动生成 示例值：dasdfrfwer32e4r
Encrypted	否	Boolean	是否设置预置密钥 示例值：true
AccessScope	否	Integer	接入环境。0：公有云网关；1：自有网关；2：公有云网关和自有网关。不填默认公有云网关。 具体含义： 公有云网关：即该设备只能接入公有云网关（就近接入） 自有网关：即该设备只能接入已经注册上线的自有网关（就近接入或固定ip接入） 公有云网关和自有网关：即该设备同时可以接入公有云网关和已经注册上线的自有网关（就近接入或固定ip接入） 示例值：0
LicensePayMode	否	Integer	license付费方式： 0，月度授权 1，永久授权 若不传则默认为月度授权，永久授权设备需要调用OrderPerLicense接口支付授权费，否则设备无法使用 示例值：0
GroupName	否	String	设备分组名称，非必选，预留参数，需要分组时传入GroupId 示例值：group-test

参数名称	必选	类型	描述
GroupId	否	String	设备分组ID，非必选，如果不填写则默认设备无分组 示例值：1a2b3c4d5e
FlowTrunc	否	Integer	设备无流量包处理方式，0: 按量付费，1: 截断加速 示例值：0

3. 输出参数

参数名称	类型	描述
DataKey	String	经过加密算法加密后的base64格式密钥 示例值：dasdfrfwer32e4r
DeviceId	String	设备ID 示例值：mna-w795bzezug
Signature	String	签名字符串 示例值："test"
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 新建设备

输入示例

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

{
  "DeviceName": "mna-test1",
  "Remark": "mna-test1",
  "DataKey": "mna-test1",
  "Encrypted": "false"
}
```

输出示例

```
{
  "Response": {
    "RequestId": "527cc5c7-0413-33e9-2adc-632e0f6a9dff",
    "DeviceId": "mna-test1",
    "DataKey": "dasdfrfwer32e4r",
    "Signature": "dasdfrfwer32e4r"
  }
}
```

```
}  
  
}
```

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
InternalServerError	内部错误。
InternalServerError.ControlRequestError	请求控制器发生错误。
InternalServerError.DuplicateDataKey	设备密钥已存在。
InternalServerError.DuplicateDeviceName	设备名已存在。
InternalServerError.UndefinedEncryptedKey	预置密钥尚未创建。
InvalidParameterValue	参数取值错误。

分组删除设备

最近更新时间：2025-04-25 01:45:01

1. 接口描述

接口请求域名：mna.tencentcloudapi.com。

删除分组中的设备

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：GroupDeleteDevice。
Version	是	String	公共参数 ，本接口取值：2021-01-19。
Region	否	String	公共参数 ，此参数为可选参数。
GroupId	是	String	分组ID 示例值："1a2b3c4d5e"
DeviceList.N	是	Array of String	待删除的设备列表 示例值：["mna-test1","mna-test2"]

3. 输出参数

参数名称	类型	描述
DeviceNum	Integer	分组中的设备数量 示例值：5
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 示例1

输入示例

```
POST / HTTP/1.1
Host: mna.tencentcloudapi.com
Content-Type: application/json
X-TC-Action: GroupDeleteDevice
```

<公共请求参数>

```
{
  "DeviceList": [
    "mna-test1",
    "mna-test2"
  ],
  "GroupId": "group-id1"
}
```

输出示例

```
{
  "Response": {
    "DeviceNum": 5,
    "RequestId": "a1434e98-16e8-41de-9b9b-27805a9cffbf"
  }
}
```

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
InternalServerError	内部错误。
InternalServerError.ControlRequestError	请求控制器发生错误。
InvalidParameterValue	参数取值错误。

新建分组

最近更新时间：2025-04-25 01:45:05

1. 接口描述

接口请求域名：mna.tencentcloudapi.com。

新建分组

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：AddGroup。
Version	是	String	公共参数 ，本接口取值：2021-01-19。
Region	否	String	公共参数 ，此参数为可选参数。
GroupName	是	String	分组的名称 示例值："group-test"
Description	否	String	分组的描述 示例值："description for group-test"

3. 输出参数

参数名称	类型	描述
GroupId	String	分组的唯一ID，仅做分组唯一区分 示例值：1a2b3c4d5e
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 示例1

输入示例

```
POST / HTTP/1.1
Host: mna.tencentcloudapi.com
Content-Type: application/json
X-TC-Action: AddGroup
```


<公共请求参数>

```
{
  "GroupName": "gname",
  "Description": "AddGroup info"
}
```

输出示例

```
{
  "Response": {
    "RequestId": "a1434e98-16e8-41de-9b9b-27805a9cffbf",
    "GroupId": "1a2b3c4d5e"
  }
}
```

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
InternalServerError	内部错误。
InternalServerError.ControlRequestError	请求控制器发生错误。

错误码	描述
InvalidParameterValue	参数取值错误。

添加硬件设备

最近更新时间：2025-04-25 01:45:05

1. 接口描述

接口请求域名：mna.tencentcloudapi.com。

添加硬件设备，生成未激活的硬件设备，可支持批量添加

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：AddHardware。
Version	是	String	公共参数 ，本接口取值：2021-01-19。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
Hardware.N	是	Array of Hardware	硬件列表

3. 输出参数

参数名称	类型	描述
Hardware	Array of Hardware	硬件设备
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 新建硬件设备

批量新建硬件设备

输入示例

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

```
{
  "Hardware": [
    {
      "SN": "AN",
      "LicenseChargingMode": 1,
      "Description": "AddHardware info"
    }
  ]
}
```

输出示例

```
{
  "Response": {
    "Hardware": [
      {
        "SN": "AN",
        "LicenseChargingMode": 1,
        "Description": "AddHardware描述",
        "HardwareId": "cpe-9oii2ew1z4"
      }
    ],
    "RequestId": "a1434e98-16e8-41de-9b9b-27805a9cffbf"
  }
}
```

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.ControlRequestError	请求控制器发生错误。
InvalidParameterValue	参数取值错误。
OperationDenied.DuplicateSN	SN已存在
OperationDenied.VendorNotRegister	当前账号尚未注册为厂商。

设置或更新密钥

最近更新时间：2025-04-25 01:45:05

1. 接口描述

接口请求域名：mna.tencentcloudapi.com。

通过此接口设置和更新预置密钥

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：CreateEncryptedKey。
Version	是	String	公共参数 ，本接口取值：2021-01-19。
Region	否	String	公共参数 ，本接口不需要传递此参数。

3. 输出参数

参数名称	类型	描述
EncryptedKey	String	预置密钥 示例值："BgkqhkiG9w0BAQ"
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 设置预置密钥

输入示例

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

{ }
```

输出示例

```
{
  "Response": {
    "RequestId": "edd378f7-2511-4692-a029-5ca3d22c1884",
    "EncryptedKey": "BgkqhkiG9w0BAQ"
  }
}
```

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
InternalServerError	内部错误。

删除设备

最近更新时间：2025-04-25 01:45:04

1. 接口描述

接口请求域名：mna.tencentcloudapi.com。

删除设备信息

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：DeleteDevice。
Version	是	String	公共参数 ，本接口取值：2021-01-19。
Region	否	String	公共参数 ，本接口不需要传递此参数。
DeviceId	是	String	删除设备的唯一ID 示例值："mna-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 删除设备接口

输入示例

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

{
```



```
"DeviceId": "mna-xxx"
}
```

输出示例

```
{
  "Response": {
    "RequestId": "xxx"
  }
}
```

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
InternalServerError	内部错误。
InternalServerError.ControlRequestError	请求控制器发生错误。
InvalidParameterValue	参数取值错误。

删除互通规则

最近更新时间：2025-04-25 01:45:04

1. 接口描述

接口请求域名：mna.tencentcloudapi.com。

删除互通规则

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：DeleteL3Conn。
Version	是	String	公共参数 ，本接口取值：2021-01-19。
Region	否	String	公共参数 ，此参数为可选参数。
L3ConnIdList.N	是	Array of String	互通规则ID列表 示例值：["l3conn-zfd9rpvxxx"]

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 示例1

输入示例

```
POST / HTTP/1.1
Host: mna.tencentcloudapi.com
Content-Type: application/json
X-TC-Action: DeleteL3Conn
<公共请求参数>

{
  "L3ConnIdList": [
    "lcn-test1"
```

```
]
}
```

输出示例

```
{
  "Response": {
    "RequestId": "bd89e515-20b4-445b-a88d-7355e76f8d22"
  }
}
```

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
InternalServerError	内部错误。
InternalServerError.ControlRequestError	请求控制器发生错误。
InvalidParameterValue	参数取值错误。

删除分组

最近更新时间：2025-04-25 01:45:04

1. 接口描述

接口请求域名：mna.tencentcloudapi.com。

删除分组

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：DeleteGroup。
Version	是	String	公共参数 ，本接口取值：2021-01-19。
Region	否	String	公共参数 ，此参数为可选参数。
GroupId	是	String	删除指定分组 示例值：1a2b3c4d5e

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 示例1

输入示例

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

{
```

```
"GroupId": "group-1"
}
```

输出示例

```
{
  "Response": {
    "RequestId": "bd89e515-20b4-445b-a88d-7355e76f8d22"
  }
}
```

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
InternalServerError	内部错误。
InternalServerError.ControlRequestError	请求控制器发生错误。
InvalidParameterValue	参数取值错误。

获取分组详细信息

最近更新时间：2025-04-25 01:45:03

1. 接口描述

接口请求域名：mna.tencentcloudapi.com。

查看分组详细信息

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：GetGroupDetail。
Version	是	String	公共参数 ，本接口取值：2021-01-19。
Region	否	String	公共参数 ，此参数为可选参数。
GroupId	是	String	分组ID 示例值：1a2b3c4d5e
PageSize	是	Integer	每页显示记录数，PageSize、PageNumber值均为-1 时，按照1页无限制条数匹配所有设备 示例值：1
PageNumber	是	Integer	每页显示记录数，PageSize、PageNumber值均为-1 时，按照1页无限制条数匹配所有设备 示例值：10
KeyWord	否	String	搜索关键字 示例值：keywords

3. 输出参数

参数名称	类型	描述
GroupInfo	GroupInfo	分组基本信息
DeviceInfos	Array of DeviceBaseInfo	分组中设备列表
Length	Integer	设备总记录条数 示例值：20
TotalPage	Integer	总页数 示例值：2
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 示例1

