

本地专用集群 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

专用集群相关接口

创建专用集群

创建专用集群订单

创建站点

删除专用集群

删除站点

查询专用集群内cos的容量信息

查询专用集群内宿主机的统计信息

查询专用集群宿主机信息

查询专用集群支持的实例规格列表

查询专用集群订单列表

查询专用集群概览信息

查询专用集群配置列表

查询专用集群列表

查询专用集群支持的可用区列表

查询站点列表

查询站点详情

修改本地专用集群信息

修改订单状态

修改机房设备信息

修改机房信息

查询本地专用集群云硬盘仓库信息

删除本地专用集群镜像缓存

创建本地专用集群云上镜像缓存

数据结构

错误码

API 文档

更新历史

最近更新时间：2025-06-05 01:14:52

第 13 次发布

发布时间：2025-06-05 01:14:47

本次发布包含了以下内容：

改善已有的文档。

修改数据结构：

- [DedicatedClusterOrder](#)
 - 废弃成员：DedicatedClusterTypeId, SupportedStorageType, SupportedUplinkSpeed, SupportedInstanceFamily
- [HostInfo](#)
 - 废弃成员：HostIp

第 12 次发布

发布时间：2025-04-24 01:10:00

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- [CreateDedicatedClusterImageCache](#)
- [DeleteDedicatedClusterImageCache](#)

第 11 次发布

发布时间：2025-03-17 01:10:06

本次发布包含了以下内容：

改善已有的文档。

修改数据结构：

- [DedicatedCluster](#)
 - 新增成员：RunningStatus

第 10 次发布

发布时间：2024-07-17 01:09:51

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- [DescribeDedicatedClusterOrders](#)
 - 新增入参：OrderTypes

修改数据结构：

- [DedicatedClusterOrder](#)
 - 新增成员：CheckStatus, DeliverExpectTime, DeliverFinishTime, CheckExpectTime, CheckFinishTime, OrderSLA, OrderPayPlan

第 9 次发布

发布时间：2024-03-12 01:14:14

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- [DescribeDedicatedClusterCbsStatistics](#)

新增数据结构：

- [SetInfo](#)

第 8 次发布

发布时间：2024-01-03 01:08:25

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- [DescribeDedicatedClusterOverview](#)
 - 新增出参：HostDetailInfo, HostStandbyCount, HostNormalCount

新增数据结构：

- [HostDetailInfo](#)

第 7 次发布

发布时间：2023-09-12 01:18:51

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- [DescribeDedicatedClusterHostStatistics](#)
 - 新增入参：HostId, StartTime, EndTime, Period

新增数据结构：

- [DetailData](#)

修改数据结构：

- [HostStatistic](#)
 - 新增成员：CpuAverage, MemAverage, NetAverage, CpuDetailData, MemDetailData, NetRateDetailData, NetPacketDetailData

第 6 次发布

发布时间：2023-02-24 01:07:16

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- [DescribeDedicatedClusterOverview](#)
 - 新增出参：VpnConnectionState, VpngwBandwidthData, LocalNetInfo, VpnConnectionBandwidthData

新增数据结构：

- [InBandwidth](#)
- [LocalNetInfo](#)
- [OutBandwidth](#)
- [VpngwBandwidthData](#)

第 5 次发布

发布时间：2023-02-22 01:10:12

本次发布包含了以下内容：

改善已有的文档。

修改数据结构：

- [HostInfo](#)
 - 新增成员：HostId

第 4 次发布

发布时间：2022-12-15 01:12:58

本次发布包含了以下内容：

改善已有的文档。

修改数据结构：

- [DedicatedClusterOrder](#)
 - 新增成员：OrderType

第 3 次发布

发布时间：2022-07-30 15:07:54

本次发布包含了以下内容：

改善已有的文档。

修改数据结构：

- [DedicatedClusterOrder](#)
 - 新增成员：PayType, TimeUnit, TimeSpan
- [DedicatedClusterOrderItem](#)
 - 新增成员：TypeFamily, SubOrderPayStatus

第 2 次发布

发布时间：2022-05-16 17:18:07

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- [CreateDedicatedClusterOrder](#)
- [CreateSite](#)
- [DeleteDedicatedClusters](#)
- [DeleteSites](#)
- [DescribeDedicatedClusterCosCapacity](#)
- [DescribeDedicatedClusterHostStatistics](#)
- [DescribeDedicatedClusterHosts](#)
- [DescribeDedicatedClusterInstanceTypes](#)
- [DescribeDedicatedClusterOrders](#)
- [DescribeDedicatedClusterOverview](#)
- [DescribeDedicatedClusterTypes](#)
- [DescribeDedicatedClusters](#)
- [DescribeDedicatedSupportedZones](#)
- [DescribeSites](#)
- [DescribeSitesDetail](#)
- [ModifyDedicatedClusterInfo](#)
- [ModifyOrderStatus](#)

- [ModifySiteDeviceInfo](#)
- [ModifySiteInfo](#)

新增数据结构：

- [CbsInfo](#)
- [CosCapacity](#)
- [CosInfo](#)
- [DedicatedCluster](#)
- [DedicatedClusterInstanceType](#)
- [DedicatedClusterOrder](#)
- [DedicatedClusterOrderItem](#)
- [DedicatedClusterType](#)
- [DedicatedClusterTypeInfo](#)
- [HostInfo](#)
- [HostStatistic](#)
- [RegionZoneInfo](#)
- [Site](#)
- [SiteDetail](#)
- [ZoneInfo](#)

第 1 次发布

发布时间：2022-05-16 14:37:22

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- [CreateDedicatedCluster](#)

简介

最近更新时间：2025-07-21 01:12:54

概述

本地专用集群（Cloud Dedicated Cluster，简称 CDC）是全新推出的基础设施类产品，目的是将中心化的公有云服务，延伸为可在客户机房落地的近场服务，融合公有云与本地 IDC 的双重优势，用户可以以本地化的时延和数据安全来使用公有云的丰富能力。

CDC 通过一体化机柜的形式在用户机房部署本地化算力，资源就近接入临近公有云地域实现统一管理，用户可通过公有云现有工具（控制台、API 等）来管理 CDC 资源。

本章节介绍的 CDC API 接口均为 API 3.0 接口。

您可以调用 API 对 CDC 进行操作，例如如创建、删除专用集群或站点等。

CDC 支持的所有接口信息，请参见 [API 概览](#)。

术语表

CDC API 接口的常见术语请参见下表：

术语	描述
专用集群	专用集群是腾讯云本地专用集群 CDC 提供服务的最小单元，一个专用集群对应一个 CDC 物理设备，用户可以为已通过勘测的场地购买专用集群。
站点	站点是用户用来放置腾讯云本地专用集群 CDC 物理设备的机房，需要满足 CDC 放置标准，并提供必要的环境和物理保障条件。
宿主机	宿主机指用户独享的物理服务器资源，基于宿主机可以创建专属的云服务器实例。宿主机为用户提供独享的物理服务器资源，满足资源独享、物理资源可见、安全合规等需求。

使用限制

- 需属于企业用户，已具备本地 IDC 机房，且支持第三方机柜置入。
- 对于具体的参数限制，请参考各接口文档中的参数说明。

API 快速入门

您可以使用 API Explorer 工具在线调用 API。

本文以创建专属集群为例，通过 API Explorer 工具调用 API 接口的步骤如下：

- 进入 [API Explorer](#) 工具页面。更多 API Explorer 工具使用信息，请参见 [使用 API Explorer](#)。
- 调用 [CreateSite](#) 接口，创建站点。
- 调用 [CreateDedicatedCluster](#) 接口，创建本地专用集群。

API 概览

最近更新時間：2025-04-24 01:10:06

专用集群相关接口

接口名称	接口功能	频率限制（次/秒）
CreateDedicatedCluster	创建专用集群	20
CreateDedicatedClusterOrder	创建专用集群订单	20
CreateSite	创建站点	20
DeleteDedicatedClusters	删除专用集群	20
DeleteSites	删除站点	20
DescribeDedicatedClusterCosCapacity	查询专用集群内cos的容量信息	20
DescribeDedicatedClusterHostStatistics	查询专用集群内宿主机的统计信息	20
DescribeDedicatedClusterHosts	查询专用集群宿主机信息	20
DescribeDedicatedClusterInstanceTypes	查询专用集群支持的实例规格列表	20
DescribeDedicatedClusterOrders	查询专用集群订单列表	20
DescribeDedicatedClusterOverview	查询专用集群概览信息	20
DescribeDedicatedClusterTypes	查询专用集群配置列表	20
DescribeDedicatedClusters	查询专用集群列表	20
DescribeDedicatedSupportedZones	查询专用集群支持的可用区列表	20
DescribeSites	查询站点列表	20
DescribeSitesDetail	查询站点详情	20
ModifyDedicatedClusterInfo	修改本地专用集群信息	20
ModifyOrderStatus	修改订单状态	20
ModifySiteDeviceInfo	修改机房设备信息	20
ModifySiteInfo	修改机房信息	20
CreateDedicatedClusterImageCache	创建本地专用集群云上镜像缓存	20
DeleteDedicatedClusterImageCache	删除本地专用集群镜像缓存	20
DescribeDedicatedClusterCbsStatistics	查询本地专用集群云硬盘仓库信息	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 01:10:55

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

注意：由于金融区和非金融区是隔离不通的，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）	cdc.ap-shanghai-fsi.tencentcloudapi.com
华南地区（深圳金融）	cdc.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：2025-03-28 01:15:09

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 cdc；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 。

参数名称	类型	必选	描述
Version	String	是	操作的 API 的版本。取值参考接口文档中入参公共参数 Version 的说明。例如云服务器的版本 2017-03-12。
SignatureMethod	String	否	签名方式，目前支持 HmacSHA256 和 HmacSHA1。只有指定此参数为 HmacSHA256 时，才使用 HmacSHA256 算法验证签名，其他情况均使用 HmacSHA1 验证签名。
Token	String	否	即 安全凭证服务 所颁发的临时安全凭证中的 Token，使用时需要将 SecretId 和 SecretKey 的值替换为临时安全凭证中的 TmpSecretId 和 TmpSecretKey。使用长期密钥时不能设置此 Token 字段。
Language	String	否	指定接口返回的语言，仅部分接口支持此参数。取值：zh-CN，en-US。zh-CN 返回中文，en-US 返回英文。

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

HTTP GET 请求结构示例：

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

Host: cvm.tencentcloudapi.com
```

HTTP POST 请求结构示例：

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

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

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

地域列表

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

地域	取值
亚太东南（曼谷）	ap-bangkok
华北地区（北京）	ap-beijing
西南地区（成都）	ap-chengdu
西南地区（重庆）	ap-chongqing
华南地区（广州）	ap-guangzhou
港澳台地区（中国香港）	ap-hongkong
亚太东南（雅加达）	ap-jakarta
华东地区（南京）	ap-nanjing
亚太东北（首尔）	ap-seoul
华东地区（上海）	ap-shanghai
华东地区（上海金融）	ap-shanghai-fsi
华南地区（深圳金融）	ap-shenzhen-fsi
亚太东南（新加坡）	ap-singapore

地域	取值
亚太东北（东京）	ap-tokyo
欧洲地区（法兰克福）	eu-frankfurt
美国东部（弗吉尼亚）	na-ashburn
美国西部（硅谷）	na-siliconvalley
南美地区（圣保罗）	sa-saopaulo

签名方法 v3

最近更新时间：2024-12-25 01:16:09

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

推荐使用 API Explorer

</> 点击调试

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

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

为什么要进行签名

签名通过以下方式帮助保护请求：

1. 验证请求者的身份
签名确保请求是由持有有效访问密钥的人发送的。请参阅控制台 [云 API 密钥](#) 页面获取密钥相关信息。
2. 保护传输中的数据
为了防止请求在传输过程中被篡改，腾讯云 API 会使用请求参数来计算请求的哈希值，并将生成的哈希值加密后作为请求的一部分，发送到腾讯云 API 服务器。服务器会使用收到的请求参数以同样的过程计算哈希值，并验证请求中的哈希值。如果请求被篡改，将导致哈希值不一致，腾讯云 API 将拒绝本次请求。

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

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

申请安全凭证

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

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

申请安全凭证的具体步骤如下：

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

签名版本 v3 签名过程

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

下面以云服务器查询广州区分例列表作为例子，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 *****。用户想查看广州区云服务器名为“未命名”的主机状态，只返回一条数据。则请求可能为：

```
curl -X POST https://cvm.tencentcloudapi.com \
-H "Authorization: TC3-HMAC-SHA256 Credential=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 中间号 (?) 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； 2. 多个头部，按照头部 key（小写）的 ASCII 升序进行拼接。 此示例计算结果是 content-type:application/json; charset=utf-8\nhost:cvm.tencentcloudapi.com\nx-tc-action:describeinstances\n。 注意：content-type 必须和实际发送的相符合，有些编程语言网络库即使未指定也会自动添加 charset 值，如果签名时和发送时不一致，服务器会返回签名校验失败。
SignedHeaders	参与签名的头部信息，说明此次请求有哪些头部参与了签名，和 CanonicalHeaders 包含的头部内容是一一对应的。content-type 和 host 为必选头部。 拼接规则： 1. 头部 key 统一转成小写； 2. 多个头部 key（小写）按照 ASCII 升序进行拼接，并且以分号 (;) 分隔。 此示例为 content-type;host;x-tc-action
HashedRequestPayload	请求正文（payload，即 body，此示例为 {"Limit": 1, "Filters": [{"Values": ["\u672a\u547d\u540d"], "Name": "instance-name"}]}）的哈希值，计算伪代码为 Lowercase(HexEncode(Hash.SHA256(RequestPayload)))，即对 HTTP 请求正文做 SHA256 哈希，然后十六进制编

字段名称	解释
	码，最后编码串转换成小写字母。对于 GET 请求，RequestPayload 固定为空字符串。此示例计算结果是 35e9c5b0e3ae67532d3c9f17ead6c90222632e5b1ff7f6e89887f1398934f064。

根据以上规则，示例中得到的规范请求串如下：

```
POST
/

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

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

2. 拼接待签名字符串

按如下格式拼接待签名字符串：

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

字段名称	解释
Algorithm	签名算法，目前固定为 TC3-HMAC-SHA256。
RequestTimestamp	请求时间戳，即请求头部的公共参数 X-TC-Timestamp 取值，取当前时间 UNIX 时间戳，精确到秒。此示例取值为 1551113065。
CredentialScope	凭证范围，格式为 Date/service/tc3_request，包含日期、所请求的服务和终止字符串（tc3_request）。Date 为 UTC 标准时间的日期，取值需要和公共参数 X-TC-Timestamp 换算的 UTC 标准时间日期一致；service 为产品名，必须与调用的产品域名一致。此示例计算结果是 2019-02-25/cvm/tc3_request。
HashedCanonicalRequest	前述步骤拼接所得规范请求串的哈希值，计算伪代码为 Lowercase(HexEncode(Hash.SHA256(CanonicalRequest)))。此示例计算结果是 7019a55be8395899b900fb5564e4200d984910f34794a27cb3fb7d10ff6a1e84。

注意：

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

根据以上规则，示例中得到的待签名字符串如下：

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

3. 计算签名

1) 计算派生签名密钥，伪代码如下：

