

腾讯云区块链服务平台 TBaaS

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

长安链体验网络相关接口

调用长安链体验网络合约执行交易

查询长安链体验网络指定高度区块的交易

调用长安链体验网络合约查询

通过交易ID查询长安链体验网络交易

长安链相关接口

批量申请长安链用户签名证书

调用长安链合约执行交易

查询长安链指定高度区块的交易

调用长安链合约查询

通过交易ID查询长安链交易

Hyperledger Fabric 2.3 相关接口

获取Fabric某区块的详细信息

获取Fabric交易的详细信息

调用Fabric用户合约执行交易

调用Fabric用户合约查询

Hyperledger Fabric 1.4 相关接口

申请用户证书

下载用户证书

查询fabric的区块列表

获取区块内交易列表

获取区块链网络概要

获取fabric最新交易列表

获取交易的详情

新增交易

查询交易

Invoke异步调用结果查询

数据结构

错误码

API 文档

更新历史

最近更新时间：2025-05-16 01:58:10

第 21 次发布

发布时间：2025-05-16 01:58:04

本次发布包含了以下内容：

改善已有的文档。

删除接口：

- `GetLatesdTransactionList`

第 20 次发布

发布时间：2023-11-23 01:26:18

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- [DownloadUserCert](#)
 - 新增出参：Cert
 - 废弃出参：CertCtx
- [Invoke](#)
 - 新增出参：TxId
 - 废弃出参：Txid

第 19 次发布

发布时间：2023-11-21 01:32:15

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- [DescribeFabricBlock](#)
- [DescribeFabricTransaction](#)
- [InvokeFabricChaincode](#)
- [QueryFabricChaincode](#)

新增数据结构：

- [Transaction](#)

第 18 次发布

发布时间：2023-07-20 01:41:44

本次发布包含了以下内容：

改善已有的文档。

删除接口：

- `CreateChaincodeAndInstallForUser`
- `GetChaincodeCompileLogForUser`
- `GetChaincodeInitializeResultForUser`
- `GetChaincodeLogForUser`

- GetChannelListForUser
- GetClusterListForUser
- GetPeerLogForUser
- InitializeChaincodeForUser

删除数据结构：

- ChannelDetailForUser
- ClusterDetailForUser
- GroupDetailForUser
- LogDetailForUser
- PeerDetailForUser

第 17 次发布

发布时间：2023-06-27 01:39:58

本次发布包含了以下内容：

改善已有的文档。

预下线接口：

- CreateChaincodeAndInstallForUser
- GetChaincodeCompileLogForUser
- GetChaincodeInitializeResultForUser
- GetChaincodeLogForUser
- GetChannelListForUser
- GetClusterListForUser
- GetPeerLogForUser
- InitializeChaincodeForUser

第 16 次发布

发布时间：2023-06-26 01:51:21

本次发布包含了以下内容：

改善已有的文档。

删除接口：

- DeployDynamicBcosContract
- GetBcosBlockByNumber
- GetBcosBlockList
- GetBcosTransByHash
- GetBcosTransList
- InvokeBcosTrans

删除数据结构：

- BcosBlockObj
- BcosTransInfo

第 15 次发布

发布时间：2023-06-20 01:47:44

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- [GetLatestTransactionList](#)

预下线接口：

- DeployDynamicBcosContract
- GetBcosBlockByNumber
- GetBcosBlockList
- GetBcosTransByHash
- GetBcosTransList
- InvokeBcosTrans

第 14 次发布

发布时间：2022-11-24 06:52:49

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- [ApplyChainMakerBatchUserCert](#)

新增数据结构：

- [SignCertCsr](#)

第 13 次发布

发布时间：2022-08-11 06:49:15

本次发布包含了以下内容：

改善已有的文档。

删除接口：

- BlockByNumberHandler
- DeployDynamicContractHandler
- GetBlockListHandler
- GetTransByHashHandler
- GetTransListHandler
- SendTransactionHandler
- TransByDynamicContractHandler

第 12 次发布

发布时间：2022-06-20 15:31:45

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- [InvokeChainMakerDemoContract](#)
- [QueryChainMakerDemoBlockTransaction](#)
- [QueryChainMakerDemoContract](#)
- [QueryChainMakerDemoTransaction](#)

第 11 次发布

发布时间：2021-08-24 08:09:26

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- [InvokeChainMakerContract](#)
- [QueryChainMakerBlockTransaction](#)
- [QueryChainMakerContract](#)
- [QueryChainMakerTransaction](#)

新增数据结构：

- [ChainMakerContractResult](#)
- [ChainMakerTransactionResult](#)