输入示例

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

{
  "GroupId": "cliGrp-xf8rboasb",
  "PageSize": "10",
  "PageNumber": "1"
}
```

输出示例

```
{
  "Response": {
    "GroupInfo": {
      "GroupId": "cliGrp-xf8rboasb",
      "GroupName": "groupname",
      "CreateTime": "1711076263000",
      "UpdateTime": "1711076263000",
      "Description": "GetGroupDetail描述",
      "DeviceNum": 0
    },
    "DeviceInfos": [
      {
        "DeviceId": "mna-det244",
        "DeviceName": "name",
        "CreateTime": "1711076263000",
        "LastTime": "1711076263000",
        "Remark": "mark",
        "AccessScope": 0,
        "LicensePayMode": 0,
        "Payer": 0,
        "GroupId": "cliGrp-xf8rboasb",
        "GroupName": "name2"
      }
    ],
    "Length": 0,
    "TotalPage": 0,
    "RequestId": "fd429f79-9914-46e7-94c3-c0695ee1c8b9"
  }
}
```

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
InternalServerError	内部错误。
InternalServerError.ControlRequestError	请求控制器发生错误。
InvalidParameterValue	参数取值错误。

获取设备详细信息

最近更新时间：2025-04-25 01:45:04

1. 接口描述

接口请求域名：mna.tencentcloudapi.com。

通过指定设备的ID查找设备详细信息

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：GetDevice。
Version	是	String	公共参数 ，本接口取值：2021-01-19。
Region	否	String	公共参数 ，本接口不需要传递此参数。
DeviceId	是	String	搜索指定设备的id 示例值："mna-xxx"

3. 输出参数

参数名称	类型	描述
DeviceDetails	DeviceDetails	设备详细信息
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 获取指定id设备信息

获取设备基本信息和网络信息

输入示例

```
{
  "DeviceId": "mna-xxx"
}
```

输出示例

```
{
  "Response": {
    "DeviceDetails": {
      "BusinessDownRate": 0,
      "BusinessUpRate": 443292176.8,
      "DeviceBaseInfo": {
        "CreateTime": "1662722094000",
        "DeviceId": "mna-94p8c5zyst",
        "DeviceName": "yusheng-test2",
        "LastTime": "1675242602000",
        "Remark": "yusheng-test"
      },
      "DeviceNetInfo": [
        {
          "DataEnable": false,
          "DataRx": 0,
          "DataTx": 0,
          "DownRate": 463128.799998,
          "DownloadLimit": "0",
          "NetInfoName": "eth0",
          "PublicIp": "9.223.110.232",
          "Rat": 0,
          "SignalStrength": 0,
          "State": 0,
          "Type": 0,
          "UpRate": 27040465.6,
          "UploadLimit": "0",
          "Vendor": 0
        },
        {
          "DataEnable": false,
          "DataRx": 0,
          "DataTx": 0,
          "DownRate": 4415987.2,
          "DownloadLimit": "0",
          "NetInfoName": "eth1",
          "PublicIp": "9.223.96.218",
          "Rat": 0,
          "SignalStrength": 0,
          "State": 0,
          "Type": 0,
          "UpRate": 428871766.399999,
          "UploadLimit": "0",
          "Vendor": 0
        }
      ],
      "GatewaySite": "gz"
    },
    "RequestId": "2ca4e2e9-f4a1-4b24-b88a-b7523099c123"
```

```
}  
  
}
```

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
InternalServerError	内部错误。
InternalServerError.ControlRequestError	请求控制器发生错误。
InvalidParameterValue	参数取值错误。

获取互通规则列表

最近更新时间：2025-04-25 01:45:02

1. 接口描述

接口请求域名：mna.tencentcloudapi.com。

获取互通规则列表

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：GetL3ConnList。
Version	是	String	公共参数 ，本接口取值：2021-01-19。
Region	否	String	公共参数 ，此参数为可选参数。
PageSize	是	Integer	每页显示记录数，PageSize、PageNumber值均为-1 时，按照1页无限制条数匹配所有设备 示例值：10
PageNumber	是	Integer	当前查看页码，PageSize、PageNumber值均为-1 时，按照1页无限制条数匹配所有设备 示例值：1
DeviceId	否	String	搜索分组的DeviceId，为空时匹配所有分组 示例值：devid

3. 输出参数

参数名称	类型	描述
L3ConnList	Array of L3ConnInfo	互通规则列表
Length	Integer	设备总记录条数 示例值：20
TotalPage	Integer	总页数 示例值：2
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 示例1

输入示例

```
POST / HTTP/1.1
Host: mna.tencentcloudapi.com
Content-Type: application/json
X-TC-Action: GetL3ConnList
<公共请求参数>

{
  "PageSize": 10,
  "PageNumber": 1,
  "DeviceId": "mna-2x2t1lhb18"
}
```

输出示例

```
{
  "Response": {
    "L3ConnList": [
      {
        "L3ConnId": "l3conn-h9boibynmp",
        "DeviceId1": "mna-2x2t1lhb18",
        "Cidr1": "192.168.1.0/26",
        "DeviceId2": "mna-2x2t1lhb13",
        "Cidr2": "192.168.1.0/26",
        "Enable": true,
        "Description": "this is descript"
      }
    ],
    "Length": 1,
    "TotalPage": 1,
    "RequestId": "e5b299c7-aaf4-4a5e-9d01-feb63273e347"
  }
}
```

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
InternalServerError	内部错误。
InternalServerError.ControlRequestError	请求控制器发生错误。
InvalidParameterValue	参数取值错误。

获取分组列表

最近更新时间：2025-04-25 01:45:02

1. 接口描述

接口请求域名：mna.tencentcloudapi.com。

获取分组列表

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：GetGroupList。
Version	是	String	公共参数 ，本接口取值：2021-01-19。
Region	否	String	公共参数 ，此参数为可选参数。
PageSize	是	Integer	每页显示记录数，PageSize、PageNumber值均为-1 时，按照1页无限制条数匹配所有设备 示例值：10
PageNumber	是	Integer	当前查看页码，PageSize、PageNumber值均为-1 时，按照1页无限制条数匹配所有设备 示例值：1
Keyword	否	String	搜索分组的关键字，为空时匹配所有分组 示例值："keywords"

3. 输出参数

参数名称	类型	描述
GroupInfos	Array of GroupInfo	设备信息列表
Length	Integer	设备总记录条数 示例值：20
TotalPage	Integer	总页数 示例值：2
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 示例1

输入示例

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

{
  "PageSize": "10",
  "PageNumber": "1",
  "Keyword": "Keyword"
}
```

输出示例

```
{
  "Response": {
    "GroupInfos": [
      {
        "GroupId": "group-id1",
        "GroupName": "gname",
        "CreateTime": "1734401551",
        "UpdateTime": "1734401551",
        "Description": "GetGroupList描述",
        "DeviceNum": 5
      }
    ],
    "Length": 0,
    "TotalPage": 0,
    "RequestId": "a1434e98-16e8-41de-9b9b-27805a9cffbf"
  }
}
```

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
InternalServerError	内部错误。
InternalServerError.ControlRequestError	请求控制器发生错误。
InvalidParameterValue	参数取值错误。

设备信息列表

最近更新时间：2025-04-25 01:45:03

1. 接口描述

接口请求域名：mna.tencentcloudapi.com。

获取设备信息列表

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：GetDevices。
Version	是	String	公共参数 ，本接口取值：2021-01-19。
Region	否	String	公共参数 ，本接口不需要传递此参数。
PageSize	是	Integer	每页显示记录数，PageSize、PageNumber值均为-1 时，按照1页无限制条数匹配所有设备 示例值：10
PageNumber	是	Integer	当前查看页码，PageSize、PageNumber值均为-1 时，按照1页无限制条数匹配所有设备 示例值：1
Keyword	否	String	搜索设备的关键字（ID或者设备名），为空时匹配所有设备 示例值："mna-xxx"
DeviceType	否	Integer	DeviceType 不传：返回所有设备； 1:自有设备； 2:三方设备 示例值：1

3. 输出参数

参数名称	类型	描述
DeviceInfos	Array of DeviceBaseInfo	设备信息列表
Length	Integer	设备总记录条数 示例值：20
TotalPage	Integer	总页数 示例值：2

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 示例

输入示例

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

{
  "PageSize": 1,
  "PageNumber": 1
}
```

输出示例

```
{
  "Response": {
    "DeviceInfos": [
      {
        "AccessScope": 0,
        "CreateTime": "1663310188000",
        "DeviceId": "mna-w795bzezug",
        "DeviceName": "dev2233",
        "FlowTrunc": 0,
        "GroupId": "",
        "GroupName": "",
        "LastTime": "1719454676000",
        "LicensePayMode": 0,
        "Payer": 0,
        "Remark": "",
        "Sn": "",
        "Vendor": ""
      }
    ],
    "Length": 899,
    "RequestId": "8fbce821-acbb-49db-ad70-345b5d353324",
    "TotalPage": 899
  }
}
```

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
InternalServerError	内部错误。
InternalServerError.ControlRequestError	请求控制器发生错误。
InvalidParameterValue	参数取值错误。

根据设备组获取数据流量统计数据

最近更新时间：2025-04-25 01:45:03

1. 接口描述

接口请求域名：mna.tencentcloudapi.com。

获取指定分组，指定时间数据流量使用情况

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：GetFlowStatisticByGroup。
Version	是	String	公共参数 ，本接口取值：2021-01-19。
Region	否	String	公共参数 ，此参数为可选参数。
GroupId	是	String	分组ID 示例值：1a2b3c4d5e
BeginTime	是	Integer	开始查找时间 示例值：1659514436
EndTime	是	Integer	截止时间 示例值：1659515436
Type	是	Integer	流量种类（1：上行流量，2：下行流量，3：上下行总和） 示例值：1
TimeGranularity	是	Integer	时间粒度（1：按小时统计，2：按天统计） 示例值：1
AccessRegion	否	String	接入区域。取值范围：['MC','AP','EU','AM'] MC=中国大陆 AP=亚太 EU=欧洲 AM=美洲。不填代表全量区域。 示例值：MC
GatewayType	否	Integer	网关类型。0：公有云网关；1：自有网关。不传默认为0。 示例值：0

3. 输出参数

参数名称	类型	描述
NetDetails	Array of NetDetails	流量详细信息

参数名称	类型	描述
MaxValue	Float	查找时间段流量使用最大值（单位：byte） 示例值：216516
AvgValue	Float	查找时间段流量使用平均值（单位：byte） 示例值：25151
TotalValue	Float	查找时间段流量使用总量（单位：byte） 示例值：35465464
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 根据分组获取用量统计信息

输入示例

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

{
  "GroupId": "cliGrp-xf8rboasbh",
  "BeginTime": 1711296000,
  "EndTime": 1711987200,
  "Type": 1,
  "TimeGranularity": 2,
  "AccessRegion": "MC",
  "GatewayType": 0
}
```