```
SecretKey = "*****"  
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=content-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=10b1a37a7301a02ca19a647ad722d5e43b4b3cff309d421d85b46093f6ab6c4f
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.utcfromtimestamp(timestamp).strftime("%Y-%m-%d")
params = {"Limit": 1, "Filters": [{"Values": [u"未命名"], "Name": "instance-name"}]}

# ***** 步骤 1: 拼接规范请求串 *****

http_request_method = "POST"
canonical_uri = "/"
canonical_querystring = ""
```

```
ct = "application/json; charset=utf-8"
payload = json.dumps(params)
canonical_headers = "content-type:%s\nhost:%s\nx-tc-action:%s\n" % (ct, host, action.lower())
signed_headers = "content-type;host;x-tc-action"
hashed_request_payload = hashlib.sha256(payload.encode("utf-8")).hexdigest()
canonical_request = (http_request_method + "\n" +
canonical_uri + "\n" +
canonical_querystring + "\n" +
canonical_headers + "\n" +
signed_headers + "\n" +
hashed_request_payload)
print(canonical_request)

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

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

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

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

Golang

```
package main

import (
    "crypto/hmac"
```

```
"crypto/sha256"
"encoding/hex"
"fmt"
"os"
"strings"
"time"
)

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

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

func main() {
    // 需要设置环境变量 TENCENTCLOUD_SECRET_ID, 值为示例的 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"
.$credentialScope."\n"
.$hashedCanonicalRequest;
echo $stringToSign.PHP_EOL;

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

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

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

Ruby

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

```
require 'json'
require 'time'
require 'openssl'

# 密钥参数
# 需要设置环境变量 TENCENTCLOUD_SECRET_ID, 值为示例的 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}\nxc-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, 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++)
    {
        snprintf(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 OPENSSSL_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 OPENSSSL_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* 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, "\nX-TC-Action:");
char value[100] = {0};
lowercase(action, value);
strcat(canonical_headers, value);
strcat(canonical_headers, "\n");
const char* signed_headers = "content-type;host;x-tc-action";
const char* payload = "{\"Limit\": 1, \"Filters\": [{\"Values\": [\"\\u672a\\u547d\\u540d\"], \"Name\": \"instance-name\"}] }";
char hashed_request_payload[100] = {0};
sha256_hex(payload, hashed_request_payload);

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

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

// ***** 步骤 3: 计算签名 *****
char k_key[64] = {0};
sprintf(k_key, "%s", "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相符合，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

最近更新時間：2024-12-25 01:16:09

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
Region	实例所在区域	ap-guangzhou
InstanceId.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-gu
angzhou&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&R
egion=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 编码前需要先以 UTF-8 进行编码。

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

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

4. 签名失败

根据实际情况，存在以下签名失败的错误码，请根据实际情况处理。

错误代码	错误描述
AuthFailure.SignatureExpire	签名过期
AuthFailure.SecretIdNotFound	密钥不存在
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=7RAM2xfNMO9EiVTNmPg06MRnCvQ%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 01:14:19

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 06:18:52

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。

专用集群相关接口

创建专用集群

最近更新时间：2025-09-12 01:25:43

1. 接口描述

接口请求域名：`cdc.tencentcloudapi.com`。

创建专用集群

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：CreateDedicatedCluster。
Version	是	String	公共参数 ，本接口取值：2020-12-14。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
Siteld	是	String	专用集群所属的Siteld 示例值：site-98dj3kd
Name	是	String	专用集群的名称 示例值：my-site
Zone	是	String	专用集群所属的可用区 示例值：ap-guangzhou
Description	否	String	专用集群的描述 示例值：my dedicated cluster

3. 输出参数

参数名称	类型	描述
DedicatedClusterId	String	创建的专用集群id 示例值：cluster-gbo27yc4
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 创建专用集群

创建专用集群

输入示例

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

<公共请求参数>

```
{
  "SiteId": "site-98dj3kd",
  "Name": "my-site",
  "Zone": "ap-guangzhou",
  "Description": "site"
}
```

输出示例

```
{
  "Response": {
    "DedicatedClusterId": "cluster-gbo27yc4",
    "RequestId": "d39d6c09-44e9-4e80-8661-77b5ff3cbc15"
  }
}
```

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: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [CNB](#), [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.CdcBindVpcFail	CDC绑定VPC失败。
InvalidParameterValue.TooLong	参数名过长。
InvalidParameterValue.ZoneMismatchRegion	zone和region不匹配。
ResourceNotFound.InvalidSiteId	站点机房无效。

创建专用集群订单

最近更新时间：2025-09-12 01:25:42

1. 接口描述

接口请求域名：cdc.tencentcloudapi.com。

创建专用集群订单

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：CreateDedicatedClusterOrder。
Version	是	String	公共参数 ，本接口取值：2020-12-14。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
DedicatedClusterId	是	String	专用集群id 示例值：cluster-gbo27yc4
DedicatedClusterTypes.N	否	Array of DedicatedClusterTypeInfo	order关联的专用集群类型数组 示例值：[{ "Id": "cluster-gbo27yc4", "Count": 1 }]
CosInfo	否	CosInfo	order关联的cos存储信息 示例值：{ "Size": 1, "Type": C05 }
CbsInfo	否	CbsInfo	order关联的cbs存储信息 示例值：{ "Size": 1, "Type": SSD }
PurchaseSource	否	String	购买来源，默认为cloudApi 示例值：cloudApi
DedicatedClusterOrderId	否	String	当调用API接口提交订单时，需要提交DedicatedClusterOrderId，此处DedicatedClusterOrderId是之前创建的订单，可通过DescribeDedicatedClusterOrders接口查询，这里传入DedicatedClusterOrderId用于调整订单和支付。 示例值：ord-aijx4ets

3. 输出参数

参数名称	类型	描述
DedicatedClusterOrderId	String	专用集群订单id 示例值：ord-j3f3kdj4
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 创建专用集群订单

创建专用集群订单

输入示例

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

{
  "DedicatedClusterId": "cluster-gbo27yc4",
  "DedicatedClusterTypes": [
    {
      "Id": "cluster-gbo27yc4",
      "Count": 1
    }
  ],
  "CosInfo": {
    "Size": 1,
    "Type": "COS"
  },
  "CbsInfo": {
    "Size": 1,
    "Type": "SSD"
  },
  "PurchaseSource": "cloudApi",
  "DedicatedClusterOrderId": "ord-12345678"
}
```

输出示例

```
{
  "Response": {
    "DedicatedClusterOrderId": "ord-0yq4a8ia",
    "RequestId": "d39d6c09-44e9-4e80-8661-77b5ff3cbc15"
  }
}
```

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: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [CNB](#), [GitHub](#), [Gitee](#)

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 [公共错误码](#)。

错误码	描述
InvalidParameterValue	参数取值错误。
InvalidParameterValue.InvalidValueDedicatedClusterCosSize	本地专用集群COS大小不正确。
InvalidParameterValue.InvalidValueDedicatedClusterDataStepSize	CBS大小不正确，它必须是40的整数倍。
ResourceNotFound.InvalidDedicatedClusterId	该资源没有找到。
ResourceNotFound.InvalidDedicatedClusterOrderId	请确认资源ID 是否存在。
ResourceNotFound.InvalidDedicatedClusterTypeId	该资源没有找到。
UnsupportedOperation.NonCustomerAppldNotSupport	不支持非CUSTOMER类型的app id。

创建站点

最近更新时间：2025-09-12 01:25:41

1. 接口描述

接口请求域名：cdc.tencentcloudapi.com。

创建站点

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：CreateSite。
Version	是	String	公共参数 ，本接口取值：2020-12-14。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
Name	是	String	站点名称 示例值：my-site
Country	是	String	站点所在国家 示例值：China
Province	是	String	站点所在省份 示例值：Guangdong
City	是	String	站点所在城市 示例值：Shenzhen
AddressLine	是	String	站点所在地区的详细地址信息 示例值：Shennan Road 10000
Description	否	String	站点描述 示例值：this site is good
Note	否	String	注意事项 示例值：the floor is not very good
FiberType	否	String	您将使用光纤类型将CDC设备连接到网络。有单模和多模两种选项。取值范围："MM","SM" 示例值：MM
OpticalStandard	否	String	您将CDC连接到网络时采用的光学标准。此字段取决于上行链路速度、光纤类型和到上游设备的距离。 示例值：1000Base-SX
PowerConnectors	否	String	电源连接器类型 示例值：380VAC3P
PowerFeedDrop	否	String	从机架上方还是下方供电。取值范围：["UP","DOWN"] 示例值：UP
MaxWeight	否	Integer	最大承重(KG) 示例值：5000
PowerDrawKva	否	Integer	功耗(KW) 示例值：4000
UplinkSpeedGbps	否	Integer	网络到腾讯云Region区域的上行链路速度(Gbps) 示例值：10

参数名称	必选	类型	描述
UplinkCount	否	Integer	将CDC连接到网络时，每台CDC网络设备(每个机架 2 台设备)使用的上行链路数量。 示例值：2
ConditionRequirement	否	Boolean	是否满足下面环境条件： 1、场地没有材料要求或验收标准会影响 CDC 设备配送和安装。 2、确定的机架位置包含： 温度范围为 41 到 104° F (5 到 40° C)。 湿度范围为 10° F (−12° C)和 8% RH (相对湿度)到 70° F(21° C)和 80% RH。 机架位置的气流方向为从前向后，且应具有足够的 CFM (每分钟立方英尺)。CFM 必须是 CDC 配置的 kVA 功耗值的 145.8 倍。 示例值：True
DimensionRequirement	否	Boolean	是否满足下面的尺寸条件： 您的装货站台可以容纳一个机架箱(高 x 宽 x 深 = 94" x 54" x 48")。 您可以提供从机架(高 x 宽 x 深 = 80" x 24" x 48")交货地点到机架最终安置位置的明确通道。测量深度时，应包括站台、走廊通道、门、转弯、坡道、货梯，并将其他通道限制考虑在内。 在最终的 CDC 安置位置，前部间隙可以为 48" 或更大，后部间隙可以为 24" 或更大。 示例值：True
RedundantNetworking	否	Boolean	是否提供冗余的上游设备(交换机或路由器)，以便两台 网络设备都能连接到网络设备。 示例值：True
PostalCode	否	Integer	站点所在地区的邮编 示例值：518000
OptionalAddressLine	否	String	站点所在地区的详细地址信息（补充） 示例值：tengda
NeedHelp	否	Boolean	是否需要腾讯云团队协助完成机架支撑工作 示例值：True
RedundantPower	否	Boolean	是否电源冗余 示例值：True
BreakerRequirement	否	Boolean	上游断路器是否具备 示例值：True

3. 输出参数

参数名称	类型	描述
SiteId	String	创建Site生成的id 示例值：site-98dj3kd
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 创建站点

创建站点

输入示例

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

{
  "Name": "my-site",
  "Description": "firstsite",
```

```
"Note": "newsite",
"FiberType": "MM",
"OpticalStandard": "1000Base-SX",
"PowerConnectors": "380VAC3P",
"PowerFeedDrop": "UP",
"MaxWeight": 500,
"PowerDrawKva": 3000,
"UplinkSpeedGbps": 10,
"UplinkCount": 2,
"ConditionRequirement": "True",
"DimensionRequirement": "True",
"RedundantNetworking": "True",
"Country": "China",
"Province": "Guangdong",
"City": "Shenzhen",
"PostalCode": "51800",
"AddressLine": "nanshan",
"OptionalAddressLine": "tengda",
"BreakerRequirement": "True",
"RedundantPower": "True",
"NeedHelp": "True"
}
```

输出示例

```
{
  "Response": {
    "SiteId": "site-c738dj",
    "RequestId": "d39d6c09-44e9-4e80-8661-77b5ff3cbc16"
  }
}
```

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: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [CNB](#), [GitHub](#), [Gitee](#)

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 [公共错误码](#)。

错误码	描述
InvalidParameterValue.InvalidAppldFormat	无效的Appld
InvalidParameterValue.TooLong	参数名过长。

删除专用集群

最近更新时间：2025-09-12 01:25:40

-

1. 接口描述

接口请求域名：`cdc.tencentcloudapi.com`。

删除专用集群

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：DeleteDedicatedClusters。
Version	是	String	公共参数 ，本接口取值：2020-12-14。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
DedicatedClusterIds.N	是	Array of String	要删除的专用集群id 示例值：["cluster-xh37dh"]

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 删除专用集群

删除专用集群

输入示例

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

{
  "DedicatedClusterIds": [
    "cluster-dfaf323"
  ]
}
```

输出示例

```
{
  "Response": {
    "RequestId": "d39d6c09-44e9-4e80-8661-77b5ff3cbc15"
  }
}
```

```
}  
  
}
```

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: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [CNB](#), [GitHub](#), [Gitee](#)

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 [公共错误码](#)。

错误码	描述
InvalidParameterValue.LimitExceeded	超出大小限制。
ResourceNotFound.InvalidDedicatedClusterId	该资源没有找到。

删除站点

最近更新时间：2025-09-12 01:25:39

1. 接口描述

接口请求域名：cdc.tencentcloudapi.com。

删除站点

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：DeleteSites。
Version	是	String	公共参数 ，本接口取值：2020-12-14。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
SiteIds.N	是	Array of String	要删除的站点id列表 示例值：["site-98dj3kd"]

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 删除站点

删除站点

输入示例

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

{
  "SiteIds": [
    "site-98dj3kd"
  ]
}
```

输出示例

```
{
  "Response": {
    "RequestId": "d39d6c09-44e9-4e80-8661-77b5ff3cbc15"
  }
}
```

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: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [CNB](#), [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.FailDeleteSite	删除site失败。
InvalidParameterValue.LimitExceeded	超出大小限制。
ResourceNotFound.InvalidSiteId	站点机房无效。

查询专用集群内cos的容量信息

最近更新时间：2025-09-12 01:25:39

1. 接口描述

接口请求域名：cdc.tencentcloudapi.com。

查询专用集群内cos的容量信息

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：DescribeDedicatedClusterCosCapacity。
Version	是	String	公共参数 ，本接口取值：2020-12-14。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
DedicatedClusterId	是	String	查询的专用集群id 示例值：cluster-gbo27yc4

3. 输出参数

参数名称	类型	描述
CosCapacity	CosCapacity	本集群内cos容量信息，单位：‘GB’ 示例值：{ "TotalCapacity": 0, "TotalFreeCapacity": 0, "TotalUsedCapacity": 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 查询专用集群内cos的容量信息