第 10 次发布

发布时间：2021-03-16 17:32:27

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- [DeployDynamicBcosContract](#)
- [GetBcosBlockByNumber](#)
- [GetBcosBlockList](#)
- [GetBcosTransByHash](#)
- [GetBcosTransList](#)
- [InvokeBcosTrans](#)

第 9 次发布

发布时间：2021-01-29 08:07:22

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- [CreateChaincodeAndInstallForUser](#)
- [GetChaincodeCompileLogForUser](#)
- [GetChaincodeInitializeResultForUser](#)
- [GetChaincodeLogForUser](#)
- [GetChannelListForUser](#)
- [GetClusterListForUser](#)
- [GetPeerLogForUser](#)
- [InitializeChaincodeForUser](#)

新增数据结构：

- [ChannelDetailForUser](#)
- [ClusterDetailForUser](#)
- [GroupDetailForUser](#)
- [LogDetailForUser](#)
- [PeerDetailForUser](#)

第 8 次发布

发布时间：2020-01-13 22:02:43

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- [DeployDynamicContractHandler](#)
- [TransByDynamicContractHandler](#)

第 7 次发布

发布时间：2020-01-09 15:50:30

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- [GetBlockTransactionListForUser](#)

修改接口：

- [ApplyUserCert](#)
 - 新增出参：CertId, CertDn

第 6 次发布

发布时间：2020-01-08 11:44:16

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- [SendTransactionHandler](#)
 - 删除入参：KeyUser

第 5 次发布

发布时间：2019-11-21 21:54:40

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- [GetTransactionDetailForUser](#)

新增数据结构：

- [EndorserGroup](#)

第 4 次发布

发布时间：2019-10-18 10:31:32

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- [ApplyUserCert](#)
- [BlockByNumberHandler](#)
- [DownloadUserCert](#)
- [GetBlockListHandler](#)
- [GetTransByHashHandler](#)
- [GetTransListHandler](#)
- [SendTransactionHandler](#)
- [SrvInvoke](#)

修改接口：

- [GetClusterSummary](#)
 - 新增出参：OrderCount
 - 删除出参：OtherChannelCount, NoneChannelCount, OtherPeerCount, OtherGroupCount, OtherChaincodeCount, OrderCertCount

新增数据结构：

- BcosBlockObj
- BcosTransInfo

第 3 次发布

发布时间：2019-05-30 19:29:39

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- [GetBlockList](#)
- [GetClusterSummary](#)
- [GetLatesdTransactionList](#)

修改接口：

- [GetInvokeTx](#)
 - 新增出参：BlockId

新增数据结构：

- [Block](#)
- [TransactionItem](#)

第 2 次发布

发布时间：2019-04-25 20:19:09

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- [GetInvokeTx](#)
 - 新增入参：GroupName
- [Invoke](#)
 - 新增入参：GroupName
- [Query](#)
 - 新增入参：GroupName

第 1 次发布

发布时间：2018-08-30 15:59:14

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- [GetInvokeTx](#)
- [Invoke](#)
- [Query](#)

新增数据结构：

- [PeerSet](#)

简介

最近更新时间：2025-04-25 01:54:53

欢迎使用腾讯云区块链（TBaaS）！

腾讯云区块链致力于为企业提供金融级别区块链基础设施服务，通过区块链云上的服务，为行业提供安全、可靠、灵活的解决方案。打造一个开放分享、能力全面，标准统一的区块链生态圈，为众多行业客户提供完备的区块链服务。

1. 术语表

术语	中文	说明
channel	通道	通道是构建在“Fabric”网络上的私有区块链，实现了数据的隔离和保密。
ledger	账本	一个channel的chain和由channel中每个peer维护的world state。
chaincode	合约	链码是Hyperledger Fabric中智能合约的具体实现，定义了应用程序与区块链的交互细节，例如上链的数据格式，对上链数据的校验逻辑等，具体取决于用户希望在链上实现怎样的业务逻辑。
block	区块	在一个通道上，（区块是）一组有序交易的集合。
peer	节点	一个网络实体，维护Ledger并运行Chaincode容器来对账本执行read-write操作。

2. API快速入门

通过“交易发起”（Invoke）接口新增、修改对象的属性值，支持同步模式和异步模式。

通过“交易查询”（Query）接口查询对象的属性值。

通过“交易异步查询”（GetInvokeTx）接口查询异步模式的交易发起接口的执行结果。

API 概览

最近更新時間：2025-05-16 01:58:10

长安链体验网络相关接口

接口名称	接口功能	频率限制（次/秒）
InvokeChainMakerDemoContract	调用长安链体验网络合约执行交易	1
QueryChainMakerDemoBlockTransaction	查询长安链体验网络指定高度区块的交易	1
QueryChainMakerDemoContract	调用长安链体验网络合约查询	1
QueryChainMakerDemoTransaction	通过交易ID查询长安链体验网络交易	1