输出示例

```
{
  "Response": {
    "AvgValue": 154434162.5,
    "MaxValue": 305576473,
    "NetDetails": [
      {
        "Current": 305576473,
        "Time": "1711555200"
      },
      {
        "Current": 3291852,
        "Time": "1711641600"
      }
    ]
  }
}
```

```
"RequestId": "e5b650a9-d994-439f-9132-3e5c56054903",
"TotalValue": 308868325
}
}
```

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 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
InternalServerError	内部错误。
InternalServerError.ControlRequestError	请求控制器发生错误。
InternalServerError.MonitorDataRequestError	监控数据请求错误。
InternalServerError.NetworkInfoRequestError	智研流量数据请求错误。
InvalidParameterValue	参数取值错误。

获取数据流量统计数据

最近更新时间：2025-04-25 01:45:03

1. 接口描述

接口请求域名：mna.tencentcloudapi.com。

获取指定设备Id，指定时间点数据流量使用情况

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：GetFlowStatistic。
Version	是	String	公共参数 ，本接口取值：2021-01-19。
Region	否	String	公共参数 ，此参数为可选参数。
DeviceId	是	String	设备ID 示例值：245aee95-f37e-55d0-5258-19520bdf4b53
BeginTime	是	Integer	开始查找时间 示例值：1659514436
EndTime	是	Integer	截止时间 示例值：1659515436
Type	是	Integer	流量种类（1：上行流量，2：下行流量，3：上下行总和） 示例值：1
TimeGranularity	是	Integer	时间粒度（1：按小时统计，2：按天统计） 示例值：1
AccessRegion	否	String	接入区域。取值范围：['MC','AP','EU','AM'] MC=中国大陆 AP=亚太 EU=欧洲 AM=美洲。不填代表全量区域。 示例值：MC
GatewayType	否	Integer	网关类型。0：公有云网关；1：自有网关。不传默认为0。 示例值：0
DeviceList.N	否	Array of String	设备ID列表，用于查询多设备流量，该字段启用时DeviceId可传"-1" 示例值：["mna-test1","mna-test2"]

3. 输出参数

参数名称	类型	描述
NetDetails	Array of NetDetails	流量详细信息
MaxValue	Float	查找时间段流量使用最大值（单位：byte） 示例值：216516
AvgValue	Float	查找时间段流量使用平均值（单位：byte） 示例值：25151
TotalValue	Float	查找时间段流量使用总量（单位：byte） 示例值：35465464
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 获取指定id设备流量使用信息

获取指定id设备流量使用信息

输入示例

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

{
  "EndTime": 1659514692,
  "Type": 1,
  "DeviceId": "mna-dev1",
  "TimeGranularity": 1,
  "BeginTime": 1659513692
}
```

输出示例

```
{
  "Response": {
    "MaxValue": 51548,
    "AvgValue": 15441,
    "TotalValue": 656546,
    "NetDetails": [
      {
        "Time": "1659513692",
        "Current": 3546545
      },
      {
        "Time": "1659513692",
```

```
"Current": 4454864
},
{
  "Time": "1659513692",
  "Current": 4848941
}
],
"RequestId": ""
}
```

示例2 获取多个设备的流量使用信息之和

输入示例

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

{
  "EndTime": 1659514692,
  "Type": 1,
  "DeviceId": "",
  "DeviceList": [
    "mna-dev1",
    "mna-dev2"
  ],
  "TimeGranularity": 1,
  "BeginTime": 1659513692
}
```

输出示例

```
{
  "Response": {
    "MaxValue": 51548,
    "AvgValue": 15441,
    "TotalValue": 656546,
    "NetDetails": [
      {
        "Time": "1659513692",
        "Current": 3546545
      },
      {
        "Time": "1659513692",
        "Current": 4454864
      },
    ]
  }
}
```

```
{
  "Time": "1659513692",
  "Current": 4848941
},
{
  "RequestId": ""
}
```

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 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
InternalServerError	内部错误。
InternalServerError.ControlRequestError	请求控制器发生错误。
InternalServerError.MonitorDataRequestError	监控数据请求错误。
InternalServerError.NetworkInfoRequestError	智研流量数据请求错误。
InvalidParameterValue	参数取值错误。

根据区域获取数据流量统计数据

最近更新时间：2025-04-25 01:45:03

1. 接口描述

接口请求域名：mna.tencentcloudapi.com。

获取指定区域，指定时间点数据流量使用情况

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：GetFlowStatisticByRegion。
Version	是	String	公共参数 ，本接口取值：2021-01-19。
Region	否	String	公共参数 ，此参数为可选参数。
BeginTime	是	Integer	开始查找时间 示例值：1659514436
EndTime	是	Integer	截止时间 示例值：1659515436
Type	是	Integer	流量种类（1：上行流量，2：下行流量，3：上下行总和） 示例值：1
TimeGranularity	是	Integer	时间粒度（1：按小时统计，2：按天统计） 示例值：1
GatewayType	是	Integer	网关类型。0：公有云网关；1：自有网关。 示例值：0
AccessRegion	否	String	接入区域。取值范围：['MC','AP','EU','AM'] MC=中国大陆 AP=亚太 EU=欧洲 AM=美洲。不填代表全量区域。 示例值：MC

3. 输出参数

参数名称	类型	描述
NetDetails	Array of NetDetails	流量详细信息
MaxValue	Float	查找时间段流量使用最大值（单位：byte） 示例值：216516

参数名称	类型	描述
AvgValue	Float	查找时间段流量使用平均值（单位：byte） 示例值：25151
TotalValue	Float	查找时间段流量使用总量（单位：byte） 示例值：35465464
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 正常场景

输入示例

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

{
  "BeginTime": 0,
  "EndTime": 0,
  "Type": 0,
  "TimeGranularity": 0,
  "AccessRegion": "MC",
  "GatewayType": 0
}
```

输出示例

```
{
  "Response": {
    "NetDetails": [
      {
        "Current": 0,
        "Time": "1735259400"
      }
    ],
    "MaxValue": 0,
    "AvgValue": 0,
    "TotalValue": 0,
    "RequestId": "bd89e515-20b4-445b-a88d-7355e76f8d22"
  }
}
```

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
InternalServerError	内部错误。
InternalServerError.ControlRequestError	请求控制器发生错误。
InternalServerError.MonitorDataRequestError	监控数据请求错误。
InternalServerError.NetworkInfoRequestError	智研流量数据请求错误。
InvalidParameterValue	参数取值错误。

获取厂商硬件列表

最近更新时间：2025-04-25 01:45:02

1. 接口描述

接口请求域名：mna.tencentcloudapi.com。

获取厂商硬件列表

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：GetHardwareList。
Version	是	String	公共参数 ，本接口取值：2021-01-19。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
PageNumber	是	Integer	页码 示例值：1
PageSize	是	Integer	页面设备数量 示例值：10
Keyword	否	String	关键字 示例值：keywords

3. 输出参数

参数名称	类型	描述
HardwareInfos	Array of HardwareInfo	硬件信息列表
Length	Integer	硬件总数 示例值：12
TotalPage	Integer	总页数 示例值：3
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 示例

输入示例

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

{
  "PageNumber": 1,
  "PageSize": 1
}
```

输出示例

```
{
  "Response": {
    "HardwareInfos": [
      {
        "ActiveTime": "1686554927",
        "CreateTime": "1685673786",
        "Description": "测试00-00",
        "DeviceId": "mna-2x2tllhb18",
        "DeviceName": "test-vendor-hardware-00-00",
        "FlowTrunc": 0,
        "GroupId": "",
        "GroupName": "",
        "LicenseChargingMode": 2,
        "LicensePayMode": 0,
        "Payer": 1,
        "SN": "test-vendor-hardware-00",
        "VendorDescription": "测试00"
      }
    ],
    "Length": 107,
    "RequestId": "50f72657-2085-49fe-839f-71f3088c7f0a",
    "TotalPage": 107
  }
}
```

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
InternalServerError	内部错误。
InternalServerError.ControlRequestError	请求控制器发生错误。
InvalidParameterValue	参数取值错误。
OperationDenied.VendorNotRegister	当前账号尚未注册为厂商。

批量获取设备流量统计

最近更新时间：2025-04-25 01:45:02

1. 接口描述

接口请求域名：mna.tencentcloudapi.com。

批量获取设备流量统计曲线

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：GetMultiFlowStatistic。
Version	是	String	公共参数 ，本接口取值：2021-01-19。
Region	否	String	公共参数 ，此参数为可选参数。
DeviceIds.N	是	Array of String	设备id列表，单次最多请求10个设备 示例值：["mna-dev1"]
BeginTime	是	Integer	1659514436 示例值：1659514436
EndTime	是	Integer	1659515000 示例值：1659515000
Type	是	Integer	统计流量类型（1：上行流量，2：下行流量，3：上下行总和） 示例值：1
TimeGranularity	是	Integer	统计时间粒度（1：按小时统计，2：按天统计） 示例值：1
AccessRegion	否	String	接入区域。取值范围：['MC','AP','EU','AM'] MC=中国大陆 AP=亚太 EU=欧洲 AM=美洲。不填代表全量区域。 示例值：MC
GatewayType	否	Integer	网关类型。0：公有云网关；1：自有网关。不传默认为0。 示例值：0

3. 输出参数

参数名称	类型	描述
FlowDetails	Array of FlowDetails	批量设备流量信息

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 批量设备流量使用信息

批量获取指定设备列表流量使用信息

输入示例

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

{
  "DeviceIds": [
    "mna-obzuio2pij",
    "mna-ps9x7eako2"
  ],
  "BeginTime": 1675255591,
  "EndTime": 1675860391,
  "Type": 1,
  "TimeGranularity": 1
}
```

输出示例