查询专用集群内cos的容量信息

输入示例

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

{
  "DedicatedClusterId": "cluster-gbo27yc4"
}
```

输出示例

```
{
  "Response": {
    "CosCapacity": {
      "TotalCapacity": 3394.92,
      "TotalFreeCapacity": 3367.41,
      "TotalUsedCapacity": 27.51
    },
    "RequestId": "41aa48c2-477b-43c0-9d93-04ef3f529e5f"
  }
}
```

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: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [CNB](#), [GitHub](#), [Gitee](#)

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 [公共错误码](#)。

错误码	描述
InvalidParameterValue.InvalidAppldFormat	无效的Appld
ResourceNotFound.InvalidDedicatedClusterId	该资源没有找到。

查询专用集群内宿主机的统计信息

最近更新时间：2025-09-12 01:25:38

1. 接口描述

接口请求域名：cdc.tencentcloudapi.com。

查询专用集群内宿主机的统计信息

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：DescribeDedicatedClusterHostStatistics。
Version	是	String	公共参数 ，本接口取值：2020-12-14。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
DedicatedClusterId	是	String	查询的专用集群id 示例值：cluster-gbo27yc4
HostId	否	String	宿主机id 示例值：host-ro7czflb
StartTime	否	String	开始时间 示例值：2023-01-01 12:00:00
EndTime	否	String	结束时间 示例值：2023-01-02 12:00:00
Period	否	String	时间范围精度，1分钟(ONE_MINUTE)/5分钟(FIVE_MINUTE) 示例值：ONE_MINUTE

3. 输出参数

参数名称	类型	描述
HostStatisticSet	Array of HostStatistic	该集群内宿主机的统计信息列表
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 查询专用集群内宿主机的信息

查询专用集群内宿主机的信息

输入示例

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

<公共请求参数>

```
{
  "DedicatedClusterId": "cluster-gbo27yc4"
}
```

输出示例

```
{
  "Response": {
    "HostStatisticSet": [
      {
        "HostType": "S5.21XLARGE320",
        "HostFamily": "S5",
        "Cpu": 84,
        "Memory": 320,
        "Count": 240,
        "CpuAverage": 50,
        "MemAverage": 50,
        "NetAverage": 100,
        "CpuDetailData": {
          "Timestamps": [
            1693274400
          ],
          "Values": [
            0
          ],
        },
        "MemDetailData": {
          "Timestamps": [
            1693274400
          ],
          "Values": [
            0
          ],
        },
        "NetRateDetailData": {
          "Timestamps": [
            1693274400
          ],
          "Values": [
            0
          ],
        },
        "NetPacketDetailData": {
          "Timestamps": [
            1693274400
          ],
          "Values": [
            0
          ],
        }
      },
    ],
    "RequestId": "41aa48c2-477b-43c0-9d93-04ef3f529e5f"
  }
}
```

```
}  
  
}
```

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: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [CNB](#), [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
InvalidParameter.InstanceTypeNotSupport	该机型暂不支持。
InvalidParameterValue.InvalidAppldFormat	无效的Appld
ResourceNotFound.InvalidDedicatedClusterId	该资源没有找到。

查询专用集群宿主机信息

最近更新时间：2025-09-12 01:25:38

1. 接口描述

接口请求域名：cdc.tencentcloudapi.com。

查询专用集群宿主机信息

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：DescribeDedicatedClusterHosts。
Version	是	String	公共参数 ，本接口取值：2020-12-14。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
DedicatedClusterId	是	String	集群id 示例值：cluster-nk6e8j6c
Offset	否	Integer	偏移量，默认为0。 示例值：0
Limit	否	Integer	返回数量，默认为20 示例值：20

3. 输出参数

参数名称	类型	描述
HostInfoSet	Array of HostInfo	宿主机信息 示例值：[{"HostId":"host-0l8k7qm","HostIp":"50.4.2.2","ServiceType":"云服务器宿主机","HostStatus":"PENDING","HostType":"标准型S5.21XLARGE320","CpuAvailable":24,"CpuTotal":24,"MemAvailable":96,"MemTotal":96}]
TotalCount	Integer	宿主机总数 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 专用集群宿主机信息

专用集群宿主机信息

输入示例

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

<公共请求参数>

```
{
  "DedicatedClusterId": "cluster-nk6e8j6c"
}
```

输出示例

```
{
  "Response": {
    "HostInfoSet": [
      {
        "HostId": "host-018k7qm",
        "HostIp": "50.4.2.2",
        "ServiceType": "云服务器宿主机",
        "HostStatus": "PENDING",
        "HostType": "标准型S5.21XLARGE320",
        "RunTime": "2022-03-29 16:37:01",
        "ExpireTime": "2025-03-29 16:37:01",
        "CpuAvailable": 24,
        "CpuTotal": 24,
        "MemAvailable": 96,
        "MemTotal": 96
      },
      {
        "HostId": "host-018k7qp",
        "HostIp": "50.4.2.3",
        "ServiceType": "云服务器宿主机",
        "HostStatus": "PENDING",
        "HostType": "标准型S5.21XLARGE320",
        "RunTime": "2022-03-30 16:37:01",
        "ExpireTime": "2025-03-30 16:37:01",
        "CpuAvailable": 24,
        "CpuTotal": 24,
        "MemAvailable": 96,
        "MemTotal": 96
      }
    ],
    "TotalCount": 2,
    "RequestId": "7ffc723c-49d2-45ae-82cc-270aa35d3c59"
  }
}
```

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: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [CNB](#), [GitHub](#), [Gitee](#)

- Tencent Cloud SDK 3.0 for PHP: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [CNB](#), [GitHub](#), [Gitee](#)

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

该接口暂无业务逻辑相关的错误码，其他错误码详见 [公共错误码](#)。

查询专用集群支持的实例规格列表

最近更新时间：2025-09-12 01:25:37

1. 接口描述

接口请求域名：cdc.tencentcloudapi.com。

查询专用集群支持的实例规格列表

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：DescribeDedicatedClusterInstanceTypes。
Version	是	String	公共参数 ，本接口取值：2020-12-14。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
DedicatedClusterId	是	String	查询的专用集群id 示例值：cluster-gbo27yc4

3. 输出参数

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

4. 示例

示例1 查询专用集群支持的实例规格列表

查询专用集群支持的实例规格列表

输入示例

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

{
  "DedicatedClusterId": "cluster-gbo27yc4"
}
```

输出示例

```
{
  "Response": {
    "DedicatedClusterInstanceTypeSet": [
      {
        "Zone": "ap-guangzhou-2",
        "InstanceType": "S1.SMALL1",
        "NetworkCard": 0,
        "Cpu": 1,
        "Memory": 1,
        "InstanceFamily": "S1",
        "TypeName": "Standard S1",
        "StorageBlockAmount": 0,
        "InstanceBandwidth": 0,
        "InstancePps": 0,
        "CpuType": "intel",
        "Gpu": 0,
        "Fpga": 0,
        "Remark": "备注",
        "Status": "SOLD_OUT"
      },
      {
        "Zone": "ap-guangzhou-2",
        "InstanceType": "S3.SMALL1",
        "NetworkCard": 0,
        "Cpu": 1,
        "Memory": 1,
        "InstanceFamily": "S3",
        "TypeName": "Standard S3",
        "StorageBlockAmount": 0,
        "InstanceBandwidth": 0,
        "InstancePps": 0,
        "CpuType": "intel",
        "Gpu": 0,
        "Fpga": 0,
        "Remark": "备注",
        "Status": "SOLD_OUT"
      }
    ],
    "RequestId": "90ddf8cf-168b-4175-99f0-28858e90634a"
  }
}
```

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: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [CNB](#), [GitHub](#), [Gitee](#)

- Tencent Cloud SDK 3.0 for Go: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [CNB](#), [GitHub](#), [Gitee](#)

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 [公共错误码](#)。

错误码	描述
InvalidParameterValue.InvalidAppldFormat	无效的Appld
ResourceNotFound.InvalidDedicatedClusterId	该资源没有找到。

查询专用集群订单列表

最近更新时间：2025-09-12 01:25:36

1. 接口描述

接口请求域名：cdc.tencentcloudapi.com。

查询专用集群订单列表

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：DescribeDedicatedClusterOrders。
Version	是	String	公共参数 ，本接口取值：2020-12-14。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
DedicatedClusterIds.N	否	Array of String	按照专用集群id过滤 示例值：["cluster-xh37dh"]
DedicatedClusterOrderIds	否	String	按照专用集群订单id过滤 示例值：["ord-xjehslf"]
Offset	否	Integer	偏移量，默认为0。关于Offset的更进一步介绍请参考 API 简介 中的相关小节。 示例值：0
Limit	否	Integer	返回数量，默认为20，最大值为100。关于Limit的更进一步介绍请参考 API 简介 中的相关小节。 示例值：20
Status	否	String	订单状态为过滤条件：PENDING INCONSTRUCTION DELIVERING DELIVERED EXPIRED CANCELLED OFFLINE 示例值：DELIVERED
ActionType	否	String	订单类型为过滤条件：CREATE EXTEND 示例值：CREATE
OrderTypes.N	否	Array of String	订单类型列表 示例值：["PUSH"]

3. 输出参数

参数名称	类型	描述
DedicatedClusterOrderSet	Array of DedicatedClusterOrder	专用集群订单列表
TotalCount	Integer	符合条件的专用集群订单总数 示例值：20
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 查询专用集群订单列表

查询专用集群订单列表

输入示例

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

{
  "DedicatedClusterIds": [
    "cluster-262n63e8"
  ],
  "Offset": 0,
  "Limit": 1,
  "ActionType": "CREATE",
  "OrderTypes": [
    "PUSH"
  ]
}
```

输出示例

```
{
  "Response": {
    "DedicatedClusterOrderSet": [
      {
        "Action": "CREATE",
        "CheckExpectTime": "0000-00-00 00:00:00",
        "CheckFinishTime": "0000-00-00 00:00:00",
        "CheckStatus": "DELIVERING",
        "Cpu": 0,
        "CreateTime": "2024-09-22T00:00:00+00:00",
        "DedicatedClusterId": "cluster-262n63e8",
        "DedicatedClusterOrderId": "ord-rd0ty398",
        "DedicatedClusterOrderItems": [
          {
            "ComputeFormat": "1 SA5t.60XLARGE1076",
            "Count": 1,
            "CreateTime": "2024-09-22T00:00:00+00:00",
            "DedicatedClusterTypeId": "dctype-00100001",
            "Description": "基础底座",
            "Name": "基础底座",
            "PowerDraw": 0,
            "SubOrderId": "sord-e3t9jkm2",
            "SubOrderPayStatus": 0,
            "SubOrderStatus": "PENDING",
            "SupportedInstanceFamily": [],
            "SupportedStorageType": [],
            "SupportedUplinkSpeed": [
              10,
              25,
              40,
              100
            ]
          }
        ]
      }
    ]
  }
}
```

```
],
  "TotalCpu": 0,
  "TotalGpu": 0,
  "TotalMem": 0,
  "TypeFamily": "CUSTOM",
  "TypeName": "BASE_EQUIPMENT.BASE_FOUNDATION",
  "Weight": 0
},
{
  "DedicatedClusterTypeId": "None",
  "DeliverExpectTime": "0000-00-00 00:00:00",
  "DeliverFinishTime": "0000-00-00 00:00:00",
  "Gpu": 0,
  "Mem": 0,
  "OrderPayPlan": null,
  "OrderSLA": null,
  "OrderStatus": "PENDING",
  "OrderType": "PUSH",
  "PayStatus": 0,
  "PayType": "ONEOFF",
  "PowerDraw": 0,
  "SupportedInstanceFamily": [
    "None"
  ],
  "SupportedStorageType": [
    "None"
  ],
  "SupportedUplinkSpeed": [
    -1
  ],
  "TimeSpan": 3,
  "TimeUnit": "YEAR",
  "Weight": 0
},
{
  "RequestId": "3da146fc-0136-4386-b632-9dbef7f06431",
  "TotalCount": 30
}
}
```

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: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [CNB](#), [GitHub](#), [Gitee](#)

- Tencent Cloud SDK 3.0 for C++: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [CNB](#), [GitHub](#), [Gitee](#)

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 [公共错误码](#)。

错误码	描述
InvalidParameterValue	参数取值错误。
InvalidParameterValue.InvalidAppldFormat	无效的Appld

查询专用集群概览信息

最近更新时间：2025-09-12 01:25:36

1. 接口描述

接口请求域名：cdc.tencentcloudapi.com。

查询专用集群概览信息

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：DescribeDedicatedClusterOverview。
Version	是	String	公共参数 ，本接口取值：2020-12-14。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
DedicatedClusterId	是	String	集群id 示例值：cluster-nk6e8j6c

3. 输出参数

参数名称	类型	描述
CvmCount	Integer	云服务器数量 示例值：10
HostCount	Integer	宿主机数量 示例值：10
VpnConnectionState	String	vpn通道状态 示例值：AVAILABLE
VpngwBandwidthData	VpngwBandwidthData	vpn网关监控数据 示例值：{"InBandwidth": {"Timestamps": [1730620800], "Values": [57513800]}, "OutBandwidth": {"Timestamps": [1730620800], "Values": [36999115.2]}}
LocalNetInfo	LocalNetInfo	本地网关信息 示例值：{ "BGPRoute": "10.0.0.0/24", "LocalIp": "10.0.0.0", "Protocol": "BGP", "VpcId": "vpc-0eyxjlsz" }
VpnConnectionBandwidthData	Array of VpngwBandwidthData	vpn网关通道监控数据 示例值：{"InBandwidth": {"Timestamps": [1730620800], "Values": [57513800]}, "OutBandwidth": {"Timestamps": [1730620800], "Values": [36999115.2]}}
HostDetailInfo	Array of HostDetailInfo	宿主机资源概览信息 示例值：[{ "CpuAvailable": 646, "CpuTotal": 1008, "HostTypeFamily": "标准型S5", "MemAvailable": 54, "MemTotal": 960 },]
HostStandbyCount	Integer	热备宿主机数量 示例值：1
HostNormalCount	Integer	普通宿主机数量

参数名称	类型	描述
		示例值：1
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 专用集群概览信息

专用集群概览信息

输入示例

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

{
  "DedicatedClusterId": "cluster-nk6e8j6c"
}
```