长安链相关接口

接口名称	接口功能	频率限制（次/秒）
ApplyChainMakerBatchUserCert	批量申请长安链用户签名证书	20
InvokeChainMakerContract	调用长安链合约执行交易	20
QueryChainMakerBlockTransaction	查询长安链指定高度区块的交易	20
QueryChainMakerContract	调用长安链合约查询	20
QueryChainMakerTransaction	通过交易ID查询长安链交易	20

Hyperledger Fabric 2.3 相关接口

接口名称	接口功能	频率限制（次/秒）
DescribeFabricBlock	获取Fabric某区块的详细信息	20
DescribeFabricTransaction	获取Fabric交易的详细信息	20
InvokeFabricChaincode	调用Fabric用户合约执行交易	20
QueryFabricChaincode	调用Fabric用户合约查询	20

Hyperledger Fabric 1.4 相关接口

接口名称	接口功能	频率限制（次/秒）
ApplyUserCert	申请用户证书	20
DownloadUserCert	下载用户证书	20
GetBlockList	查询fabric的区块列表	20
GetBlockTransactionListForUser	获取区块内交易列表	20
GetClusterSummary	获取区块链网络概要	20
GetLatestTransactionList	获取fabric最新交易列表	20
GetTransactionDetailForUser	获取交易的详情	5
Invoke	新增交易	50
Query	查询交易	50
GetInvokeTx	Invoke异步调用结果查询	50

TrustSQL业务场景相关接口

接口名称	接口功能	频率限制（次/秒）
SrvInvoke	trustsql服务统一接口	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 01:29:55

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

2. 通信协议

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

3. 请求方法

支持的 HTTP 请求方法:

- POST (推荐)
- GET

POST 请求支持的 Content-Type 类型:

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

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

4. 字符编码

均使用 UTF-8 编码。

公共参数

最近更新时间：2024-11-15 02:02:23

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输入公共参数 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 tbaas；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
华北地区（北京）	ap-beijing
华南地区（广州）	ap-guangzhou
港澳台地区（中国香港）	ap-hongkong
华东地区（上海金融）	ap-shanghai-fsi

签名方法 v3

最近更新时间：2024-12-25 01:59:44

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：

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

签名版本 v3 签名过程

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

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

- 云服务器默认已开通，该接口很常用；
- 该接口是只读的，不会改变现有资源的状态；
- 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, 否则容易出错
DateTime date = new DateTime(1970, 1, 1, 0, 0, 0, 0, DateTimeKind.Utc).AddSeconds(1551113065);
string requestPayload = "{\"Limit\": 1, \"Filters\": [{\"Values\": [\"\\u672a\\u547d\\u540d\"], \"Name\": \"instance-name\"}]\"}";

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

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

NodeJS

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

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

function getHash(message, encoding = 'hex') {
const hash = crypto.createHash('sha256')
return hash.update(message).digest(encoding)
}

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

function main(){
```



```
// 密钥参数
// 需要设置环境变量 TENCENTCLOUD_SECRET_ID, 值为示例的 AKID*****
const SECRET_ID = process.env.TENCENTCLOUD_SECRET_ID
// 需要设置环境变量 TENCENTCLOUD_SECRET_KEY, 值为示例的 *****
const SECRET_KEY = process.env.TENCENTCLOUD_SECRET_KEY

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

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

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

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

// ***** 步骤 2: 拼接待签名字符串 *****
const algorithm = "TC3-HMAC-SHA256"
const hashedCanonicalRequest = getHash(canonicalRequest);
const credentialScope = date + "/" + service + "/" + "tc3_request"
const stringToSign = algorithm + "\\n" +
timestamp + "\\n" +
credentialScope + "\\n" +
hashedCanonicalRequest
console.log(stringToSign)

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

// ***** 步骤 4: 拼接 Authorization *****
const authorization = algorithm + " " +
"Credential=" + SECRET_ID + "/" + credentialScope + ", " +
"SignedHeaders=" + signedHeaders + ", " +
```

```

"Signature=" + signature
console.log(authorization)

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

```

C++

```

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

using namespace std;

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

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

string sha256Hex(const string &str)
{
    char buf[3];
    unsigned char hash[SHA256_DIGEST_LENGTH];
    SHA256_CTX sha256;
    SHA256_Init(&sha256);

```

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

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

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

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

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

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

    return (ss.str());
}

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

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