```
{
  "Response": {
    "FlowDetails": [
      {
        "AvgValue": 26524619322,
        "DeviceId": "mna-obzuio2pij",
        "MaxValue": 44183705788,
        "NetDetails": [
          {
            "Current": 421334,
            "Time": "1675756800"
          },
          {
            "Current": 32279199625,
            "Time": "1675760400"
          },
          {
            "Current": 44183705788,
            "Time": "1675771200"
          }
        ]
      }
    ]
  }
}
```

```
},
{
  "Current": 29635150541,
  "Time": "1675774800"
},
{
  "TotalValue": 106098477288
},
{
  "AvgValue": 31803650050.5,
  "DeviceId": "mna-ps9x7eako2",
  "MaxValue": 50892172491,
  "NetDetails": [
    {
      "Current": 22711,
      "Time": "1675756800"
    },
    {
      "Current": 46732569185,
      "Time": "1675760400"
    },
    {
      "Current": 29589835815,
      "Time": "1675771200"
    },
    {
      "Current": 50892172491,
      "Time": "1675774800"
    }
  ],
  "TotalValue": 127214600202
},
{
  "RequestId": "ffad4f52-505f-4563-9431-b2ca1b5a887e"
}
```

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
InternalServerError	内部错误。
InternalServerError.MonitorDataRequestError	监控数据请求错误。
InvalidParameterValue	参数取值错误。

获取流量监控信息

最近更新时间：2025-04-25 01:45:02

1. 接口描述

接口请求域名：mna.tencentcloudapi.com。

获取单设备的实时流量统计指标

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：GetNetMonitor。
Version	是	String	公共参数 ，本接口取值：2021-01-19。
Region	否	String	公共参数 ，此参数为可选参数。
DeviceId	是	String	设备id 示例值："mna-xxx"
BeginTime	是	Integer	开始时间 示例值：1662622200
EndTime	是	Integer	结束时间 示例值：1662624800
Metrics	是	String	统计指标（上行速率："TxRate":bit/s，下行速率："RxRate":bit/s，丢包："Loss":%，时延："RTT":ms） 示例值："TxRate"
GatewayType	否	Integer	网关类型。0：公有云网关；1：自有网关。不传默认为0。 示例值：0

3. 输出参数

参数名称	类型	描述
MonitorData	Array of MonitorData	监控数据
AccessRegion	String	接入区域。取值范围：['MC','AP','EU','AM'] MC=中国大陆 AP=亚太 EU=欧洲 AM=美洲 示例值：MC

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 获取流量监控信息

获取流量监控信息

输入示例

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

{
  "Metrics": "TxRate",
  "EndTime": 1662624800,
  "DeviceId": "mna-xxx",
  "BeginTime": 1662622200
}
```

输出示例

```
{
  "Response": {
    "MonitorData": [
      {
        "Time": "1673601780",
        "BusinessMetrics": 492350276.319998,
        "SlotNetInfo": [
          {
            "NetInfoName": "eth0",
            "PublicIP": "9.223.110.232",
            "Current": 27582845.919998
          },
          {
            "NetInfoName": "eth1",
            "PublicIP": "9.223.96.218",
            "Current": 395158790.239997
          }
        ]
      },
      {
        "Time": "1673601840",
        "BusinessMetrics": 412655416,
        "SlotNetInfo": [
```

```
{
  "NetInfoName": "eth0",
  "PublicIP": "9.223.110.232",
  "Current": 27228367.066665
},
{
  "NetInfoName": "eth1",
  "PublicIP": "9.223.96.218",
  "Current": 397505721.866666
}
],
{
  "Time": "1673601900",
  "BusinessMetrics": 411823866.666665,
  "SlotNetInfo": [
    {
      "NetInfoName": "eth0",
      "PublicIP": "9.223.110.232",
      "Current": 28047154.266665
    },
    {
      "NetInfoName": "eth1",
      "PublicIP": "9.223.96.218",
      "Current": 396059005.333333
    }
  ],
  "Time": "1673601960",
  "BusinessMetrics": 412108687.733331,
  "SlotNetInfo": [
    {
      "NetInfoName": "eth0",
      "PublicIP": "9.223.110.232",
      "Current": 29244132.933331
    },
    {
      "NetInfoName": "eth1",
      "PublicIP": "9.223.96.218",
      "Current": 395156858.266664
    }
  ],
  "Time": "1673602020",
  "BusinessMetrics": 412778279.866666,
  "SlotNetInfo": [
    {
      "NetInfoName": "eth0",
```



```
"PublicIP": "9.223.110.232",
"Current": 26898350.666666
},
{
  "NetInfoName": "eth1",
  "PublicIP": "9.223.96.218",
  "Current": 398191465.733331
}
],
{
  "Time": "1673602080",
  "BusinessMetrics": 412630582.533333,
  "SlotNetInfo": [
    {
      "NetInfoName": "eth0",
      "PublicIP": "9.223.110.232",
      "Current": 28580784.266665
    },
    {
      "NetInfoName": "eth1",
      "PublicIP": "9.223.96.218",
      "Current": 396357362.933332
    }
  ]
},
{
  "Time": "1673602140",
  "BusinessMetrics": 412431913.466665,
  "SlotNetInfo": [
    {
      "NetInfoName": "eth0",
      "PublicIP": "9.223.110.232",
      "Current": 27329174.8
    },
    {
      "NetInfoName": "eth1",
      "PublicIP": "9.223.96.218",
      "Current": 397404181.866666
    }
  ]
},
{
  "Time": "1673602200",
  "BusinessMetrics": 410982346.266665,
  "SlotNetInfo": [
    {
      "NetInfoName": "eth0",
      "PublicIP": "9.223.110.232",
      "Current": 29038580.399999
```

```
},
{
  "NetInfoName": "eth1",
  "PublicIP": "9.223.96.218",
  "Current": 394201611.866666
}
],
},
{
  "Time": "1673602260",
  "BusinessMetrics": 409923330.933332,
  "SlotNetInfo": [
    {
      "NetInfoName": "eth0",
      "PublicIP": "9.223.110.232",
      "Current": 28212205.599999
    },
    {
      "NetInfoName": "eth1",
      "PublicIP": "9.223.96.218",
      "Current": 393937825.066666
    }
  ]
},
{
  "Time": "1673602320",
  "BusinessMetrics": 353662726.399999,
  "SlotNetInfo": [
    {
      "NetInfoName": "eth0",
      "PublicIP": "9.223.110.232",
      "Current": 25376171.542857
    },
    {
      "NetInfoName": "eth1",
      "PublicIP": "9.223.96.218",
      "Current": 338834939.199999
    }
  ]
},
"AccessRegion": "MC",
"RequestId": "odgadasd-dasdasd-dadsada"
}
```

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
InternalServerError	内部错误。
InternalServerError.ControlRequestError	请求控制器发生错误。
InternalServerError.NetworkInfoRequestError	智研流量数据请求错误。
InvalidParameterValue	参数取值错误。

获取公钥

最近更新时间：2025-04-25 01:45:01

1. 接口描述

接口请求域名：mna.tencentcloudapi.com。

获取公钥用于验签

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：GetPublicKey。
Version	是	String	公共参数 ，本接口取值：2021-01-19。
Region	否	String	公共参数 ，本接口不需要传递此参数。

3. 输出参数

参数名称	类型	描述
PublicKey	String	非对称公钥 示例值："-----BEGIN RSA Public Key----- \nMIIBIjANBgkqhkiG9w0Bd7Fqz8X\lno+TronsIDAQAB\n-----END RSA Public 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 获取公钥

输入示例

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

{ }
```

输出示例

```
{
  "Response": {
    "PublicKey": "-----BEGIN RSA Public Key-----\nMIIB\n-----END RSA Public Key-----",
    "RequestId": "edd378f7-2511-4692-a029-5ca3d22c1884"
  }
}
```

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
InternalServerError	内部错误。

下载用量统计数据

最近更新时间：2025-04-25 01:45:01

1. 接口描述

接口请求域名：mna.tencentcloudapi.com。

在用量统计页面下载流量数据

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：GetStatisticData。
Version	是	String	公共参数 ，本接口取值：2021-01-19。
Region	否	String	公共参数 ，本接口不需要传递此参数。
DeviceId	是	String	设备ID。若不指定设备，可传"-1" 示例值："mna-xxx"
BeginTime	是	Integer	统计开始时间，单位：s 示例值：1661212800
EndTime	是	Integer	统计结束时间，单位：s 示例值：1661587200
TimeGranularity	是	Integer	聚合粒度： 1:按小时统计 2:按天统计 示例值：1
AccessRegion	否	String	接入区域。取值范围：['MC','AP','EU','AM'] MC=中国大陆 AP=亚太 EU=欧洲 AM=美洲。不填代表全量区域。 示例值：MC
GatewayType	否	Integer	网关类型。0：公有云网关；1：自有网关。不传默认为0。 示例值：0
DeviceList.N	否	Array of String	设备ID列表，最多10个设备，下载多个设备流量和时使用，此时DeviceId可传"-1" 示例值：["mna-test1","mna-test2"]
GroupId	否	String	设备分组ID，若不指定分组则不传，按分组下载数据时使用 示例值："group-test"

3. 输出参数

参数名称	类型	描述
FilePath	String	文件地址url 示例值: "http://xxxx"
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 获取指定id的设备流量用量统计文件

获取流量统计数据

输入示例

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

{
  "EndTime": 1659514692,
  "DeviceId": "mna-test1",
  "TimeGranularity": 1,
  "BeginTime": 1659513692,
  "AccessRegion": "MC",
  "GatewayType": 0
}
```

输出示例

```
{
  "Response": {
    "RequestId": "test-req",
    "FilePath": "http://geekyang-cos-1257943044.cos-internal.ap-guangzhou.tencentcos.cn/test/175527768616861696.xlsx"
  }
}
```

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
InternalServerError	内部错误。
InternalServerError.ControlRequestError	请求控制器发生错误。
InternalServerError.FileIOError	文件读写异常。
InternalServerError.MonitorDataRequestError	监控数据请求错误。
InternalServerError.NetworkInfoRequestError	智研流量数据请求错误。
InvalidParameter	参数错误。