输出示例

```
{
  "Response": {
    "CvmCount": 70,
    "HostCount": 15,
    "HostDetailInfo": [
      {
        "CpuAvailable": 646,
        "CpuTotal": 1008,
        "HostTypeFamily": "标准型S5",
        "MemAvailable": 54,
        "MemTotal": 960
      },
      {
        "CpuAvailable": 0,
        "CpuTotal": 384,
        "HostTypeFamily": "大数据型BMDK1",
        "MemAvailable": 0,
        "MemTotal": 1536
      },
      {
        "CpuAvailable": 144,
        "CpuTotal": 240,
        "HostTypeFamily": "标准型SK1",
        "MemAvailable": 672,
        "MemTotal": 864
      },
      {
        "CpuAvailable": 208,
        "CpuTotal": 540,
        "HostTypeFamily": "标准型SA2",
        "MemAvailable": 76,
```

```
"MemTotal": 896
},
{
  "CpuAvailable": 220,
  "CpuTotal": 464,
  "HostTypeFamily": "标准型SH2",
  "MemAvailable": 190,
  "MemTotal": 884
},
{
  "CpuAvailable": 102,
  "CpuTotal": 512,
  "HostTypeFamily": "标准型S6",
  "MemAvailable": 868,
  "MemTotal": 1880
}
],
"HostNormalCount": 15,
"HostStandbyCount": 0,
"LocalNetInfo": {
  "BGPRoute": "10.0.0.0/24",
  "LocalIp": "10.0.0.0",
  "Protocol": "BGP",
  "VpcId": "vpc-0eyxjlsz"
},
"RequestId": "52fb9236-337f-425e-8cab-7e00dbfa5073",
"VpnConnectionBandwidthData": [
  {
    "InBandwidth": {
      "Timestamps": [
        1730620800,
        1730624400,
        1730628000,
        1730631600,
        1730635200,
        1730638800,
        1730642400,
        1730646000,
        1730649600,
        1730653200,
        1730656800,
        1730660400,
        1730664000,
        1730667600,
        1730671200,
        1730674800,
        1730678400,
        1730682000,
        1730685600,
        1730689200,
        1730692800,
        1730696400,
        1730700000,
        1730703600,
        1730707200,
        1730710800,
        1730714400,
```

1730718000,
1730721600,
1730725200,
1730728800,
1730732400,
1730736000,
1730739600,
1730743200,
1730746800,
1730750400,
1730754000,
1730757600,
1730761200,
1730764800,
1730768400,
1730772000,
1730775600,
1730779200,
1730782800,
1730786400,
1730790000,
1730793600,
1730797200,
1730800800,
1730804400,
1730808000,
1730811600,
1730815200,
1730818800,
1730822400,
1730826000,
1730829600,
1730833200,
1730836800,
1730840400,
1730844000,
1730847600,
1730851200,
1730854800,
1730858400,
1730862000,
1730865600,
1730869200,
1730872800,
1730876400,
1730880000,
1730883600,
1730887200,
1730890800,
1730894400,
1730898000,
1730901600,
1730905200,
1730908800,
1730912400,
1730916000,
1730919600,

1730923200,
1730926800,
1730930400,
1730934000,
1730937600,
1730941200,
1730944800,
1730948400,
1730952000,
1730955600,
1730959200,
1730962800,
1730966400,
1730970000,
1730973600,
1730977200,
1730980800,
1730984400,
1730988000,
1730991600,
1730995200,
1730998800,
1731002400,
1731006000,
1731009600,
1731013200,
1731016800,
1731020400,
1731024000,
1731027600,
1731031200,
1731034800,
1731038400,
1731042000,
1731045600,
1731049200,
1731052800,
1731056400,
1731060000,
1731063600,
1731067200,
1731070800,
1731074400,
1731078000,
1731081600,
1731085200,
1731088800,
1731092400,
1731096000,
1731099600,
1731103200,
1731106800,
1731110400,
1731114000,
1731117600,
1731121200,
1731124800,

```

1731128400,
1731132000,
1731135600,
1731139200,
1731142800,
1731146400,
1731150000,
1731153600,
1731157200,
1731160800,
1731164400,
1731168000,
1731171600,
1731175200,
1731178800,
1731182400,
1731186000,
1731189600,
1731193200,
1731196800,
1731200400,
1731204000,
1731207600,
1731211200,
1731214800,
1731218400,
1731222000,
1731225600
],
"Values": [
3,
3,
3,
3,
3,
2,
3,
2,
3,
3,
3,
2,
2,
3,
3,
2,
2,
3,
3,
3,
3,
3,
2,
2,
3,
3,
3,
3,
3,
3,
3,
2,
2,
3,
3,
3,
3,
2,

```


3,
3,
3,
2,
3,
3,
3,
3,
3,
2,
2,
2,
3,
3,
2,
3,
3,
2,
2,
2,
3,
2,
2,
2,
2,
3,
2,
2,
2,
2,
3,
2,
2,
3,
2,
3,
3,
3,
3,
3,
3,
3,
3,
2,
3,
2,
3,
3,
3,
3,
3,
3,
3,
3,
3,
3,
3,
3,
2,
2,

```
2,  
3,  
3,  
3,  
3,  
3,  
3,  
3,  
3,  
3,  
2,  
3,  
3,  
3,  
2,  
3,  
3,  
3,  
2,  
3,  
2,  
2,  
2,  
2,  
3,  
2,  
2,  
2,  
3,  
2,  
2,  
2,  
3,  
2,  
0  
]  
},  
"OutBandwidth": {  
"Timestamps": [  
1730620800,  
1730624400,  
1730628000,  
1730631600,  
1730635200,  
1730638800,  
1730642400,  
1730646000,  
1730649600,  
1730653200,  
1730656800,  
1730660400,  
1730664000,  
1730667600,  
1730671200,  
1730674800,  
1730678400,  
1730682000,  
1730685600,  
1730689200,  
1730692800,  
1730696400,  
1730700000,  
1730703600,  
1730707200,
```

1730710800,
1730714400,
1730718000,
1730721600,
1730725200,
1730728800,
1730732400,
1730736000,
1730739600,
1730743200,
1730746800,
1730750400,
1730754000,
1730757600,
1730761200,
1730764800,
1730768400,
1730772000,
1730775600,
1730779200,
1730782800,
1730786400,
1730790000,
1730793600,
1730797200,
1730800800,
1730804400,
1730808000,
1730811600,
1730815200,
1730818800,
1730822400,
1730826000,
1730829600,
1730833200,
1730836800,
1730840400,
1730844000,
1730847600,
1730851200,
1730854800,
1730858400,
1730862000,
1730865600,
1730869200,
1730872800,
1730876400,
1730880000,
1730883600,
1730887200,
1730890800,
1730894400,
1730898000,
1730901600,
1730905200,
1730908800,
1730912400,

1730916000,
1730919600,
1730923200,
1730926800,
1730930400,
1730934000,
1730937600,
1730941200,
1730944800,
1730948400,
1730952000,
1730955600,
1730959200,
1730962800,
1730966400,
1730970000,
1730973600,
1730977200,
1730980800,
1730984400,
1730988000,
1730991600,
1730995200,
1730998800,
1731002400,
1731006000,
1731009600,
1731013200,
1731016800,
1731020400,
1731024000,
1731027600,
1731031200,
1731034800,
1731038400,
1731042000,
1731045600,
1731049200,
1731052800,
1731056400,
1731060000,
1731063600,
1731067200,
1731070800,
1731074400,
1731078000,
1731081600,
1731085200,
1731088800,
1731092400,
1731096000,
1731099600,
1731103200,
1731106800,
1731110400,
1731114000,
1731117600,

1730696400,
1730700000,
1730703600,
1730707200,
1730710800,
1730714400,
1730718000,
1730721600,
1730725200,
1730728800,
1730732400,
1730736000,
1730739600,
1730743200,
1730746800,
1730750400,
1730754000,
1730757600,
1730761200,
1730764800,
1730768400,
1730772000,
1730775600,
1730779200,
1730782800,
1730786400,
1730790000,
1730793600,
1730797200,
1730800800,
1730804400,
1730808000,
1730811600,
1730815200,
1730818800,
1730822400,
1730826000,
1730829600,
1730833200,
1730836800,
1730840400,
1730844000,
1730847600,
1730851200,
1730854800,
1730858400,
1730862000,
1730865600,
1730869200,
1730872800,
1730876400,
1730880000,
1730883600,
1730887200,
1730890800,
1730894400,
1730898000,

1730901600,
1730905200,
1730908800,
1730912400,
1730916000,
1730919600,
1730923200,
1730926800,
1730930400,
1730934000,
1730937600,
1730941200,
1730944800,
1730948400,
1730952000,
1730955600,
1730959200,
1730962800,
1730966400,
1730970000,
1730973600,
1730977200,
1730980800,
1730984400,
1730988000,
1730991600,
1730995200,
1730998800,
1731002400,
1731006000,
1731009600,
1731013200,
1731016800,
1731020400,
1731024000,
1731027600,
1731031200,
1731034800,
1731038400,
1731042000,
1731045600,
1731049200,
1731052800,
1731056400,
1731060000,
1731063600,
1731067200,
1731070800,
1731074400,
1731078000,
1731081600,
1731085200,
1731088800,
1731092400,
1731096000,
1731099600,
1731103200,

```
1731106800,  
1731110400,  
1731114000,  
1731117600,  
1731121200,  
1731124800,  
1731128400,  
1731132000,  
1731135600,  
1731139200,  
1731142800,  
1731146400,  
1731150000,  
1731153600,  
1731157200,  
1731160800,  
1731164400,  
1731168000,  
1731171600,  
1731175200,  
1731178800,  
1731182400,  
1731186000,  
1731189600,  
1731193200,  
1731196800,  
1731200400,  
1731204000,  
1731207600,  
1731211200,  
1731214800,  
1731218400,  
1731222000,  
1731225600  
],  
"Values": [  
27902,  
27212,  
34068,  
35574,  
34763,  
35373,  
29562,  
35340,  
35225,  
34355,  
35791,  
35595,  
34760,  
34317,  
35281,  
34520,  
35289,  
34290,  
34390,  
38823,  
35067,
```

35140,
42503,
36539,
34158,
34456,
39232,
35156,
249015,
38536,
35572,
31001,
35056,
35384,
35803,
35833,
35740,
34728,
36121,
34128,
34936,
35885,
35299,
34788,
34992,
35018,
29729,
53373,
32819,
35469,
27599,
35763,
35607,
34508,
34389,
35306,
34701,
35156,
34824,
34725,
26990,
34928,
34544,
34794,
33903,
34865,
35257,
127611,
32624,
34674,
27960,
34319,
35358,
34461,
36056,
34992,
35969,
34758,

34210,
36837,
35938,
33063,
33124,
34617,
34936,
38358,
35416,
35152,
33708,
33658,
35879,
33824,
36567,
43210,
43636,
46429,
35114,
35928,
39920,
34467,
34886,
34723,
39131,
31112,
34796,
37624,
33059,
39443,
37853,
33829,
34404,
36205,
33834,
36399,
41434,
39064,
38236,
33369,
34415,
34783,
40968,
34573,
36284,
33947,
34553,
35586,
34397,
34849,
33322,
37771,
36288,
35050,
29110,
35064,
36701,

```
34830,
28867,
34549,
34136,
35893,
36196,
34852,
34716,
34749,
34806,
34987,
35181,
35592,
35498,
34400,
34141,
38190,
37542,
34853,
40032,
38876,
41314,
36059,
35347,
25510,
34786,
38482,
34550,
35296,
37584,
38982,
35284,
34912,
0
]
},
"OutBandwidth": {
  "Timestamps": [
    1730620800,
    1730624400,
    1730628000,
    1730631600,
    1730635200,
    1730638800,
    1730642400,
    1730646000,
    1730649600,
    1730653200,
    1730656800,
    1730660400,
    1730664000,
    1730667600,
    1730671200,
    1730674800,
    1730678400,
    1730682000,
    1730685600,
```

1730689200,
1730692800,
1730696400,
1730700000,
1730703600,
1730707200,
1730710800,
1730714400,
1730718000,
1730721600,
1730725200,
1730728800,
1730732400,
1730736000,
1730739600,
1730743200,
1730746800,
1730750400,
1730754000,
1730757600,
1730761200,
1730764800,
1730768400,
1730772000,
1730775600,
1730779200,
1730782800,
1730786400,
1730790000,
1730793600,
1730797200,
1730800800,
1730804400,
1730808000,
1730811600,
1730815200,
1730818800,
1730822400,
1730826000,
1730829600,
1730833200,
1730836800,
1730840400,
1730844000,
1730847600,
1730851200,
1730854800,
1730858400,
1730862000,
1730865600,
1730869200,
1730872800,
1730876400,
1730880000,
1730883600,
1730887200,
1730890800,

1730894400,
1730898000,
1730901600,
1730905200,
1730908800,
1730912400,
1730916000,
1730919600,
1730923200,
1730926800,
1730930400,
1730934000,
1730937600,
1730941200,
1730944800,
1730948400,
1730952000,
1730955600,
1730959200,
1730962800,
1730966400,
1730970000,
1730973600,
1730977200,
1730980800,
1730984400,
1730988000,
1730991600,
1730995200,
1730998800,
1731002400,
1731006000,
1731009600,
1731013200,
1731016800,
1731020400,
1731024000,
1731027600,
1731031200,
1731034800,
1731038400,
1731042000,
1731045600,
1731049200,
1731052800,
1731056400,
1731060000,
1731063600,
1731067200,
1731070800,
1731074400,
1731078000,
1731081600,
1731085200,
1731088800,
1731092400,
1731096000,

```
1731099600,  
1731103200,  
1731106800,  
1731110400,  
1731114000,  
1731117600,  
1731121200,  
1731124800,  
1731128400,  
1731132000,  
1731135600,  
1731139200,  
1731142800,  
1731146400,  
1731150000,  
1731153600,  
1731157200,  
1731160800,  
1731164400,  
1731168000,  
1731171600,  
1731175200,  
1731178800,  
1731182400,  
1731186000,  
1731189600,  
1731193200,  
1731196800,  
1731200400,  
1731204000,  
1731207600,  
1731211200,  
1731214800,  
1731218400,  
1731222000,  
1731225600  
],  
"Values": [  
14743,  
13036,  
13841,  
11986,  
14205,  
12232,  
14241,  
11255,  
32616,  
32911,  
14676,  
11300,  
13258,  
14360,  
10414,  
11934,  
12258,  
11524,  
46844,
```

618757,
441117,
14550,
621280,
587274,
16082,
285697,
566534,
35427,
17471,
249220,
16333,
43087,
16624,
12179,
21936,
13806,
18801,
13401,
12249,
10875,
11303,
11201,
263156,
73029,
18724,
264555,
449618,
686591,
47155,
15872,
11577,
176658,
13757,
266241,
12524,
12406,
12678,
13477,
15617,
19549,
15652,
16489,
16828,
12518,
12430,
13571,
27827,
14885,
14349,
13194,
11612,
264983,
260379,
286792,
260640,
259787,

263219,
261059,
437908,
271498,
264451,
257998,
259888,
260100,
266845,
262762,
263858,
263620,
262946,
260613,
263291,
280029,
301617,
276933,
360990,
392133,
389615,
349783,
398237,
506978,
409018,
262950,
269338,
258466,
265593,
260187,
262383,
262769,
266388,
266964,
268590,
265538,
264164,
317219,
496280,
604524,
261015,
258793,
263620,
272089,
629272,
289891,
261537,
259547,
262861,
266082,
263891,
263102,
261475,
263224,
261281,
16596,
17447,

```
13329,
12544,
11800,
11304,
12536,
10488,
12197,
17483,
11364,
11322,
257165,
260621,
260356,
260428,
260607,
260234,
262239,
260919,
259631,
258579,
259974,
260478,
268614,
263499,
260490,
261597,
260948,
261713,
262219,
261705,
254206,
260921,
266181,
262548,
262275,
0
]
}
}
],
"VpnConnectionState": "AVAILABLE",
"VpngwBandwidthData": {
  "InBandwidth": {
    "Timestamps": [
      1730620800,
      1730624400,
      1730628000,
      1730631600,
      1730635200,
      1730638800,
      1730642400,
      1730646000,
      1730649600,
      1730653200,
      1730656800,
      1730660400,
      1730664000,
```