```
}
return output;
}

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

string service = "cvm";
string host = "cvm.tencentcloudapi.com";
string region = "ap-guangzhou";
string action = "DescribeInstances";
string version = "2017-03-12";
int64_t timestamp = 1551113065;
string date = get_data(timestamp);

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

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

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

// ***** 步骤 4: 拼接 Authorization *****
```

```
string authorization = algorithm + " " + "Credential=" + SECRET_ID + "/" + credentialScope + ", "
+ "SignedHeaders=" + signedHeaders + ", " + "Signature=" + signature;
cout << authorization << endl;

string curlcmd = "curl -X POST https://" + host + "\n"
+ " -H \"Authorization: \" + authorization + "\"\n"
+ " -H \"Content-Type: application/json; charset=utf-8\" + "\n"
+ " -H \"Host: \" + host + "\"\n"
+ " -H \"X-TC-Action: \" + action + "\"\n"
+ " -H \"X-TC-Timestamp: \" + RequestTimestamp + "\"\n"
+ " -H \"X-TC-Version: \" + version + "\"\n"
+ " -H \"X-TC-Region: \" + region + "\"\n"
+ " -d '\" + payload + '\"';
cout << curlcmd << endl;
return 0;
};
```

C

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

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

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

void hmac_sha256(const char* key, int key_len,
const char* input, int input_len,
unsigned char* output, unsigned int* output_len)
{
    HMAC_CTX *h;
```

```
#if 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* const lut = "0123456789abcdef";

    char add_out[128] = {0};
    char temp[2] = {0};
    for (size_t i = 0; i < input_len; ++i)
    {
        const unsigned char c = input[i];
        temp[0] = lut[c >> 4];
        strcat(add_out, temp);
        temp[0] = lut[c & 15];
        strcat(add_out, temp);
    }
    strncpy(output, add_out, 128);
}

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

int main()
{
    // 密钥参数
    // 需要设置环境变量 TENCENTCLOUD_SECRET_ID, 值为示例的 AKID*****
    const char* SECRET_ID = getenv("TENCENTCLOUD_SECRET_ID");
    // 需要设置环境变量 TENCENTCLOUD_SECRET_KEY, 值为示例的 *****
    const char* SECRET_KEY = getenv("TENCENTCLOUD_SECRET_KEY");
    const char* service = "cvm";
    const char* host = "cvm.tencentcloudapi.com";
    const char* region = "ap-guangzhou";
    const char* action = "DescribeInstances";
```

```
const char* version = "2017-03-12";
int64_t timestamp = 1551113065;
char date[20] = {0};
get_utc_date(timestamp, date, sizeof(date));

// ***** 步骤 1: 拼接规范请求串 *****
const char* http_request_method = "POST";
const char* canonical_uri = "/";
const char* canonical_query_string = "";
char canonical_headers[100] = {"Content-type:application/json; charset=utf-8\nhost:"};
strcat(canonical_headers, host);
strcat(canonical_headers, "\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

最近更新时间：2024-12-25 01:59:44

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
InstanceIds.0	待查询的实例 ID	ins-09dx96dg
Offset	偏移量	0
Limit	最大允许输出	20
Version	接口版本号	2017-03-12

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

2.1. 对参数排序

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

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

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

2.2. 拼接请求字符串

此步骤生成请求字符串。

将把上一步排序好的请求参数格式化成“参数名称=参数值”的形式，如对 Action 参数，其参数名称为 "Action"，参数值为 "DescribeInstances"，因此格式化后即为 Action=DescribeInstances。

注意：“参数值”为原始值而非 url 编码后的值。

然后将格式化后的各个参数用"&"拼接在一起，最终生成的请求字符串为：

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

2.3. 拼接签名原文字符串

此步骤生成签名原文字符串。

签名原文字符串由以下几个参数构成：

1. 请求方法: 支持 POST 和 GET 方式，这里使用 GET 请求，注意方法为全大写。
2. 请求主机: 查看实例列表(DescribeInstances)的请求域名为：cvm.tencentcloudapi.com。实际的请求域名根据接口所属模块的不同而不同，详见各接口说明。
3. 请求路径: 当前版本云API的请求路径固定为 /。
4. 请求字符串: 即上一步生成的请求字符串。

签名原串的拼接规则为：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("evm.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 01:49:05

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 06:48:00

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。

长安链体验网络相关接口

调用长安链体验网络合约执行交易

最近更新时间：2025-04-25 01:54:51

1. 接口描述

接口请求域名：tbaas.tencentcloudapi.com。

调用长安链体验网络合约执行交易

默认接口请求频率限制：1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：InvokeChainMakerDemoContract。
Version	是	String	公共参数 ，本接口取值：2018-04-16。
Region	是	String	公共参数 ，详见产品支持的 地域列表 ，本接口仅支持其中的: ap-beijing。
ClusterId	是	String	网络ID，可在区块链网络详情或列表中获取 示例值：chainmaker-demo
ChainId	是	String	业务链ID，可在网络概览页获取 示例值：chain_test
ContractName	是	String	合约名称，可在合约管理中获取 示例值：fact
FuncName	是	String	合约方法名 示例值：send
FuncParam	否	String	合约方法入参，json格式字符串，key/value都是string类型的map 示例值：{"name":"test"}
AsyncFlag	否	Integer	是否异步执行，1为是，否则为0；如果异步执行，可使用返回值中的交易TxID查询执行结果 示例值：1

3. 输出参数

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

4. 示例

示例1 调用长安链体验网络合约执行交易

调用长安链体验网络合约执行交易

输入示例

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

{
  "ChainId": "chain_test",
  "FuncParam": "{\"time\":\"2021-07-26\", \"file_hash\":\"123456\", \"file_name\":\"test.jpg\"}",
  "ClusterId": "chainmaker-demo",
  "FuncName": "save",
  "AsyncFlag": "0",
  "ContractName": "fact"
}
```