获取厂商硬件设备列表

最近更新时间：2025-04-25 01:45:01

1. 接口描述

接口请求域名：mna.tencentcloudapi.com。

获取厂商硬件设备列表

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：GetVendorHardware。
Version	是	String	公共参数 ，本接口取值：2021-01-19。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
PageNumber	是	Integer	页码 示例值：1
PageSize	是	Integer	页面数量 示例值：10
Keyword	否	String	关键字 示例值：mna-xxx
Status	否	Integer	激活状态， 空：全部； 1:待激活； 2:已激活； 示例值：1

3. 输出参数

参数名称	类型	描述
VendorHardware	Array of VendorHardware	硬件信息列表
Length	Integer	设备总数 示例值：10
TotalPage	Integer	总页数 示例值：3

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 厂商查看设备

厂商查看设备

输入示例

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

{
  "PageNumber": 1,
  "PageSize": 1
}
```

输出示例

```
{
  "Response": {
    "Length": 1,
    "RequestId": "728a4c82-b31c-4b32-9cee-3ba1fdd1903c",
    "TotalPage": 2,
    "VendorHardware": [
      {
        "CreateTime": "1685591205",
        "Description": "",
        "DeviceId": "",
        "Payer": 0,
        "HardwareId": "cpe-5n9n3o63t2",
        "LicenseChargingMode": 2,
        "LicensePayMode": -1,
        "SN": "cloud-sn",
        "Status": 1
      }
    ]
  }
}
```

示例2 获取厂商设备列表

获取厂商设备列表

输入示例

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

```
{
  "PageNumber": 1,
  "PageSize": 10,
  "Keyword": "keywords",
  "Status": 0
}
```

输出示例

```
{
  "Response": {
    "VendorHardware": [
      {
        "HardwareId": "cpe-2ycsnymh2u",
        "SN": "cpe-sn-04",
        "CreateTime": "1685449404",
        "Status": 0,
        "Payer": 0,
        "ActiveTime": "1685449404",
        "Description": "this is descript",
        "DeviceId": "mna-dev1",
        "LicenseChargingMode": 0,
        "LicensePayMode": -1,
        "LastOnlineTime": "1685449404"
      }
    ],
    "Length": 1,
    "TotalPage": 1,
    "RequestId": "728a4c82-b31c-4b32-9cee-3ba1fdd1903c"
  }
}
```

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
InternalServerError	内部错误。
InternalServerError.ControlRequestError	请求控制器发生错误。
InvalidParameterValue	参数取值错误。
OperationDenied.VendorNotRegister	当前账号尚未注册为厂商。

更新设备

最近更新时间：2025-04-25 01:45:00

1. 接口描述

接口请求域名：mna.tencentcloudapi.com。

更新设备信息

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：UpdateDevice。
Version	是	String	公共参数 ，本接口取值：2021-01-19。
Region	否	String	公共参数 ，本接口不需要传递此参数。
DeviceId	是	String	设备id 示例值："mna-xxx"
DeviceName	否	String	设备名称 示例值："dev"
Remark	否	String	设备备注 示例值："remark"
UpdateNetInfo.N	否	Array of UpdateNetInfo	更新设备网络信息
FlowTrunc	否	Integer	设备无流量包处理方式，0: 按量付费，1: 截断加速 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

更新设备

输入示例

```
POST / HTTP/1.1
Host: mna.tencentcloudapi.com
Content-Type: application/json
X-TC-Action: UpdateDevice
```

<公共请求参数>

```
{
  "DeviceId": "mna-yesydf",
  "DeviceName": "name1",
  "Remark": "remarks",
  "UpdateNetInfo": [
    {
      "Type": 0,
      "DataEnable": true,
      "UploadLimit": 1,
      "DownloadLimit": 1,
      "NetInfoName": "netname"
    }
  ]
}
```

输出示例

```
{
  "Response": {
    "RequestId": "a400ac72-8f93-4340-a862-f74b56942703"
  }
}
```

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
InternalServerError	内部错误。
InternalServerError.ControlRequestError	请求控制器发生错误。
InternalServerError.DuplicateDeviceName	设备名已存在。
InvalidParameterValue	参数取值错误。

更新互通规则备注

最近更新时间：2025-04-25 01:44:59

1. 接口描述

接口请求域名：mna.tencentcloudapi.com。

更新互通规则备注

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：UpdateL3Conn。
Version	是	String	公共参数 ，本接口取值：2021-01-19。
Region	否	String	公共参数 ，此参数为可选参数。
L3ConnId	是	String	互通规则ID 示例值：l3conn-c0rkbqhig8
Description	否	String	互通规则备注 示例值："this is discript"

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 示例1

输入示例

```
POST / HTTP/1.1
Host: mna.tencentcloudapi.com
Content-Type: application/json
X-TC-Action: UpdateL3Conn
<公共请求参数>
```



```
{
  "L3ConnId": "l3conn-c0rkbqhig8",
  "Description": "this is descript"
}
```

输出示例

```
{
  "Response": {
    "RequestId": "7ab57398-d79f-4195-b0d4-3aef1e43b9c7"
  }
}
```

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
InternalServerError	内部错误。
InternalServerError.ControlRequestError	请求控制器发生错误。
InvalidParameterValue	参数取值错误。

更新分组信息

最近更新时间：2025-04-25 01:45:00

1. 接口描述

接口请求域名：mna.tencentcloudapi.com。

更新分组备注

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：UpdateGroup。
Version	是	String	公共参数 ，本接口取值：2021-01-19。
Region	否	String	公共参数 ，此参数为可选参数。
GroupId	是	String	分组ID 示例值："1a2b3c4d5e"
Description	否	String	分组备注 示例值："this is descript"

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 示例1

输入示例

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

```
{
  "GroupId": "cliGrp-xf8rboasbh",
  "Description": "this is descript"
}
```

输出示例

```
{
  "Response": {
    "RequestId": "bcc2c1d2-be38-4249-a50b-d9c487a36354"
  }
}
```

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
InternalServerError	内部错误。
InternalServerError.ControlRequestError	请求控制器发生错误。
InvalidParameterValue	参数取值错误。

更新互通规则CIDR

最近更新时间：2025-04-25 01:44:59

1. 接口描述

接口请求域名：mna.tencentcloudapi.com。

更新互通规则CIDR

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：UpdateL3Cidr。
Version	是	String	公共参数 ，本接口取值：2021-01-19。
Region	否	String	公共参数 ，此参数为可选参数。
L3ConnId	是	String	互通规则ID 示例值："l3conn-c0rkbqhig"
Cidr1	是	String	互通规则CIDR 示例值："192.168.0.0/28"
DeviceId1	否	String	互通设备ID 示例值：mna-f8v7e6o43g
DeviceId2	否	String	互通设备ID 示例值：mna-f8v7e6o432
Cidr2	否	String	互通规则CIDR 示例值："192.168.0.16/28"

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 示例1

输入示例

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

```
{
  "L3ConnId": "l3conn-c0rkbqhig",
  "DeviceId1": "mna-f8v7e6o432",
  "Cidr1": "192.168.0.16/28",
  "DeviceId2": "mna-f8v7e6o43g",
  "Cidr2": "192.168.0.16/28"
}
```

输出示例

```
{
  "Response": {
    "RequestId": "7ab57398-d79f-4195-b0d4-3aef1e43b9c7"
  }
}
```

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
InternalServerError	内部错误。
InternalServerError.ControlRequestError	请求控制器发生错误。
InvalidParameterValue	参数取值错误。

修改互通规则开关

最近更新时间：2025-04-25 01:44:59

1. 接口描述

接口请求域名：mna.tencentcloudapi.com。

更新互通规则开关

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：UpdateL3Switch。
Version	是	String	公共参数 ，本接口取值：2021-01-19。
Region	否	String	公共参数 ，此参数为可选参数。
L3ConnId	是	String	互通规则ID 示例值："id-1"
Enable	否	Boolean	互通规则开关 示例值：true

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 示例1

输入示例

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

```
{
  "L3ConnId": "id",
  "Enable": true
}
```

输出示例

```
{
  "Response": {
    "RequestId": "7fa2c35a-ecca-454b-9521-65e17601143e"
  }
}
```

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
InternalServerError	内部错误。
InternalServerError.ControlRequestError	请求控制器发生错误。
InvalidParameterValue	参数取值错误。

更新硬件信息

最近更新时间：2025-04-25 01:45:00

1. 接口描述

接口请求域名：mna.tencentcloudapi.com。

更新硬件信息

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：UpdateHardware。
Version	是	String	公共参数 ，本接口取值：2021-01-19。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
HardwareId	是	String	硬件ID 示例值：cpe-6zmf86knqu
SN	否	String	硬件序列号 示例值：adsqda
Description	否	String	设备备注 示例值：description

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 更新硬件信息

更新硬件信息

输入示例

```
POST / HTTP/1.1
Host: mna.tencentcloudapi.com
Content-Type: application/json
```

```
X-TC-Action: UpdateHardware
```

<公共请求参数>

```
{
  "HardwareId": "cpe-6zmf86knqu",
  "SN": "adsqda",
  "Description": "this is descript"
}
```

输出示例

```
{
  "Response": {
    "RequestId": "4ff707e0-c87a-6cff-5c65-93f21e212a81"
  }
}
```

示例2 示例

示例

输入示例

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