1730667600,
1730671200,
1730674800,
1730678400,
1730682000,
1730685600,
1730689200,
1730692800,
1730696400,
1730700000,
1730703600,
1730707200,
1730710800,
1730714400,
1730718000,
1730721600,
1730725200,
1730728800,
1730732400,
1730736000,
1730739600,
1730743200,
1730746800,
1730750400,
1730754000,
1730757600,
1730761200,
1730764800,
1730768400,
1730772000,
1730775600,
1730779200,
1730782800,
1730786400,
1730790000,
1730793600,
1730797200,
1730800800,
1730804400,
1730808000,
1730811600,
1730815200,
1730818800,
1730822400,
1730826000,
1730829600,
1730833200,
1730836800,
1730840400,
1730844000,
1730847600,
1730851200,
1730854800,
1730858400,
1730862000,
1730865600,
1730869200,

1730872800,
1730876400,
1730880000,
1730883600,
1730887200,
1730890800,
1730894400,
1730898000,
1730901600,
1730905200,
1730908800,
1730912400,
1730916000,
1730919600,
1730923200,
1730926800,
1730930400,
1730934000,
1730937600,
1730941200,
1730944800,
1730948400,
1730952000,
1730955600,
1730959200,
1730962800,
1730966400,
1730970000,
1730973600,
1730977200,
1730980800,
1730984400,
1730988000,
1730991600,
1730995200,
1730998800,
1731002400,
1731006000,
1731009600,
1731013200,
1731016800,
1731020400,
1731024000,
1731027600,
1731031200,
1731034800,
1731038400,
1731042000,
1731045600,
1731049200,
1731052800,
1731056400,
1731060000,
1731063600,
1731067200,
1731070800,
1731074400,

```
1731078000,  
1731081600,  
1731085200,  
1731088800,  
1731092400,  
1731096000,  
1731099600,  
1731103200,  
1731106800,  
1731110400,  
1731114000,  
1731117600,  
1731121200,  
1731124800,  
1731128400,  
1731132000,  
1731135600,  
1731139200,  
1731142800,  
1731146400,  
1731150000,  
1731153600,  
1731157200,  
1731160800,  
1731164400,  
1731168000,  
1731171600,  
1731175200,  
1731178800,  
1731182400,  
1731186000,  
1731189600,  
1731193200,  
1731196800,  
1731200400,  
1731204000,  
1731207600,  
1731211200,  
1731214800,  
1731218400,  
1731222000,  
1731225600  
],  
"Values": [  
57513800,  
68094611.2,  
79186521.6,  
60534504,  
78897427.2,  
63237265.6,  
79178713.6,  
60393168,  
81129268,  
51220798.4,  
81060523.2,  
52434900.8,  
78564217.6,
```

59975670.4,
81873972.8,
80544771.2,
81795468.8,
65186712,
80439569.6,
70954862.4,
79333094.4,
77677355.2,
91380812.8,
56897305.6,
79889120,
49745224,
59596936,
64410828,
309474268.8,
63769486.4,
82992758.4,
65433073.6,
58786961.6,
82539596.8,
64908304,
80435241.6,
70007491.2,
57498075.2,
58827907.2,
66269364.8,
67107579.2,
80139716.8,
66853212.8,
82855864,
65938686.4,
57210636,
78715593.6,
95870603.2,
53011934.4,
81546336,
81019800,
80505835.2,
60604449.6,
81215440,
78220001.6,
82425334.4,
56864844.8,
81018763.2,
81355521.6,
56918340.8,
78134118.4,
79408327.2,
81605227.2,
53292964.8,
79475179.2,
55133446.4,
52500392,
147282484.8,
77581524.8,
52168883.2,

79898684.8,
79855960,
81320417.6,
54317814.4,
80578881.6,
80061288,
82889003.2,
78551977.6,
78345451.2,
78937691.2,
85937094.4,
56158705.6,
77940177.6,
77725707.2,
59669937.6,
58775310.4,
78025105.6,
61351566.4,
56525313.6,
79121019.2,
92959604.8,
59355982.4,
94920427.2,
77243414.4,
94681272,
81153572,
62910486.4,
82772734.4,
65417800,
82540843.2,
64158774.4,
65148057.6,
80479744,
82003993.6,
78950216,
80165797.6,
80415985.6,
74961745.6,
78699220.8,
69556329.6,
68660648,
78437993.6,
70334991.2,
78815097.6,
95033219.2,
89046334.4,
80321555.2,
79508270.4,
79257760,
80480252.8,
87541744,
78998846.4,
78616494.4,
76649696.8,
82663540.8,
74417185.6,
83371833.6,

```
75758566.4,
81349233.6,
97204876.8,
83409332.8,
63912732.8,
69716824,
62615419.2,
84209148.8,
73905688,
77948118.4,
68794940.8,
79513627.2,
79306006.4,
76419441.6,
64419851.2,
79324332.8,
62255126.4,
59583392,
58508554.4,
56799534.4,
82361259.2,
77848576,
77652697.6,
79184798.4,
76229579.2,
81378388.8,
79585252.8,
80168929.6,
82020480,
67726940.8,
64205820.8,
80931279.2,
72156193.6,
83008113.6,
82825956.8,
74081539.2,
73529696,
86476326.4,
96271894.4,
80897329.6,
76712478.4,
0
],
},
"OutBandwidth": {
"Timestamps": [
1730620800,
1730624400,
1730628000,
1730631600,
1730635200,
1730638800,
1730642400,
1730646000,
1730649600,
1730653200,
1730656800,
```

1730660400,
1730664000,
1730667600,
1730671200,
1730674800,
1730678400,
1730682000,
1730685600,
1730689200,
1730692800,
1730696400,
1730700000,
1730703600,
1730707200,
1730710800,
1730714400,
1730718000,
1730721600,
1730725200,
1730728800,
1730732400,
1730736000,
1730739600,
1730743200,
1730746800,
1730750400,
1730754000,
1730757600,
1730761200,
1730764800,
1730768400,
1730772000,
1730775600,
1730779200,
1730782800,
1730786400,
1730790000,
1730793600,
1730797200,
1730800800,
1730804400,
1730808000,
1730811600,
1730815200,
1730818800,
1730822400,
1730826000,
1730829600,
1730833200,
1730836800,
1730840400,
1730844000,
1730847600,
1730851200,
1730854800,
1730858400,
1730862000,

1730865600,
1730869200,
1730872800,
1730876400,
1730880000,
1730883600,
1730887200,
1730890800,
1730894400,
1730898000,
1730901600,
1730905200,
1730908800,
1730912400,
1730916000,
1730919600,
1730923200,
1730926800,
1730930400,
1730934000,
1730937600,
1730941200,
1730944800,
1730948400,
1730952000,
1730955600,
1730959200,
1730962800,
1730966400,
1730970000,
1730973600,
1730977200,
1730980800,
1730984400,
1730988000,
1730991600,
1730995200,
1730998800,
1731002400,
1731006000,
1731009600,
1731013200,
1731016800,
1731020400,
1731024000,
1731027600,
1731031200,
1731034800,
1731038400,
1731042000,
1731045600,
1731049200,
1731052800,
1731056400,
1731060000,
1731063600,
1731067200,

```
1731070800,  
1731074400,  
1731078000,  
1731081600,  
1731085200,  
1731088800,  
1731092400,  
1731096000,  
1731099600,  
1731103200,  
1731106800,  
1731110400,  
1731114000,  
1731117600,  
1731121200,  
1731124800,  
1731128400,  
1731132000,  
1731135600,  
1731139200,  
1731142800,  
1731146400,  
1731150000,  
1731153600,  
1731157200,  
1731160800,  
1731164400,  
1731168000,  
1731171600,  
1731175200,  
1731178800,  
1731182400,  
1731186000,  
1731189600,  
1731193200,  
1731196800,  
1731200400,  
1731204000,  
1731207600,  
1731211200,  
1731214800,  
1731218400,  
1731222000,  
1731225600  
],  
"Values": [  
36999115.2,  
22211576,  
26118236,  
24996732.8,  
26356196,  
21585738.4,  
23174764.8,  
26288543.2,  
26900160.8,  
99745402.4,  
35897952,
```

```
23224077.6,  
31193186.4,  
26717264,  
29806122.4,  
27018134.4,  
24869877.6,  
21821033.6,  
97160590.4,  
1029837471.2,  
826950242.4,  
27306068,  
783213719.2,  
735737281.6,  
39429408,  
367326570.4,  
828035058.4,  
21186025.6,  
108722878.4,  
273322955.2,  
40808271.2,  
100828276.8,  
35679386.4,  
25438852,  
35213413.6,  
25728751.2,  
40379624,  
31101035.2,  
28520564.8,  
21724843.2,  
25904977.6,  
25237004,  
291225818.4,  
231092901.6,  
31492640,  
287575102.4,  
319184240.8,  
961222693.6,  
110587819.2,  
24934650.4,  
45784839.2,  
232243744.8,  
23227712,  
290225629.6,  
21282664,  
30819294.4,  
31482244,  
24500372.8,  
32165135.2,  
32049396.8,  
37681712,  
30539736,  
27284070.4,  
27414778.4,  
30744979.2,  
27339835.2,  
45225961.6,  
37585009.6,
```

41766510.4,
26610352.8,
27001920.8,
304289661.6,
293225852.8,
332127696.8,
284271099.2,
339752028,
294501792,
293052457.6,
873748238.4,
294181468.8,
305642092.8,
300915957.6,
301319784.8,
292014061.6,
296302379.2,
297040997.6,
296378999.2,
319333328,
303434059.2,
297750983.2,
291683778.4,
315162956,
416917124.8,
309748029.6,
738242148,
700079303.2,
695702679.2,
699579744.8,
473733113.6,
828144643.2,
518231894.4,
301627165.6,
309032832,
313636580.8,
281889774.4,
281066460.8,
283792223.2,
306457615.2,
303149468,
289305379.2,
294973648,
291938220.8,
305713736,
458142682.4,
839144659.2,
813517673.6,
344749780,
292966759.2,
288344075.2,
295100691.2,
857292094.4,
313324036,
287645835.2,
301090163.2,
288435984,

```
293894850.4,  
301641326.4,  
295098173.6,  
291204644,  
320189738.4,  
299422432,  
275096212.8,  
30181125.6,  
26611652,  
24890440.8,  
21409076,  
23864282.4,  
26384297.6,  
21293265.6,  
26109288,  
46862366.4,  
21076459.2,  
25791008,  
288045636.8,  
296784739.2,  
303341800,  
297660486.4,  
289597639.2,  
290814651.2,  
296367544.8,  
298151473.6,  
304698419.2,  
307828205.6,  
289707242.4,  
297436875.2,  
294823195.2,  
305869428.8,  
285937694.4,  
301310613.6,  
294514848.8,  
291559908.8,  
301528724.8,  
296791919.2,  
299264318.4,  
299598874.4,  
301682541.6,  
289346253.6,  
304087192.8,  
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: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [CNB](#), [GitHub](#), [Gitee](#)

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

该接口暂无业务逻辑相关的错误码，其他错误码详见 [公共错误码](#)。

查询专有集群配置列表

最近更新时间：2025-09-12 01:25:33

1. 接口描述

接口请求域名：`cdc.tencentcloudapi.com`。

查询专有集群配置列表

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：DescribeDedicatedClusterTypes。
Version	是	String	公共参数 ，本接口取值：2020-12-14。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
Name	否	String	模糊匹配专用集群配置名称 示例值：my-type
DedicatedClusterTypeIds.N	否	Array of String	待查询的专用集群配置id列表 示例值：["dctype-lkm17jbu"]
Offset	否	Integer	偏移量，默认为0。关于Offset的更进一步介绍请参考 API 简介 中的相关小节。 示例值：0
Limit	否	Integer	返回数量，默认为20，最大值为100。关于Limit的更进一步介绍请参考 API 简介 中的相关小节。 示例值：20
IsCompute	否	Boolean	是否只查询计算规格类型 示例值：true

3. 输出参数

参数名称	类型	描述
DedicatedClusterTypeSet	Array of DedicatedClusterType	专用集群配置列表
TotalCount	Integer	符合条件的个数 示例值：20
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 查询专有集群配置列表

查询专用集群订单配置列表。

输入示例

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

```
{
  "Name": "yuan dedicated",
  "DedicatedClusterTypeIds": [
    "dctype-lkm17jbu",
    "dctype-1yo16f1a",
    "dctype-b38d7s16",
    "dctype-z387eu8j"
  ],
  "Offset": 0,
  "Limit": 10,
  "IsCompute": true
}
```

输出示例

```
{
  "Response": {
    "DedicatedClusterTypeSet": [
      {
        "DedicatedClusterTypeId": "dctype-lkm17jbu",
        "Description": "first type",
        "Name": "yuan dedicated",
        "CreateTime": "2020-12-16T06:21:41Z",
        "SupportedStorageType": [],
        "SupportedUplinkGiB": [
          40,
          100
        ],
        "SupportedInstanceFamily": [
          "S5"
        ],
        "Weight": 3000,
        "PowerDrawKva": "2000.800000",
        "ComputeFormatDesc": "M5.21XLARGE700 4、S5.21XLARGE320 2"
      },
      {
        "DedicatedClusterTypeId": "dctype-1yo16f1a",
        "Description": "seco type",
        "Name": "abc dedicated",
        "CreateTime": "2020-12-16T06:33:49Z",
        "SupportedStorageType": [],
        "SupportedUplinkGiB": [
          13,
          100
        ],
        "SupportedInstanceFamily": [
          "S1",
          "S5"
        ],
        "Weight": 3273,
```

```
"PowerDrawKva": "4156.700000",
"ComputeFormatDesc": "M5.21XLARGE700 4、S5.21XLARGE320 2"
},
{
  "DedicatedClusterTypeId": "dctype-b38d7s16",
  "Description": "heynew",
  "Name": "type und",
  "CreateTime": "2020-12-16T06:33:49Z",
  "SupportedStorageType": [],
  "SupportedUplinkGiB": [
    10,
    40
  ],
  "SupportedInstanceFamily": [
    "S3"
  ],
  "Weight": 3724,
  "PowerDrawKva": "3829.100000",
  "ComputeFormatDesc": "M5.21XLARGE700 4、S5.21XLARGE320 2"
},
{
  "DedicatedClusterTypeId": "dctype-z387eu8j",
  "Description": "third one",
  "Name": "unknow",
  "CreateTime": "2020-12-16T06:33:49Z",
  "SupportedStorageType": [],
  "SupportedUplinkGiB": [
    10,
    40
  ],
  "SupportedInstanceFamily": [],
  "Weight": 3724,
  "PowerDrawKva": "3829.100000",
  "ComputeFormatDesc": "M5.21XLARGE700 4、S5.21XLARGE320 2"
}
],
"TotalCount": 4,
"RequestId": "66221c03-1429-4b8b-b66a-dffa2771536e"
}
}
```

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: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [CNB](#), [GitHub](#), [Gitee](#)

- Tencent Cloud SDK 3.0 for .NET: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [CNB](#), [GitHub](#), [Gitee](#)

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 [公共错误码](#)。

错误码	描述
InvalidParameterValue	参数取值错误。
InvalidParameterValue.InvalidAppldFormat	无效的Appld

查询专用集群列表

最近更新时间：2025-09-12 01:25:32

1. 接口描述

接口请求域名：`cdc.tencentcloudapi.com`。

查询专用集群列表

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：DescribeDedicatedClusters。
Version	是	String	公共参数 ，本接口取值：2020-12-14。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
DedicatedClusterIds.N	否	Array of String	按照一个或者多个实例ID查询。实例ID形如： <code>cluster-xxxxxxxx</code> 示例值：["cluster-du3hdu"]
Zones.N	否	Array of String	按照可用区名称过滤 示例值：ap-guangzhou
Sitelds.N	否	Array of String	按照站点id过滤 示例值：site-378xs3g
LifecycleStatuses.N	否	Array of String	按照专用集群生命周期过滤 示例值：Pending
Name	否	String	模糊匹配专用集群名称 示例值：my-cluster
Offset	否	Integer	偏移量，默认为0。关于Offset的更进一步介绍请参考 API 简介 中的相关小节。 示例值：0
Limit	否	Integer	返回数量，默认为20，最大值为100。关于Limit的更进一步介绍请参考 API 简介 中的相关小节。 示例值：20

3. 输出参数

参数名称	类型	描述
DedicatedClusterSet	Array of DedicatedCluster	符合查询条件的专用集群列表 示例值：[{ "DedicatedClusterId": "cluster-e88qh1ue", "Zone": "ap-guangzhou-2", "Description": "this is a description", "Name": "wechen-first-dc", "LifecycleStatus": "PENDING", "CreateTime": "2020-12-10T06:15:43Z", "Siteld": "site-38d9ck" }]
TotalCount	Integer	符合条件的专用集群数量。 示例值：20
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 查询专用集群列表

查询专用集群列表

输入示例

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

{}
```

输出示例

```
{
  "Response": {
    "DedicatedClusterSet": [
      {
        "DedicatedClusterId": "cluster-e88qh1ue",
        "Zone": "ap-guangzhou-2",
        "Description": "this is a description",
        "Name": "wechen-first-dc",
        "LifecycleStatus": "PENDING",
        "CreateTime": "2020-12-10T06:15:43Z",
        "SiteId": "site-38d9ck"
      },
      {
        "DedicatedClusterId": "cluster-1hza0fbu",
        "Zone": "ap-guangzhou-2",
        "Description": "this is a description",
        "Name": "wechen-first-dc",
        "LifecycleStatus": "PENDING",
        "CreateTime": "2020-12-10T06:15:50Z",
        "SiteId": "site-38d9ck"
      },
      {
        "DedicatedClusterId": "cluster-ptrfvhyw",
        "Zone": "ap-guangzhou-2",
        "Description": "this is a description",
        "Name": "wechen-first-dc",
        "LifecycleStatus": "PENDING",
        "CreateTime": "2020-12-10T06:17:26Z",
        "SiteId": "site-38d9ck"
      }
    ],
    "TotalCount": 3,
    "RequestId": "92b47775-2f39-4ad0-b5dd-0a93a8f12636"
  }
}
```

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: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [CNB](#), [GitHub](#), [Gitee](#)

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 [公共错误码](#)。

错误码	描述
InvalidParameterValue	参数取值错误。
InvalidParameterValue.InvalidAppldFormat	无效的Appld
InvalidParameterValue.LimitExceeded	超出大小限制。

查询专用集群支持的可用区列表

最近更新时间：2025-09-12 01:25:32

1. 接口描述

接口请求域名：cdc.tencentcloudapi.com。

查询专用集群支持的可用区列表

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：DescribeDedicatedSupportedZones。
Version	是	String	公共参数 ，本接口取值：2020-12-14。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
Regions.N	否	Array of Integer	传入region列表 示例值：[4]

3. 输出参数

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

4. 示例

示例1 查询专用集群支持的可用区列表

输入示例

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

{
  "Regions": [
    4
  ]
}
```

输出示例

```
{
  "Response": {
    "RequestId": "932b1bb8-8b29-405a-a4f9-1a46213c05f6",
    "ZoneSet": [
      {
        "RegionId": 4,
        "Zones": [
          {
            "Zone": "ap-shanghai-1",
            "ZoneId": 200001,
            "ZoneName": "上海一区",
            "ZoneState": "AVAILABLE"
          },
          {
            "Zone": "ap-shanghai-2",
            "ZoneId": 200002,
            "ZoneName": "上海二区",
            "ZoneState": "AVAILABLE"
          },
          {
            "Zone": "ap-shanghai-3",
            "ZoneId": 200003,
            "ZoneName": "上海三区",
            "ZoneState": "AVAILABLE"
          },
          {
            "Zone": "ap-shanghai-4",
            "ZoneId": 200004,
            "ZoneName": "上海四区",
            "ZoneState": "AVAILABLE"
          },
          {
            "Zone": "ap-shanghai-5",
            "ZoneId": 200005,
            "ZoneName": "上海五区",
            "ZoneState": "AVAILABLE"
          },
          {
            "Zone": "ap-shanghai-6",
            "ZoneId": 200006,
            "ZoneName": "上海六区",
            "ZoneState": "AVAILABLE"
          },
          {
            "Zone": "ap-shanghai-7",
            "ZoneId": 200007,
            "ZoneName": "上海七区",
            "ZoneState": "AVAILABLE"
          }
        ]
      }
    ]
  }
}
```

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: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [CNB](#), [GitHub](#), [Gitee](#)

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 [公共错误码](#)。

错误码	描述
InvalidParameterValue.InvalidValueRegion	region无效。
InvalidParameterValue.Region	无效Region ID 。
InvalidParameterValue.ZoneMismatchRegion	zone和region不匹配。

查询站点列表

最近更新时间：2025-09-12 01:25:31

1. 接口描述

接口请求域名：cdc.tencentcloudapi.com。

查询站点列表

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：DescribeSites。
Version	是	String	公共参数 ，本接口取值：2020-12-14。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
Sitelds.N	否	Array of String	按照站点id过滤 示例值：["site-98dj3kd"]
Name	否	String	模糊匹配站点名称 示例值：first-site
Offset	否	Integer	偏移量，默认为0。关于Offset的更进一步介绍请参考 API 简介 中的相关小节。 示例值：0
Limit	否	Integer	返回数量，默认为20，最大值为100。关于Limit的更进一步介绍请参考 API 简介 中的相关小节。 示例值：20

3. 输出参数

参数名称	类型	描述
SiteSet	Array of Site	符合查询条件的站点列表 示例值：[{ "Siteld": "site-98dj3kd", "Name": "my-site", "Description": "my first site", "CreateTime": "2020-12-25T08:39:57Z" }]
TotalCount	Integer	符合条件的站点数量。 示例值：20
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 查询站点列表

查询站点列表

输入示例

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

```
X-TC-Action: DescribeSites
```

```
<公共请求参数>
```

```
{}
```

输出示例

```
{
  "Response": {
    "SiteSet": [
      {
        "SiteId": "site-98dj3kd",
        "Name": "my-site",
        "Description": "my first site",
        "CreateTime": "2020-12-25T08:39:57Z"
      }
    ],
    "TotalCount": 1,
    "RequestId": "60f56861-42af-4778-96c2-018d35141a7f"
  }
}
```

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: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [CNB](#), [GitHub](#), [Gitee](#)

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

该接口暂无业务逻辑相关的错误码，其他错误码详见 [公共错误码](#)。

查询站点详情

最近更新时间：2025-09-12 01:25:31

1. 接口描述

接口请求域名：cdc.tencentcloudapi.com。

查询站点详情

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer

</> 点击调试

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

2. 输入参数

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

参数名称	必选	类型	描述
Action	是	String	公共参数 ，本接口取值：DescribeSitesDetail。
Version	是	String	公共参数 ，本接口取值：2020-12-14。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
Sitelds.N	否	Array of String	按照站点id过滤 示例值：["site-98dj3kd"]
Offset	否	Integer	偏移量，默认为0。关于Offset的更进一步介绍请参考 API 简介 中的相关小节。 示例值：0
Limit	否	Integer	返回数量，默认为20，最大值为100。关于Limit的更进一步介绍请参考 API 简介 中的相关小节。 示例值：20
Name	否	String	按照站定名称模糊匹配 示例值：my-site

3. 输出参数

参数名称	类型	描述
SiteDetailSet	Array of SiteDetail	站点详情 示例值：[{"Siteld": "site-98dj3kd", "Name": "sitename", "Description": "this is a description", "PowerConnectors": "1000Base-SX", "OptionalAddressLine": "详细地址", "Country": "China", "Province": "guangdong", "City": "shenzhen", "PostalCode": 518000}]
TotalCount	Integer	符合条件的站点总数 示例值：20
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 查询站点详情

查询站点详情

输入示例

```
POST / HTTP/1.1
Host: cdc.tencentcloudapi.com
```

```
Content-Type: application/json
X-TC-Action: DescribeSitesDetail
<公共请求参数>
```

```
{
  "SiteIds": [
    "site-98dj3kd"
  ]
}
```

输出示例

```
{
  "Response": {
    "SiteDetailSet": [
      {
        "SiteId": "site-98dj3kd",
        "Name": "sitename",
        "Description": "this is a description",
        "CreateTime": "2020-12-25T08:39:57Z",
        "FiberType": "MM",
        "UplinkSpeedGbps": 0,
        "UplinkCount": 0,
        "OpticalStandard": "MM",
        "RedundantNetworking": false,
        "PowerConnectors": "1000Base-SX",
        "PowerFeedDrop": "above",
        "PowerDrawKva": 0.0,
        "ConditionRequirement": false,
        "DimensionRequirement": false,
        "NeedHelp": false,
        "BreakerRequirement": false,
        "RedundantPower": false,
        "MaxWeight": 0,
        "AddressLine": "Nanshan Road",
        "OptionalAddressLine": "详细地址",
        "Country": "China",
        "Province": "guangdong",
        "City": "shenzhen",
        "PostalCode": 518000
      }
    ],
    "TotalCount": 1,
    "RequestId": "a07a5f9c-277c-4f03-a338-c2863982b1a6"
  }
}
```

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: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [CNB](#), [GitHub](#), [Gitee](#)

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 [公共错误码](#)。

错误码	描述
InvalidParameterValue.InvalidAppldFormat	无效的AppId

修改本地专用集群信息

最近更新时间：2025-09-12 01:25:30

1. 接口描述

接口请求域名：cdc.tencentcloudapi.com。

修改本地专用集群信息

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：ModifyDedicatedClusterInfo。
Version	是	String	公共参数 ，本接口取值：2020-12-14。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
DedicatedClusterId	是	String	本地专用集群ID 示例值：cluster-6qvrhmg
Name	否	String	集群的新名称 示例值：task_success
Zone	否	String	集群的新可用区 示例值：ap-guangzhou-2
Description	否	String	集群的新描述信息 示例值：this is a description
SiteId	否	String	集群所在站点 示例值：site-pp2cbp9k

3. 输出参数

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

4. 示例

示例1 修改本地专用集群信息

修改本地专用集群信息

输入示例

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

```
{
  "DedicatedClusterId": "cluster-dfaf323",
  "Description": "专用集群",
  "Zone": "ap-guangzhou-2",
  "Name": "专用集群",
  "SiteId": "site-pp2cbp9k"
}
```

输出示例

```
{
  "Response": {
    "RequestId": "3ceecd2-2f24-4b3f-81eb-3461376xxc2f"
  }
}
```

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: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [CNB](#), [GitHub](#), [Gitee](#)

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 [公共错误码](#)。

错误码	描述
InvalidParameterValue.TooLong	参数名过长。
InvalidParameterValue.ZoneMismatchRegion	zone和region不匹配。
InvalidParameterValue.ZoneNotSupported	当前可用区暂未支持。
MissingParameter	缺少参数错误。
MissingParameter.AtLeastOne	至少输入一个入参。
ResourceInsufficient.CloudDiskUnavailable	云硬盘余量不足。
ResourceNotFound.InvalidDedicatedClusterId	该资源没有找到。
ResourceNotFound.InvalidSiteId	站点机房无效。

修改订单状态

最近更新时间：2025-09-12 01:25:30

1. 接口描述

接口请求域名：cdc.tencentcloudapi.com。

修改大订单、小订单的状态

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer

</> 点击调试

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

2. 输入参数

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

参数名称	必选	类型	描述
Action	是	String	公共参数 ，本接口取值：ModifyOrderStatus。
Version	是	String	公共参数 ，本接口取值：2020-12-14。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
Status	是	String	要更新成的状态 示例值：DELIVERED
DedicatedClusterOrderId	否	String	大订单ID，可以在本地专用集群的基础信息页获取大订单ID 示例值：ord-aijx4ets
SubOrderIds.N	否	Array of String	小订单ID，进入大订单的详情页，可以看到小订单ID 示例值：["sord-aijxdfasdf"]

3. 输出参数

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

4. 示例

示例1 修改订单状态

修改订单状态

输入示例

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

{
  "DedicatedClusterOrderId": "ord-aijx4ets",
  "SubOrderIds": [
    "sord-aijxdfasdf"
  ],
}
```

```
"Status": "DELIVERED"
}
```

输出示例

```
{
  "Response": {
    "RequestId": "c1097ff7-eead-47de-bb86-2b9d88d29175"
  }
}
```

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: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [CNB](#), [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
ResourceNotFound.InvalidDedicatedClusterOrderId	请确认资源ID 是否存在。

修改机房设备信息

最近更新时间：2025-09-12 01:25:29

1. 接口描述

接口请求域名：cdc.tencentcloudapi.com。

修改机房设备信息

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：ModifySiteDeviceInfo。
Version	是	String	公共参数 ，本接口取值：2020-12-14。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
SiteId	是	String	机房ID 示例值：site-pjmtpk14
FiberType	否	String	您将使用光纤类型将CDC设备连接到网络。有单模和多模两种选项。 示例值：SM
OpticalStandard	否	String	您将CDC连接到网络时采用的光学标准。此字段取决于上行链路速度、光纤类型和到上游设备的距离。 示例值：MM
PowerConnectors	否	String	电源连接器类型 示例值：220VAC50HZ
PowerFeedDrop	否	String	从机架上方还是下方供电。取值范围：["UP","DOWN"] 示例值：UP
MaxWeight	否	Integer	最大承重(KG) 示例值：800
PowerDrawKva	否	Integer	功耗(KW) 示例值：5
UplinkSpeedGbps	否	Integer	网络到腾讯云Region区域的上行链路速度(Gbps) 示例值：10
UplinkCount	否	Integer	将CDC连接到网络时，每台CDC网络设备(每个机架 2 台设备)使用的上行链路数量。 示例值：3
ConditionRequirement	否	Boolean	是否满足下面环境条件： 1、场地没有材料要求或验收标准会影响 CDC 设备配送和安装。 2、确定的机架位置包含： 温度范围为 41 到 104° F (5 到 40° C)。 湿度范围为 10° F (-12° C)和 8% RH (相对湿度)到 70° F(21° C)和 80% RH。 机架位置的气流方向为从前向后，且应具有足够的 CFM (每分钟立方英尺)，散热功率须达到CDC运行功率值的 145.8 倍以上。 示例值：true
DimensionRequirement	否	Boolean	是否满足下面的尺寸条件： 您的装货站台可以容纳一个机架箱(高 x 宽 x 深 = 94" x 54" x 48")。 您可以提供从机架(高 x 宽 x 深 = 80" x 24" x 48")交货地点到机架最终安置位置的明确通道。测量深度时，应包括站台、走廊通道、门、转弯、坡道、货梯，并将其他通道限制考虑在内。