```
{
  "HardwareId": "cpe-53g4emz6zc",
  "SN": "dd"
}
```

输出示例

```
{
  "Response": {
    "RequestId": "004fa540-6023-46a1-8f92-bcf24d74e24f"
  }
}
```

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
InternalServerError	内部错误。
InternalServerError.ControlRequestError	请求控制器发生错误。
InvalidParameterValue	参数取值错误。
OperationDenied.DuplicateSN	SN已存在
OperationDenied.VendorNotRegister	当前账号尚未注册为厂商。

订购一次性授权License

最近更新时间：2025-04-25 01:45:00

1. 接口描述

接口请求域名：mna.tencentcloudapi.com。

购买一次性授权License

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：OrderPerLicense。
Version	是	String	公共参数 ，本接口取值：2021-01-19。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
DeviceId	是	String	购买永久授权License的设备ID，如果是厂商未激活设备采用HardwareId 示例值："mna-test1"
Type	是	Integer	设备类型，0: SDK，1: CPE，作为用户创建或激活设备时传0，作为厂商创建待激活设备时传1 示例值：0
RollBack	否	Boolean	购买失败后是否回滚（删除）设备，默认true，如果设备绑定了生效中的流量包则不能回滚。 示例值：true
AutoVoucher	否	Boolean	是否自动选择代金券，默认false。 有多张券时的选择策略：按照可支付订单全部金额的券，先到期的券，可抵扣金额最大的券，余额最小的券，现金券 这个优先级进行扣券，且最多只抵扣一张券。 示例值：false
VoucherIds.N	否	Array of String	指定代金券ID。自动选择代金券时此参数无效。目前只允许传入一张代金券。 注：若指定的代金券不符合订单抵扣条件，则正常支付，不扣券 示例值：["OLSKVPGO4KUIU8HFDTEST1"]

3. 输出参数

参数名称	类型	描述
ResourceId	String	一次性授权License的资源ID 示例值：56d11777-50f7-4c60-9c89-e7076c8529a9-0
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 示例1

输入示例

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

{
  "DeviceId": "mna-devid",
  "Type": 0,
  "AutoVoucher": true,
  "VoucherIds": [
    "v-1"
  ]
}
```

输出示例

```
{
  "Response": {
    "RequestId": "1206563f-f13f-4647-aaa8-37fa86954cc4-1",
    "ResourceId": "56d11777-50f7-4c60-9c89-e7076c8529a9-0"
  }
}
```

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 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.TransactionException	交易流程异常
InternalError	内部错误。
InvalidParameter	参数错误。
OperationDenied	操作被拒绝。
OperationDenied.DeviceNotFound	该设备未开启永久授权
OperationDenied.InsufficientBalance	余额不足
OperationDenied.NotAllowedToPay	无支付权限
OperationDenied.RepeatPurchase	重复购买
OperationDenied.UnauthorizedUser	未实名认证
UnauthorizedOperation	未授权操作。
UnauthorizedOperation.NoPermission	未验证服务权限

订购流量包

最近更新时间：2025-04-25 01:45:00

1. 接口描述

接口请求域名：mna.tencentcloudapi.com。

购买预付费流量包

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：OrderFlowPackage。
Version	是	String	公共参数 ，本接口取值：2021-01-19。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
PackageType	是	String	流量包规格类型。可取值如下： DEVICE_1_FLOW_20G、DEVICE_2_FLOW_50G、 DEVICE_3_FLOW_100G、 DEVICE_5_FLOW_500G，分别代表20G、50G、100G、500G档位的流量包。 档位也影响流量包可绑定的设备数量上限： 20G：最多绑定1个设备 50G：最多绑定2个设备 100G：最多绑定3个设备 500G：最多绑定5个设备 示例值：DEVICE_1_FLOW_20G
DeviceList.N	是	Array of String	流量包绑定的设备ID列表。捆绑设备个数上限取决于包的规格档位： 20G：最多绑定1个设备 50G：最多绑定2个设备 100G：最多绑定3个设备 500G：最多绑定5个设备 示例值：["mna-test1","mna-test2"]
AutoRenewFlag	是	Boolean	是否自动续费，该选项和流量截断冲突，只能开启一个 示例值：true
PackageRegion	是	Integer	区域标识，0：国内，1：国外 示例值：0
FlowTruncFlag	否	Boolean	是否开启流量截断功能，该选项和自动续费冲突 示例值：false
AutoVoucher	否	Boolean	是否自动选择代金券，默认false。 有多张券时的选择策略：按照可支付订单全部金额的券，先到期的券，可抵扣金额最大的券，余

参数名称	必选	类型	描述
			额最小的券，现金券 这个优先级进行扣券，且最多只抵扣一张券。 示例值：true
VoucherIds.N	否	Array of String	指定代金券ID。自动选择代金券时此参数无效。目前只允许传入一张代金券。 注：若指定的代金券不符合订单抵扣条件，则正常支付，不扣券 示例值：["OLSKVPGO4KUIU8HFDTEST1"]

3. 输出参数

参数名称	类型	描述
ResourceId	String	流量包的唯一资源ID 示例值：56d11777-50f7-4c60-9c89-e7076c8529a9-0
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 示例1

输入示例

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

{
  "PackageType": "DEVICE_1_FLOW_20G",
  "DeviceList": [
    "mna-test1",
    "mna-test2"
  ],
  "AutoRenewFlag": true,
  "PackageRegion": 1
}
```

输出示例

```
{
  "Response": {
    "RequestId": "1206563f-f13f-4647-aaa8-37fa86954cc4-1",
    "ResourceId": "56d11777-50f7-4c60-9c89-e7076c8529a9-0"
  }
}
```

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
FailedOperation.TransactionException	交易流程异常
InternalError	内部错误。
InvalidParameter	参数错误。
LimitExceeded	超过配额限制。
OperationDenied	操作被拒绝。
OperationDenied.InsufficientBalance	余额不足
OperationDenied.NotAllowedToPay	无支付权限
OperationDenied.UnauthorizedUser	未实名认证
ResourceInUse	资源被占用。
UnauthorizedOperation	未授权操作。
UnauthorizedOperation.NoPermission	未验证服务权限
UnauthorizedOperation.UnopenedLiveService	直播服务未开通

查询流量告警信息

最近更新时间：2025-04-25 01:45:03

1. 接口描述

接口请求域名：mna.tencentcloudapi.com。

根据AppId查询用户设置的流量告警信息，包括阈值，回调url和key

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：GetFlowAlarmInfo。
Version	是	String	公共参数 ，本接口取值：2021-01-19。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。

3. 输出参数

参数名称	类型	描述
AlarmValue	Integer	流量包的告警阈值 示例值：20
NotifyUrl	String	告警通知回调url 示例值：http://ip:port/callurl
CallbackKey	String	告警通知回调key 示例值：xd23y5cb89bx23432
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 示例1

输入示例

```
POST / HTTP/1.1
Host: mna.tencentcloudapi.com
Content-Type: application/json
X-TC-Action: GetFlowAlarmInfo
```

<公共请求参数>

{}

输出示例

```
{
  "Response": {
    "AlarmValue": 20,
    "NotifyUrl": "http://ip:port/callurl",
    "CallbackKey": "xd23y5cb89bx23432",
    "RequestId": "1206563f-f13f-4647-aaa8-37fa86954cc4-1"
  }
}
```

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
InternalServerError	内部错误。
InvalidParameter	参数错误。
OperationDenied.IllegalRequest	非法请求，可能是重放攻击、伪造攻击。

修改流量包自动续费标识

最近更新时间：2025-04-25 01:45:00

1. 接口描述

接口请求域名：mna.tencentcloudapi.com。

可开启/关闭流量包自动续费，不影响当前周期正在生效的流量包。

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：ModifyPackageRenewFlag。
Version	是	String	公共参数 ，本接口取值：2021-01-19。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
ResourceId	是	String	流量包的唯一资源ID 示例值：56d11777-50f7-4c60-9c89-e7076c8529a9-0
RenewFlag	是	Boolean	自动续费标识。true代表自动续费，false代表不自动续费 示例值：true

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 示例1

输入示例

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

{
  "ResourceId": "56d11777-50f7-4c60-9c89-e7076c8529a9-0",
```

```
"RenewFlag": false
}
```

输出示例

```
{
  "Response": {
    "RequestId": "1206563f-f13f-4647-aaa8-37fa86954cc4-1"
  }
}
```

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 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
InternalServerError	内部错误。
InvalidParameter	参数错误。
OperationDenied	操作被拒绝。
OperationDenied.IllegalRequest	非法请求，可能是重放攻击、伪造攻击。
OperationDenied.ModifiedOrRenewed	资源包已经变配或续费

错误码	描述
OperationDenied.TruncFlagOn	截断开关已经被开启
ResourceNotFound	资源不存在。
ResourceUnavailable	资源不可用。

设置用户流量告警信息

最近更新时间：2025-04-25 01:45:00

1. 接口描述

接口请求域名：mna.tencentcloudapi.com。

设置用户流量告警信息接口，通过该接口设置流量包告警阈值以及告警时回调的url和key

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：SetNotifyUrl。
Version	是	String	公共参数 ，本接口取值：2021-01-19。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
NotifyUrl	是	String	告警通知回调url 示例值：http://ip:port/callurl
CallbackKey	是	String	告警通知回调key 示例值：xd23y5cb89bx23432
AlarmValue	否	Integer	流量包的告警阈值 示例值：20

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 示例1

输入示例

```
POST / HTTP/1.1
Host: mna.tencentcloudapi.com
Content-Type: application/json
X-TC-Action: SetNotifyUrl
```


<公共请求参数>

```
{
  "AlarmValue": 20,
  "NotifyUrl": "http://ip:port/callurl",
  "CallbackKey": "xd23y5cb89bx23432"
}
```

输出示例

```
{
  "Response": {
    "RequestId": "1206563f-f13f-4647-aaa8-37fa86954cc4-1"
  }
}
```

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
InternalServerError	内部错误。
InvalidParameter	参数错误。

错误码	描述
OperationDenied	操作被拒绝。
OperationDenied.IllegalRequest	非法请求，可能是重放攻击、伪造攻击。

获取流量包列表

最近更新时间：2025-04-25 01:45:03

1. 接口描述

接口请求域名：mna.tencentcloudapi.com。

获取流量包列表

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：GetFlowPackages。
Version	是	String	公共参数 ，本接口取值：2021-01-19。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
PageNumber	是	Integer	页码，从1开始 示例值：1
PageSize	是	Integer	每页个数 示例值：10
ResourceId	否	String	流量包的唯一资源ID 示例值：live-jjc000eABvgr
DeviceId	否	String	流量包绑定的设备ID 示例值：mna-dev1
Status	否	Integer	流量包状态，0：未生效，1：有效期内，2：已过期 示例值：1

3. 输出参数

参数名称	类型	描述
PackageList	Array of FlowPackageInfo	流量包列表
Total	Integer	总数 示例值：12
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 示例