参数名称	必选	类型	描述
			在最终的 CDC安置位置，前部间隙可以为 48" 或更大，后部间隙可以为 24" 或更大。 示例值：true
RedundantNetworking	否	Boolean	是否提供冗余的上游设备(交换机或路由器)，以便实现网络出口的高可用。 示例值：true
NeedHelp	否	Boolean	是否需要腾讯云团队协助完成机架支撑工作 示例值：true
RedundantPower	否	Boolean	是否电源冗余 示例值：true
BreakerRequirement	否	Boolean	上游断路器是否具备 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 修改机房设备信息

修改机房设备信息

输入示例

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

{
  "SiteId": "site-eilhmr2",
  "FiberType": "SM",
  "OpticalStandard": "MM",
  "PowerConnectors": "220VAC50HZ",
  "PowerFeedDrop": "UP",
  "MaxWeight": 800,
  "PowerDrawKva": 5,
  "UplinkSpeedGbps": 10,
  "UplinkCount": 2,
  "ConditionRequirement": "true",
  "DimensionRequirement": "true",
  "RedundantNetworking": "true",
  "NeedHelp": "true",
  "RedundantPower": "true",
  "BreakerRequirement": "true"
}
```

输出示例

```
{
  "Response": {
    "RequestId": "3ceeecd2-2f24-4b3f-81eb-346137682c2f"
  }
}
```

```
}  
  
}
```

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: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [CNB](#), [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
MissingParameter.AtLeastOne	至少输入一个入参。

修改机房信息

最近更新时间：2025-09-12 01:25:29

1. 接口描述

接口请求域名：cdc.tencentcloudapi.com。

修改机房信息

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：ModifySiteInfo。
Version	是	String	公共参数 ，本接口取值：2020-12-14。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
SiteId	是	String	机房ID 示例值：site-pjmtpk14
Name	否	String	站点名称 示例值：测试机房
Description	否	String	站点描述 示例值：机房详细描述信息
Note	否	String	注意事项 示例值：机房注意事项
Country	否	String	站点所在国家 示例值：中国
Province	否	String	站点所在省份 示例值：广东
City	否	String	站点所在城市 示例值：深圳
PostalCode	否	String	站点所在地区的邮编 示例值：518000
AddressLine	否	String	站点所在地区的详细地址信息 示例值：南山区

3. 输出参数

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

4. 示例

示例1 修改机房信息

修改机房信息

输入示例

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

{
  "SiteId": "site-pjmtpk14",
  "Name": "new_name",
  "Description": "new_description",
  "Note": "new_note",
  "Country": "China",
  "Province": "Guangdong",
  "City": "Shenzhen",
  "PostalCode": "510000",
  "AddressLine": "腾讯滨海大厦"
}
```

输出示例

```
{
  "Response": {
    "RequestId": "3ceeeed2-2f24-4b3f-81eb-346137682c3f"
  }
}
```

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: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [CNB](#), [GitHub](#), [Gitee](#)

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 [公共错误码](#)。

错误码	描述
InvalidParameterValue.TooLong	参数名过长。
MissingParameter.AtLeastOne	至少输入一个入参。

查询本地专用集群云硬盘仓库信息

最近更新时间：2025-09-12 01:25:39

-

1. 接口描述

接口请求域名：cdc.tencentcloudapi.com。

查询本地专用集群云硬盘仓库信息

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：DescribeDedicatedClusterCbsStatistics。
Version	是	String	公共参数 ，本接口取值：2020-12-14。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
DedicatedClusterId	是	String	查询的专用集群id 示例值：cluster-gbo27yc4
SetId	否	String	云硬盘仓库id 示例值：set-mm4hzptu
StartTime	否	String	开始时间 示例值：2023-01-01 12:00:00
EndTime	否	String	结束时间 示例值：2023-01-02 12:00:00
Period	否	String	时间范围精度，1分钟(ONE_MINUTE)/5分钟(FIVE_MINUTE) 示例值：ONE_MINUTE
Offset	否	Integer	偏移量，默认为0。 示例值：0
Limit	否	Integer	返回数量，默认为20 示例值：20

3. 输出参数

参数名称	类型	描述
SetList	Array of SetInfo	云硬盘仓库信息
TotalCount	Integer	总数 示例值：1
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 查询本地专用集群云硬盘信息

查询本地专用集群云硬盘信息

输入示例

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

{
  "DedicatedClusterId": "cluster-262n63e8"
}
```

输出示例

```
{
  "Response": {
    "RequestId": "68dd812b-30ea-4d05-af65-be2bd13bd51e",
    "SetList": [
      {
        "Await": {
          "Timestamps": [],
          "Values": []
        },
        "CreateTime": "2022-12-09 15:20:09",
        "ReadIO": {
          "Timestamps": [],
          "Values": []
        },
        "ReadTraffic": {
          "Timestamps": [],
          "Values": []
        },
        "SetId": "set-mm4hzptu",
        "SetName": "HCBS_GZCDC_Z133_CDCCBS_Y0SH1125G_D0003",
        "SetSize": 33.4,
        "SetStatus": "RUNNING",
        "SetType": "ssd",
        "Util": {
          "Timestamps": [],
          "Values": []
        },
        "WriteIO": {
          "Timestamps": [],
          "Values": []
        },
        "WriteTraffic": {
          "Timestamps": [],
          "Values": []
        }
      },
      {
        "TotalCount": 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: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [CNB](#), [GitHub](#), [Gitee](#)

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

该接口暂无业务逻辑相关的错误码，其他错误码详见 [公共错误码](#)。

删除本地专用集群镜像缓存

最近更新时间：2025-09-12 01:25:40

1. 接口描述

接口请求域名：cdc.tencentcloudapi.com。

删除本地专用集群的云上镜像缓存

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：DeleteDedicatedClusterImageCache。
Version	是	String	公共参数 ，本接口取值：2020-12-14。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
DedicatedClusterId	是	String	集群id 示例值：cluster-d8htgb6k
ImageId	是	String	镜像id 示例值：img-cxi9luq2

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 删除本地专用集群的云上镜像缓存

删除本地专用集群的云上镜像缓存

输入示例

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

{
  "DedicatedClusterId": "cluster-d8htgb6k",
  "ImageId": "img-1n01e98u"
}
```

输出示例

```
{
  "Response": {
    "RequestId": "27d4ca38-284b-4ffb-ae1d-e9ff12c065b8"
  }
}
```

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: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [CNB](#), [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.InvalidImageState	镜像的状态不支持镜像同步
InternalServerError	内部错误。
InvalidParameter.ImageNotCacheInCdc	当前镜像没有在本地专用集群中缓存
UnsupportedOperation.SyncCdcImageNotSupport	账户不支持cdc镜像同步，需要先加白

创建本地专有集群云上镜像缓存

最近更新时间：2025-09-12 01:25:42

1. 接口描述

接口请求域名：cdc.tencentcloudapi.com。

创建云上镜像缓存到本地专用集群中

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：CreateDedicatedClusterImageCache。
Version	是	String	公共参数 ，本接口取值：2020-12-14。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
DedicatedClusterId	是	String	集群ID 示例值：cluster-d8htgb6k
ImageId	是	String	镜像ID 示例值：img-cxi9luq2

3. 输出参数

参数名称	类型	描述
TaskId	Integer	任务id 示例值：36182738
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）。定位问题时需提供该次请求的 RequestId。

4. 示例

示例1 创建云上镜像缓存

创建云上镜像缓存

输入示例

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

{
  "DedicatedClusterId": "cluster-d8htgb6k",
  "ImageId": "img-1n01e98u"
}
```

输出示例

```
{
  "Response": {
    "TaskId": 15249,
    "RequestId": "b621f9e4-ea1c-455c-98b4-7f85d857f222"
  }
}
```

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: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [CNB](#), [GitHub](#), [Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [CNB](#), [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.ImageIsCachedInCdc	当前镜像已经在本地专用集群中缓存
LimitExceeded.CdclImageTaskQuota	当前集群的镜像任务数量已达上限
UnsupportedOperation.SyncCdclImageNotSupport	账户不支持cdc镜像同步，需要先加白

数据结构

最近更新时间：2025-06-05 01:14:52

CbsInfo

购买的cbs信息

被如下接口引用：CreateDedicatedClusterOrder。

名称	类型	必选	描述
Size	Integer	是	cbs存储大小，单位TB 示例值：1
Type	String	是	cbs存储类型，默认为SSD 示例值：SSD

CosCapacity

CDC集群内cos的容量信息

被如下接口引用：DescribeDedicatedClusterCosCapacity。

名称	类型	描述
TotalCapacity	Float	已购cos的总容量大小，单位GB 示例值：3394.92
TotalFreeCapacity	Float	剩余可用cos的容量大小，单位GB 示例值：3367.41
TotalUsedCapacity	Float	已用cos的容量大小，单位GB 示例值：27.51

CosInfo

用于购买页面添加cos信息

被如下接口引用：CreateDedicatedClusterOrder。

名称	类型	必选	描述
Size	Integer	是	COS存储大小，单位TB 示例值：1
Type	String	是	COS存储类型，默认为cos 示例值：COS

DedicatedCluster

专用集群列表

被如下接口引用：DescribeDedicatedClusters。

名称	类型	描述
DedicatedClusterId	String	专用集群id。如"cluster-xxxxx"。 示例值：cluster-du3hdu32
Zone	String	专用集群所属可用区名称。 示例值：ap-guangzhou-2
Description	String	专用集群的描述。 示例值：this is a description
Name	String	专用集群的名称。

名称	类型	描述
		示例值：wechen-first-dc
LifecycleStatus	String	专用集群的生命周期。如"PENDING"。 示例值：PENDING
CreateTime	Timestamp ISO8601	专用集群的创建时间。 示例值：2020-12-10T06:15:43Z
Siteld	String	专用集群所属的站点id。 示例值：site-38d9ck
RunningStatus	String	专用集群的运营状态 示例值：NORMAL

DedicatedClusterInstanceType

专用宿主机支持的实例规格列表

被如下接口引用：DescribeDedicatedClusterInstanceTypes。

名称	类型	描述
Zone	String	可用区 示例值：ap-guangzhou-2
InstanceType	String	规格名称 示例值：S1.SMALL1
NetworkCard	Integer	网卡类型，例如：25代表25G网卡 示例值：0
Cpu	Integer	实例的CPU核数，单位：核。 示例值：1
Memory	Integer	实例内存容量，单位：GB。 示例值：1
InstanceFamily	String	实例机型系列。 示例值：S1
TypeName	String	机型名称。 示例值：Standard S1
StorageBlockAmount	Integer	本地存储块数量。 示例值：0
InstanceBandwidth	Float	内网带宽，单位Gbps。 示例值：0
InstancePps	Integer	网络收发包能力，单位万PPS。 示例值：0
CpuType	String	处理器型号。 示例值：intel
Gpu	Integer	实例的GPU数量。 示例值：0
Fpga	Integer	实例的FPGA数量。 示例值：0
Remark	String	机型描述 示例值：这是S1机型
Status	String	实例是否售卖。取值范围： - SELL：表示实例可购买 - SOLD_OUT：表示实例已售罄。 示例值：SOLD_OUT

DedicatedClusterOrder

专用集群订单

被如下接口引用：DescribeDedicatedClusterOrders。

名称	类型	描述
DedicatedClusterId	String	专用集群id 示例值：cluster-xh37dh
Weight	Integer	地板承重要求(KG) 示例值：0
PowerDraw	Float	功率要求(KW) 示例值：0
OrderStatus	String	订单状态 示例值：PENDING
CreateTime	Timestamp ISO8601	订单创建的时间 示例值：2024-10-16T07:18:37Z
DedicatedClusterOrderId	String	大订单ID 示例值：ord-aijx4ets
Action	String	订单类型，创建CREATE或扩容EXTEND 示例值：CREATE
DedicatedClusterOrderItems	Array of DedicatedClusterOrderItem	子订单详情列表 示例值： [{"ComputeFormat":"1SA5t.60XLARGE1076","Count":1,"DedicatedClusterTy45ndlenk","SubOrderId":"sord-gfwblx76","Weight":0}]
Cpu	Integer	cpu值 示例值：240
Mem	Integer	mem值 示例值：1076
Gpu	Integer	gpu值 示例值：0
PayStatus	Integer	0代表未支付，1代表已支付 示例值：1
PayType	String	支付方式，一次性、按月、按年 示例值：MONTHLY
TimeUnit	String	购买时长的单位 示例值：YEAR
TimeSpan	Integer	购买时长 示例值：1
OrderType	String	订单类型 示例值：CUSTOM
CheckStatus	String	验收状态 注意：此字段可能返回 null，表示取不到有效值。 示例值：DELIVERING
DeliverExpectTime	String	交付预期时间 注意：此字段可能返回 null，表示取不到有效值。 示例值：2024-01-01 12:00:00
DeliverFinishTime	String	交付实际完成时间 注意：此字段可能返回 null，表示取不到有效值。 示例值：2024-01-01 12:00:00
CheckExpectTime	String	验收预期时间 注意：此字段可能返回 null，表示取不到有效值。

名称	类型	描述
		示例值：2024-01-01 12:00:00
CheckFinishTime	String	验收实际完成时间 注意：此字段可能返回 null，表示取不到有效值。 示例值：2024-01-01 12:00:00
OrderSLA	String	订单SLA 注意：此字段可能返回 null，表示取不到有效值。 示例值：null
OrderPayPlan	String	订单支付计划 注意：此字段可能返回 null，表示取不到有效值。 示例值：null