输入示例

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

{
  "PageNumber": 1,
  "PageSize": 1
}
```

输出示例

```
{
  "Response": {
    "PackageList": [
      {
        "ActiveTime": 1734613941,
        "AppId": 251198806,
        "CapacityRemain": 50000,
        "CapacityRemainPrecise": 50000,
        "CapacitySize": 50000,
        "CreateTime": 1734613920,
        "DeviceList": [
          "mna-6t5t54hcry",
          "mna-s72mu68gh3"
        ],
        "ExpireTime": 1737292340,
        "ModifyStatus": 0,
        "PackageType": "DEVICE_2_FLOW_50G",
        "RenewFlag": true,
        "ResourceId": "live-jjc000eABvgr_qt",
        "Status": 1,
        "TruncFlag": false
      }
    ],
    "RequestId": "e6972818-74b2-4228-a5ec-70c11e37fe6c",
    "Total": 506
  }
}
```

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 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。
InvalidParameter	参数错误。
OperationDenied	操作被拒绝。

活跃设备数量统计

最近更新时间：2025-04-25 01:45:04

1. 接口描述

接口请求域名：mna.tencentcloudapi.com。

活跃设备数量统计

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：GetActiveDeviceCount。
Version	是	String	公共参数 ，本接口取值：2021-01-19。
Region	否	String	公共参数 ，此参数为可选参数。
Period	否	Integer	查询粒度。0:day, 1:week, 2:month, 不传默认为day 示例值：0
StartTime	否	Integer	开始时间。单位秒 示例值：1732176361
EndTime	否	Integer	结束时间。单位秒 示例值：1732176361
DevGroup	否	String	设备组, 不传查询全部 示例值：group1
LicenseType	否	Integer	license类型, 不传查询全部, 1: 租户月付, 2: 厂商月付, 3: 永久授权 示例值：1

3. 输出参数

参数名称	类型	描述
ActiveDeviceList	Array of ActiveDeviceList	激活设备统计 示例值：[{ "count": 1, "time": "2024-11-20" }]
Period	Integer	查询粒度，0:day, 1:week, 2:month, 不传默认为day 示例值：1
StartTime	String	开始时间 示例值：2024-11-20

参数名称	类型	描述
EndTime	String	结束时间 示例值：2024-11-20
DevGroup	String	设备组 示例值：group1
LicenseType	String	license类型, 不传查询全部, 1: 租户月付, 2: 厂商月付, 3: 永久授权 示例值：1
Appld	String	租户ID 示例值：251198806
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 活跃设备数量统计

查询活跃设备数量统计

输入示例

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

{
  "Period": 0,
  "StartTime": 1732176361,
  "EndTime": 1732176362,
  "DevGroup": "comollit",
  "LicenseType": 3
}
```

输出示例

```
{
  "Response": {
    "ActiveDeviceList": [
      {
        "Count": 1,
        "Time": "2024-11-20"
      }
    ],
    "DevGroup": "group1",
    "EndTime": "2024-11-20",
    "LicenseType": "1",
    "Period": 0,
  }
}
```

```
"StartTime": "2024-11-20",
"RequestId": "3c140219-cfe9-470e-b241-907877d6fb03"
}
}
```

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。

下载活跃设备数量统计

最近更新时间：2025-04-25 01:45:04

1. 接口描述

接口请求域名：mna.tencentcloudapi.com。

下载活跃设备数量统计

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：DownloadActiveDeviceCount。
Version	是	String	公共参数 ，本接口取值：2021-01-19。
Region	否	String	公共参数 ，此参数为可选参数。
Period	否	Integer	查询粒度。0:day, 1:week, 2:month, 不传默认为day 示例值：0
StartTime	否	Integer	开始时间。单位秒 示例值：1732176361
EndTime	否	Integer	结束时间。单位秒 示例值：1732176361
DevGroup	否	String	设备组, 不传查询全部 示例值：group1
LicenseType	否	Integer	license类型, 不传查询全部, 1: 租户月付, 2: 厂商月付, 3: 永久授权 示例值：1

3. 输出参数

参数名称	类型	描述
FilePath	String	URL地址 示例值：http://cos.com
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 下载活跃设备数量统计

下载活跃设备数量统计

输入示例

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

{
  "Period": 0,
  "StartTime": 1732176361,
  "EndTime": 1732176362,
  "DevGroup": "comollit",
  "LicenseType": 3
}
```

输出示例

```
{
  "Response": {
    "FilePath": "http://cos.com",
    "RequestId": "3c140219-cfe9-470e-b241-907877d6fb03"
  }
}
```

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 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
InternalServerError	内部错误。
InvalidParameterValue	参数取值错误。

获取设备付费模式

最近更新时间：2025-04-25 01:45:04

1. 接口描述

接口请求域名：mna.tencentcloudapi.com。

获取设备付费模式

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：GetDevicePayMode。
Version	是	String	公共参数 ，本接口取值：2021-01-19。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
DeviceIdList.N	是	Array of String	设备ID列表 示例值：["mna-test1", "mna-test2", "mna-test3"]

3. 输出参数

参数名称	类型	描述
Result	Array of DevicePayModeInfo	结果信息
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 示例1

输入示例

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

```
{
  "DeviceIdList": [
    "mna-test1",
    "mna-test2",
    "mna-test3"
  ]
}
```

输出示例