DedicatedClusterOrderItem

专用集群子订单

被如下接口引用：DescribeDedicatedClusterOrders。

名称	类型	描述
DedicatedClusterTypeId	String	专用集群类型id 示例值：dctype-01000001
SupportedStorageType	Array of String	支持的存储类型列表 示例值：["SSD"]
SupportedUplinkSpeed	Array of Integer	支持的上连交换机的链路传输速率(GiB) 示例值：[10,25,40,100]
SupportedInstanceFamily	Array of String	支持的实例族列表 示例值：["S5"]
Weight	Integer	地板承重要求(KG) 示例值：0
PowerDraw	Float	功率要求(KW) 示例值：0
SubOrderStatus	String	订单状态 示例值：PENDING
CreateTime	Timestamp ISO8601	订单创建的时间 示例值：2022-08-15 15:08:12
SubOrderId	String	子订单ID 示例值：ord-aijx4ets
Count	Integer	关联的集群规格数量 示例值：1
Name	String	规格简单描述 示例值：通用小型机
Description	String	规格详细描述 示例值：适用于...场景
TotalCpu	Integer	CPU数 示例值：1
TotalMem	Integer	内存数 示例值：1
TotalGpu	Integer	GPU数 示例值：1
TypeName	String	规格英文名 示例值：CBS.SSD

名称	类型	描述
ComputeFormat	String	规格展示 示例值：M5.21XLARGE700 4、S5.21XLARGE320 2
TypeFamily	String	规格类型 示例值：COMPUTE
SubOrderPayStatus	Integer	0未支付，1已支付 示例值：0

DedicatedClusterType

专用集群配置

被如下接口引用：DescribeDedicatedClusterTypes。

名称	类型	描述
DedicatedClusterTypeId	String	配置id 示例值：dctype-00000000
Description	String	配置描述，对应描述 示例值：适用于xxx场景
Name	String	配置名称，对应计算资源类型 示例值：通用小型机规格
CreateTime	Timestamp ISO8601	创建配置的时间 示例值：2024-10-30 17:14:13
SupportedStorageType	Array of String	支持的存储类型列表 示例值：COS
SupportedUplinkGiB	Array of Integer	支持的上连交换机的链路传输速率 示例值：10
SupportedInstanceFamily	Array of String	支持的实例族列表 示例值：M5
Weight	Integer	地板承重要求(KG) 示例值：1000
PowerDrawKva	Float	功率要求(KW) 示例值：100
ComputeFormatDesc	String	显示计算资源规格详情，存储等资源不显示 示例值：M5.21XLARGE700、S5.21XLARGE320

DedicatedClusterTypeInfo

DedicatedClusterType => (Id, Count)

被如下接口引用：CreateDedicatedClusterOrder。

名称	类型	必选	描述
Id	String	是	集群类型Id 示例值：dctype-00000000
Count	Integer	是	集群类型个数 示例值：1

DetailData

带有时间戳的详细数据。

被如下接口引用：DescribeDedicatedClusterCbsStatistics, DescribeDedicatedClusterHostStatistics。

名称	类型	描述
Timestamps	Array of Float	时间戳 示例值: [0]
Values	Array of Float	对应的具体值 示例值: [0]

HostDetailInfo

宿主机资源的概览详细信息。

被如下接口引用：DescribeDedicatedClusterOverview。

名称	类型	描述
HostTypeFamily	String	类型族 示例值: 标准型S5
CpuTotal	Float	总CPU 示例值: 64
CpuAvailable	Float	可用CPU 示例值: 64
MemTotal	Float	总内存 示例值: 320
MemAvailable	Float	可用内存 示例值: 320

HostInfo

CDC宿主机的详细信息

被如下接口引用：DescribeDedicatedClusterHosts。

名称	类型	描述
ServiceType	String	云服务类型 示例值: 云服务器宿主机
HostStatus	String	宿主机运行状态 示例值: ONLINE
HostType	String	宿主机类型 示例值: CG2-25G
CpuAvailable	Integer	cpu可用数 示例值: 32
CpuTotal	Integer	cpu总数 示例值: 32
MemAvailable	Integer	内存可用数 示例值: 64
MemTotal	Integer	内存总数 示例值: 64
RunTime	String	运行时间 示例值: 2020/12/18
ExpireTime	String	到期时间 示例值: 2023/12/18
HostId	String	宿主机id 示例值: host-xxxx

HostStatistic

CDC集群内宿主机的统计信息

被如下接口引用：DescribeDedicatedClusterHostStatistics。

名称	类型	描述
HostType	String	宿主机规格 示例值：S5.21XLARGE320
HostFamily	String	宿主机机型系列 示例值：S5
Cpu	Integer	宿主机的CPU核数，单位：核 示例值：84
Memory	Integer	宿主机内存大小，单位：GB 示例值：320
Count	Integer	该规格宿主机的数量 示例值：240
CpuAverage	Float	平均cpu负载百分比 示例值：10
MemAverage	Float	平均内存使用率百分比 示例值：10
NetAverage	Float	平均网络流量 示例值：10
CpuDetailData	DetailData	cpu详细监控数据 示例值：{"Timestamps": [1693274400], "Values": [0]}
MemDetailData	DetailData	内存详细数据 示例值：{"Timestamps": [1693274400], "Values": [0]}
NetRateDetailData	DetailData	网络速率详细数据 示例值：{"Timestamps": [1693274400], "Values": [0]}
NetPacketDetailData	DetailData	网速包详细数据 示例值：{"Timestamps": [1693274400], "Values": [0]}

InBandwidth

入带宽数据

被如下接口引用：DescribeDedicatedClusterOverview。

名称	类型	描述
Timestamps	Array of Float	时间戳 示例值：[0]
Values	Array of Float	时间对应的值 示例值：[0]

LocalNetInfo

本地网络信息

被如下接口引用：DescribeDedicatedClusterOverview。

名称	类型	描述
Protocol	String	协议 示例值：BGP
VpcId	String	网络id

名称	类型	描述
		示例值：vpc-xxxx
BGPRoute	String	路由信息 示例值：10.0.0.0/24
LocalIp	String	本地IP 示例值：10.0.0.0

OutBandwidth

出带宽数据。

被如下接口引用：DescribeDedicatedClusterOverview。

名称	类型	描述
Timestamps	Array of Float	时间戳 示例值：[0]
Values	Array of Float	对应时间的值 示例值：[0]

RegionZoneInfo

RegionZoneInfo信息

被如下接口引用：DescribeDedicatedSupportedZones。

名称	类型	描述
RegionId	Integer	Region id 示例值：1
Zones	Array of ZoneInfo	ZoneInfo数组 示例值：[{"Zone": "ap-guangzhou-1", "ZoneName": "广州一区", "ZoneId": "100001", "ZoneState": "AVAILABLE"}]

SetInfo

云硬盘的仓库级别信息

被如下接口引用：DescribeDedicatedClusterCbsStatistics。

名称	类型	描述
SetId	String	云硬盘仓库id 示例值：set-mm4hzptu
SetName	String	云硬盘仓库名称 示例值：HCBS_GZCDC_Z133_CDCCBS_Y0SH1125G_D0003
SetType	String	云硬盘仓库类型 示例值：ssd
SetSize	Float	云硬盘仓库容量 示例值：33.4
SetStatus	String	云硬盘仓库状态 示例值：RUNNING
CreateTime	String	云硬盘仓库创建时间 示例值：2022-12-09 15:20:09
ReadTraffic	DetailData	读流量 示例值：{"Timestamps": [1693274400], "Values": [0]}

名称	类型	描述
WriteTraffic	DetailData	写流量 示例值: {"Timestamps": [1693274400], "Values": [0]}
ReadIO	DetailData	读IO 示例值: {"Timestamps": [1693274400], "Values": [0]}
WriteIO	DetailData	写IO 示例值: {"Timestamps": [1693274400], "Values": [0]}
Await	DetailData	平均等待时间 示例值: {"Timestamps": [1693274400], "Values": [0]}
Util	DetailData	利用率 示例值: {"Timestamps": [1693274400], "Values": [0]}

Site

客户站点信息

被如下接口引用: DescribeSites。

名称	类型	描述
Name	String	站点名称 示例值: my-site
SiteId	String	站点id 示例值: site-98dj3kd
Description	String	站点描述 示例值: my first site
CreateTime	Timestamp ISO8601	站点创建时间 示例值: 2020-12-25T08:39:57Z

SiteDetail

站点详情

被如下接口引用: DescribeSitesDetail。

名称	类型	描述
SiteId	String	站点id 示例值: "site-u37d8sh"
Name	String	站点名称 示例值: "my-site"
Description	String	站点描述 示例值: "this site is good"
CreateTime	Timestamp ISO8601	站点创建时间 示例值: "2020-12-25T10:26:11Z"
FiberType	String	光纤类型 示例值: "MM"
UplinkSpeedGbps	Integer	网络到腾讯云Region区域的上行链路速度 示例值: 0
UplinkCount	Integer	将CDC连接到网络时, 每台CDC网络设备(每个机架 2 台设备)使用的上行链路数量。 示例值: 0
OpticalStandard	String	将CDC连接到网络时采用的光学标准 示例值: "MM"

名称	类型	描述
RedundantNetworking	Boolean	是否提供冗余的上游设备(交换机或路由器)，以便两台 网络设备都能连接到网络设备。 示例值: false
PowerConnectors	String	电源连接器类型 示例值: "MM"
PowerFeedDrop	String	从机架上方还是下方供电。 示例值: "above"
PowerDrawKva	Float	功耗(KW) 示例值: 0.0
ConditionRequirement	Boolean	是否满足下面环境条件: 1、场地没有材料要求或验收标准会影响 CDC 设备配送和安装。 2、确定的机架位置包含: 温度范围为 41 到 104° F (5 到 40° C)。 湿度范围为 10° F (–12° C) 和 8% RH (相对湿度)到 70° F(21° C)和 80% RH。 机架位置的气流方向为从前向后，且应具有足够的 CFM (每分钟立方英尺)。CFM 必须是 CDC 配置的 kVA 功耗值的 145.8 倍。 示例值: false
DimensionRequirement	Boolean	是否满足下面的尺寸条件: 您的装货站台可以容纳一个机架箱(高 x 宽 x 深 = 94" x 54" x 48")。 您可以提供从机架(高 x 宽 x 深 = 80" x 24" x 48")交货地点到机架最终安置位置的明确通道。测量深度时，应包括站台、走廊通道、门、转弯、坡道、货梯，并将其他通道限制考虑在内。 在最终的 CDC安置位置，前部间隙可以为 48" 或更大，后部间隙可以为 24" 或更大。 示例值: false
MaxWeight	Integer	最大承重(KG) 示例值: 0
AddressLine	String	站点地址 示例值: "Furong Road"
OptionalAddressLine	String	站点所在地区的详细地址信息（补充） 示例值: 305号
NeedHelp	Boolean	是否需要腾讯云团队协助完成机架支撑工作 示例值: true
BreakerRequirement	Boolean	上游断路器是否具备 示例值: false
RedundantPower	Boolean	是否电源冗余 示例值: false
Country	String	站点所在国家 示例值: "China"
Province	String	站点所在省份 示例值: "Hunan"
City	String	站点所在城市 示例值: "Changsha"
PostalCode	Integer	站点所在地区的邮编 示例值: 414012

VpngwBandwidthData

VPN网关的流量监控数据。

被如下接口引用：DescribeDedicatedClusterOverview。

名称	类型	描述
OutBandwidth	OutBandwidth	出带宽流量 示例值: {"Timestamps": [1693274400],"Values": [0]}

名称	类型	描述
InBandwidth	InBandwidth	入带宽流量 示例值：{"Timestamps": [1693274400],"Values": [0]}

ZoneInfo

可用区信息

被如下接口引用：DescribeDedicatedSupportedZones。

名称	类型	描述
Zone	String	可用区名称 示例值：ap-guangzhou-1
ZoneName	String	可用区描述 示例值：广州一区
ZoneId	Integer	可用区ID 示例值：100001
ZoneState	String	可用区状态，包含AVAILABLE和UNAVAILABLE。AVAILABLE代表可用，UNAVAILABLE代表不可用。 示例值：AVAILABLE

错误码

最近更新时间：2025-04-24 01:10:03

功能说明

如果返回结果中存在 Error 字段，则表示调用 API 接口失败。例如：

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

Error 中的 Code 表示错误码，Message 表示该错误的具体信息。

错误码列表

公共错误码

错误码	说明
ActionOffline	接口已下线。
AuthFailure.InvalidAuthorization	请求头部的 Authorization 不符合腾讯云标准。
AuthFailure.InvalidSecretId	密钥非法（不是云 API 密钥类型）。
AuthFailure.MFAFailure	MFA 错误。
AuthFailure.SecretIdNotFound	密钥不存在。请在 控制台 检查密钥是否已被删除或者禁用，如状态正常，请检查密钥是否填写正确，注意前后不得有空格。
AuthFailure.SignatureExpire	签名过期。Timestamp 和服务器时间相差不得超过五分钟，请检查本地时间是否和标准时间同步。
AuthFailure.SignatureFailure	签名错误。签名计算错误，请对照调用方式中的签名方法文档检查签名计算过程。
AuthFailure.TokenFailure	token 错误。
AuthFailure.UnauthorizedOperation	请求未授权。请参考 CAM 文档对鉴权的说明。
DryRunOperation	DryRun 操作，代表请求将会是成功的，只是多传了 DryRun 参数。
FailedOperation	操作失败。
InternalError	内部错误。
InvalidAction	接口不存在。
InvalidParameter	参数错误（包括参数格式、类型等错误）。
InvalidParameterValue	参数取值错误。
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。

错误码	说明
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.CdcBindVpcFail	CDC绑定VPC失败。
FailedOperation.FailDeleteSite	删除site失败。
FailedOperation.InvalidImageState	镜像的状态不支持镜像同步
InvalidParameter.ImageIsCachedInCdc	当前镜像已经在本地专用集群中缓存
InvalidParameter.ImageNotCacheInCdc	当前镜像没有在本地专用集群中缓存
InvalidParameter.InstanceTypeNotSupport	该机型暂不支持。
InvalidParameterValue.InvalidAppIdFormat	无效的AppId
InvalidParameterValue.InvalidValueDedicatedClusterCosSize	本地专用集群COS大小不正确。
InvalidParameterValue.InvalidValueDedicatedClusterDataStepSize	CBS大小不正确，它必须是40的整数倍。
InvalidParameterValue.InvalidValueRegion	region无效。
InvalidParameterValue.LimitExceeded	超出大小限制。
InvalidParameterValue.Region	无效Region ID 。
InvalidParameterValue.TooLong	参数名过长。
InvalidParameterValue.ZoneMismatchRegion	zone和region不匹配。
InvalidParameterValue.ZoneNotSupported	当前可用区暂未支持。
LimitExceeded.CdcImageTaskQuota	当前集群的镜像任务数量已达上限

错误码	说明
MissingParameter.AtLeastOne	至少输入一个入参。
ResourceInsufficient.CloudDiskUnavailable	云硬盘余量不足。
ResourceNotFound.InvalidDedicatedClusterId	该资源没有找到。
ResourceNotFound.InvalidDedicatedClusterOrderId	请确认资源ID 是否存在。
ResourceNotFound.InvalidDedicatedClusterTypeId	该资源没有找到。
ResourceNotFound.InvalidSiteId	站点机房无效。
UnsupportedOperation.NonCustomerAppIdNotSupport	不支持非CUSTOMER类型的app id。
UnsupportedOperation.SyncCdclImageNotSupport	账户不支持cdc镜像同步，需要先加白