```
{
  "Response": {
    "Result": [
      {
        "DeviceId": "mna-test1",
        "PayMode": 1,
        "PayModeDesc": "50G流量包",
        "ResourceId": "56d11777-50f7-4c60-9c89-e7076c8529a9-0"
      },
      {
        "DeviceId": "mna-test2",
        "PayMode": 1,
        "PayModeDesc": "20G流量包",
        "ResourceId": "56d11777-50f7-4c60-9c89-e7076c8529a9-0"
      },
      {
        "DeviceId": "mna-test3",
        "PayMode": 0,
        "PayModeDesc": "按流量后付费"
      }
    ],
    "RequestId": "1206563f-f13f-4647-aaa8-37fa86954cc4-1"
  }
}
```

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
InternalServerError	内部错误。
InvalidParameter	参数错误。
OperationDenied	操作被拒绝。

数据结构

最近更新时间：2025-04-08 01:20:58

ActivateHardware

激活设备

被如下接口引用：ActivateHardware。

名称	类型	必选	描述
Vendor	String	是	厂商名称 示例值：vendor1
SN	String	是	设备SN序列号 示例值：AN
DeviceName	String	是	设备名称 示例值：mna-w795bzezug
Description	String	否	备注 示例值：this is descript
DataKey	String	否	设备密钥 示例值：123drer43sr
AccessScope	Integer	否	接入环境。0：公有云网关；1：自有网关；2：公有云网关和自有网关。不填默认公有云网关。具体含义： 公有云网关：即该设备只能接入公有云网关（就近接入） 自有网关：即该设备只能接入已经注册上线的自有网关（就近接入或固定ip接入） 公有云网关和自有网关：即该设备同时可以接入公有云网关和已经注册上线的自有网关（就近接入或固定ip接入） 示例值：0
LicensePayMode	Integer	否	当付费方为租户时，可选择租户license付费方式： 0，月度授权 1，永久授权 若不传则默认为月度授权。 当付费方为厂商时，此参数无效 示例值：0
GroupId	String	否	设备分组ID 示例值：cliGrp-xf8rboasbh
GroupName	String	否	设备分组名称，预留参数，需要分组时传入GroupId 示例值：group-name
FlowTrunc	Integer	否	设备无流量包处理方式，0: 按量付费，1: 截断加速 示例值：0
DeviceId	String	否	激活后的设备ID 示例值：mna-w795bzezug

ActiveDeviceList

激活设备数统计

被如下接口引用：GetActiveDeviceCount。

名称	类型	描述
Count	Integer	数量 示例值：1
Time	String	时间 示例值：2024-11-20

DeviceBaseInfo

设备的基本信息

被如下接口引用：GetDevice, GetDevices, GetGroupDetail。

名称	类型	描述
DeviceId	String	设备唯一ID 示例值："mna-2x2tllhb18"
DeviceName	String	设备名称 示例值："mna-2x2tllhb18"
CreateTime	String	设备创建的时间，单位：ms 示例值：169043203113091072
LastTime	String	设备最后在线时间，单位：ms 示例值：169043203113091072
Remark	String	设备的备注 示例值："remark"
AccessScope	Integer	接入环境。0：公有云网关；1：自有网关；2：公有云网关和自有网关。默认公有云网关。具体含义：公有云网关：即该设备只能接入公有云网关（就近接入）自有网关：即该设备只能接入已经注册上线的自有网关（就近接入或固定ip接入）公有云网关和自有网关：即该设备同时可以接入公有云网关和已经注册上线的自有网关（就近接入或固定ip接入） 示例值：0
LicensePayMode	Integer	license授权有效期 0：月度授权 1：永久授权 示例值：1
Payer	Integer	付费方 0：厂商付费 1：客户付费 示例值：1
GroupId	String	设备分组ID 示例值："cliGrp-xf8rboasbh"
GroupName	String	设备分组名称 示例值："cliGrp-xf8rboasbh"
FlowTrunc	Integer	设备无流量包处理方式，0：按量付费，1：截断加速 示例值：0
Sn	String	设备sn 示例值："AN"
Vendor	String	厂商 示例值："vendor"

DeviceDetails

设备详细信息

被如下接口引用：GetDevice。

名称	类型	描述
DeviceBaseInfo	DeviceBaseInfo	设备基本信息
DeviceNetInfo	Array of DeviceNetInfo	设备网络信息
GatewaySite	String	聚合服务器地址 示例值：gz
BusinessDownRate	Float	业务下行速率 示例值：10
BusinessUpRate	Float	业务上行速率 示例值：12

DeviceNetInfo

设备网络状态信息

被如下接口引用：GetDevice。

名称	类型	描述
Type	Integer	网络类型： 0:数据 1:Wi-Fi 2:有线 示例值：1
DataEnable	Boolean	启用/禁用 示例值：true
UploadLimit	String	上行限速 示例值：10000
DownloadLimit	String	下行限速 示例值：10000
DataRx	Integer	接收实时速率 示例值：100
DataTx	Integer	发送实时速率 示例值：100
Vendor	Integer	运营商类型： 1: 中国移动； 2: 中国电信； 3: 中国联通 示例值：1
State	Integer	连接状态： 0:无连接 1:连接中 2:已连接 示例值：1
PublicIp	String	公网IP 示例值："xxx.xxx.xxx.xx"

名称	类型	描述
SignalStrength	Integer	信号强度/单位：dbm 示例值：-85
Rat	Integer	数据网络类型： -1：无效值 2：2G 3：3G 4：4G 5：5G 示例值：4
NetInfoName	String	网卡名 示例值："eth1"
DownRate	Float	下行实时速率（浮点数类型代替上一版本DataRx的整型） 示例值：12.3
UpRate	Float	上行实时速率（浮点数类型代替上一版本TxRate的整型） 示例值：1.23

DevicePayModelInfo

设备付费模式信息

被如下接口引用：GetDevicePayMode。

名称	类型	描述
DeviceId	String	设备ID 示例值：mna-dev1
PayMode	Integer	付费模式。 1：预付费流量包 0：按流量后付费 示例值：1
PayModeDesc	String	付费模式描述 示例值：50G流量包
ResourceId	String	流量包ID，仅当付费模式为流量包类型时才有。 示例值：res-1

FlowDetails

设备流量信息

被如下接口引用：GetMultiFlowStatistic。

名称	类型	描述
NetDetails	Array of NetDetails	流量数据点
DeviceId	String	设备ID 示例值："mna-xxx"
MaxValue	Float	流量最大值（单位：bytes） 示例值：51548

名称	类型	描述
AvgValue	Float	流量平均值（单位：bytes） 示例值：15441
TotalValue	Float	流量总值（单位：bytes） 示例值：656546

FlowPackageInfo

流量包信息

被如下接口引用：GetFlowPackages。

名称	类型	描述
ResourceId	String	流量包的唯一资源ID 示例值：resource-id
AppId	Integer	流量包所属的用户AppId 示例值：1009876765
PackageType	String	流量包规格类型。可取值如下： DEVICE_1_FLOW_20G、DEVICE_2_FLOW_50G、 DEVICE_3_FLOW_100G、 DEVICE_5_FLOW_500G，分别代表20G、50G、100G、500G档位的流量包。 档位也影响流量包可绑定的设备数量上限： 20G：最多绑定1个设备 50G：最多绑定2个设备 100G：最多绑定3个设备 500G：最多绑定5个设备 示例值：DEVICE_1_FLOW_20G
Status	Integer	流量包状态，0：未生效，1：有效期内，2：已过期 示例值：1
CreateTime	Integer	购买时间，Unix时间戳格式，单位：秒 示例值：1697181612
ActiveTime	Integer	生效时间，Unix时间戳格式，单位：秒 示例值：1697181612
ExpireTime	Integer	过期时间，Unix时间戳格式，单位：秒 示例值：1699860012
DeviceList	Array of String	流量包绑定的设备ID列表 示例值：["mna-test1","mna-test2"]
CapacitySize	Integer	流量包总容量，单位：MB 示例值：20000
CapacityRemain	Integer	流量包余量，单位：MB 示例值：10000
RenewFlag	Boolean	自动续费标识。true代表自动续费，false代表不自动续费 示例值：true
ModifyStatus	Integer	资源包变更状态，0：未发生变配；1：变配中；2：已变配或已续费 示例值：0
TruncFlag	Boolean	流量截断标识。true代表开启流量截断，false代表不开启流量截断 示例值：false

名称	类型	描述
CapacityRemainPrecise	Integer	流量包精确余量，单位：MB 示例值：200

GroupInfo

分组的基本信息

被如下接口引用：GetGroupDetail, GetGroupList。

名称	类型	必选	描述
GroupId	String	否	分组ID 示例值："1a2b3c4d5e"
GroupName	String	否	分组名 示例值："group-test"
CreateTime	String	否	分组创建的时间，单位：ms 示例值：169043203113091072
UpdateTime	String	否	分组更新的时间，单位：ms 示例值：169043203113091072
Description	String	否	分组描述 示例值："description for group"
DeviceNum	Integer	否	分组中的设备数量 示例值：5

Hardware

新建Hardware入参

被如下接口引用：AddHardware。

名称	类型	必选	描述
SN	String	是	硬件序列号 示例值：AN
LicenseChargingMode	Integer	是	license计费模式： 1，租户付费 2，厂商月付费 3，厂商永久授权 示例值：1
Description	String	否	设备描述 示例值：this is descript
HardwareId	String	否	硬件ID，入参无需传递 示例值：cpe-9oii2ew1z4

HardwareInfo

硬件信息

被如下接口引用：GetHardwareList。

名称	类型	描述
DeviceId	String	设备ID 示例值: mna-dev1
DeviceName	String	设备名称 示例值: dev-name
ActiveTime	String	激活时间 示例值: 1676800844
LastOnlineTime	String	最后在线时间 示例值: 1676800844
Description	String	备注 示例值: this is descript
VendorDescription	String	厂商备注 示例值: 厂商备注
LicenseChargingMode	Integer	license计费模式: 1, 租户月付费 2, 厂商月付费 3, license永久授权 注: 后续将废弃此参数, 新接入请使用LicensePayMode和Payer 示例值: 1
CreateTime	String	创建时间 示例值: 1676800844
SN	String	硬件序列号 示例值: AN
LicensePayMode	Integer	license授权有效期 0: 月度授权 1: 永久授权 示例值: 0
Payer	Integer	付费方 0: 客户付费 1: 厂商付费 示例值: 1
GroupId	String	设备分组ID 示例值: "group-1"
GroupName	String	设备分组名称 示例值: "g-name"
FlowTrunc	Integer	设备无流量包处理方式, 0: 按量付费, 1: 截断加速 示例值: 0

L3ConnInfo

三层互通规则信息

被如下接口引用: GetL3ConnList。

名称	类型	必选	描述
L3ConnId	String	否	互通规则ID 示例值: l3conn-abcd
DeviceId1	String	否	互通设备ID 示例值: mna-dex

名称	类型	必选	描述
Cidr1	String	否	互通规则CIDR 示例值：173.12.0.0/16
DeviceId2	String	否	互通设备ID 示例值：mna-cde
Cidr2	String	否	互通规则CIDR 示例值：173.13.0.0/16
Enable	Boolean	否	互通规则启用状态 示例值：true
Description	String	否	互通规则描述 示例值：this is descript

MonitorData

流量监控指标

被如下接口引用：GetNetMonitor。

名称	类型	描述
Time	String	时间点：s 示例值：1662622200
BusinessMetrics	Float	业务指标（bps/ms/%） 示例值：1.0
SlotNetInfo	Array of SlotNetInfo	网卡状态信息

NetDetails

网络详细信息

被如下接口引用：GetFlowStatistic, GetFlowStatisticByGroup, GetFlowStatisticByRegion, GetMultiFlowStatistic。

名称	类型	描述
Current	Float	流量值（byte） 示例值：98
Time	String	时间点，单位：s 示例值：1690424186

SlotNetInfo

网卡流量指标数据

被如下接口引用：GetNetMonitor。

名称	类型	描述
NetInfoName	String	网卡名 示例值："eth0"
PublicIP	String	公网IP 示例值："xxx.xxx.xx.xxx"

名称	类型	描述
Current	Float	指标数据（bps/ms/%） 示例值：2.3

UpdateNetInfo

更新设备网络状态信息

被如下接口引用：UpdateDevice。

名称	类型	必选	描述
Type	Integer	否	网络类型： 0:数据 1:Wi-Fi 示例值：1
DataEnable	Boolean	否	启用/禁用 示例值：true
UploadLimit	Integer	否	上行限速：bit 示例值：10000
DownloadLimit	Integer	否	下行限速：bit 示例值：10000
NetInfoName	String	否	网卡名 示例值："eth1"

VendorHardware

厂商硬件详细信息

被如下接口引用：GetVendorHardware。

名称	类型	描述
HardwareId	String	硬件id 示例值：cpe-ids1
SN	String	硬件序列号 示例值：AN
CreateTime	String	创建时间 示例值：1680019225
Status	Integer	激活状态，空：全部；1:待激活；2:已激活 示例值：1
ActiveTime	String	激活时间 示例值：1680019225
Description	String	厂商备注 示例值：this is descript
DeviceId	String	设备id 示例值：mna-dev1
LicenseChargingMode	Integer	license计费模式：1，租户月付费 2，厂商月付费 3，license永久授权 注：设备为租户付费且未激活（未选择月付还是永久付费）时，此参数返回1，仅代表租户付费。后续

名称	类型	描述
		将废弃此参数，新接入请使用LicensePayMode和Payer 示例值：1
LastOnlineTime	String	最后在线时间 示例值：1680019225
LicensePayMode	Integer	license授权有效期 0：月度授权 1：永久授权 -1：未知 示例值：0
Payer	Integer	付费方 0：客户付费 1：厂商付费 示例值：0

错误码

最近更新时间：2025-04-08 01:20:58

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 <code>Authorization</code> 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.TransactionException	交易流程异常
InternalError.ControlRequestError	请求控制器发生错误。
InternalError.DuplicateDataKey	设备密钥已存在。
InternalError.DuplicateDeviceName	设备名已存在。
InternalError.FileIOError	文件读写异常。
InternalError.MonitorDataRequestError	监控数据请求错误。

错误码	说明
InternalError.NetworkInfoRequestError	智研流量数据请求错误。
InternalError.UndefinedEncryptedKey	预置密钥尚未创建。
OperationDenied	操作被拒绝。
OperationDenied.DeviceNotFound	该设备未开启永久授权
OperationDenied.DuplicateSN	SN已存在
OperationDenied.HardwareHasActivated	输入SN对应的硬件已经激活
OperationDenied.HardwareNotExist	输入SN对应的硬件不存在。
OperationDenied.IllegalRequest	非法请求，可能是重放攻击、伪造攻击。
OperationDenied.InsufficientBalance	余额不足
OperationDenied.L3CidrOverLap	互通规则CIDR重叠
OperationDenied.L3ConnectionOverSize	互通规则数超过最大限制150条
OperationDenied.ModifiedOrRenewed	资源包已经变配或续费
OperationDenied.NotAllowedToPay	无支付权限
OperationDenied.RepeatPurchase	重复购买
OperationDenied.TruncFlagOn	截断开关已经被开启
OperationDenied.UnauthorizedUser	未实名认证
OperationDenied.VendorNotRegister	当前账号尚未注册为厂商。
UnauthorizedOperation.NoPermission	未验证服务权限
UnauthorizedOperation.UnopenedLiveService	直播服务未开通