输出示例

```
{
  "Response": {
    "RequestId": "2974e7ac-72e2-49c3-ae39-bc2bc6a7330b",
    "Result": {
      "Code": 0,
      "CodeMessage": "OK",
      "Message": "",
      "TxId": "963b8210f10e45e6ab6fe5cd37ba7f6719db5948a38c494ca9137fa2b8c9033d",
      "GasUsed": 9573483,
      "Result": "b2s="
    }
  }
}
```

5. 开发者资源

腾讯云 API 平台

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

API Inspector

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

SDK

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

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

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

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

错误码	描述
AuthFailure	CAM签名/鉴权错误。
FailedOperation.BaaSStopServing	为带来更好的服务与体验，长安链体验网络维护升级中，请稍后再试。
FailedOperation.InvalidOperation	用户非法操作。
FailedOperation.UserInBlackList	您因违反用户协议，目前无法使用长安链体验网络。
FailedOperation.UserNoJoinDemoCluster	用户未加入体验网络。
InternalError.ServiceError	服务异常，请重试。
InternalError.ServicePanic	交易服务异常，请重试。
InternalError.UnknownError	交易服务未知错误，请重试。
InvalidParameter.InputDataViolation	输入参数存在违规内容。
InvalidParameterValue.IllegalValue	请求参数错误。

查询长安链体验网络指定高度区块的交易

最近更新时间：2025-04-25 01:54:51

1. 接口描述

接口请求域名：tbaas.tencentcloudapi.com。

查询长安链体验网络指定高度区块的交易

默认接口请求频率限制：1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：QueryChainMakerDemoBlockTransaction。
Version	是	String	公共参数 ，本接口取值：2018-04-16。
Region	是	String	公共参数 ，详见产品支持的 地域列表 ，本接口仅支持其中的：ap-beijing。
ClusterId	是	String	网络ID，可在区块链网络详情或列表中获取 示例值：chainmaker-demo
ChainId	是	String	业务链ID，可在网络概览页获取 示例值：chain_test
BlockHeight	是	Integer	区块高度 示例值：1

3. 输出参数

参数名称	类型	描述
Result	Array of ChainMakerTransactionResult	区块交易
BlockHeight	Integer	区块高度 示例值：1
TxCount	Integer	交易数量 示例值：1
BlockTimestamp	Integer	区块时间戳，单位是秒 示例值：1625572262
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 查询长安链体验网络指定高度区块的交易

查询长安链体验网络指定高度区块的交易

输入示例

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

<公共请求参数>

```
{
  "BlockHeight": "2",
  "ChainId": "chain_test",
  "ClusterId": "chainmaker-demo"
}
```

输出示例

```
{
  "Response": {
    "RequestId": "0c64831e-a310-47c7-bf6c-a625c5cfde17",
    "BlockHeight": 2,
    "TxCount": 1,
    "BlockTimestamp": 1627267176,
    "Result": [
      {
        "Code": 0,
        "CodeMessage": "",
        "Message": "",
        "TxId": "f6cf450f9b0b496f9ee6d332e739b8b182fe1e3b34ed446ab068a151f1ee8cbe",
        "Timestamp": 1627267173,
        "GasUsed": 230871,
        "BlockHeight": 2,
        "ContractEvent": ""
      }
    ]
  }
}
```

5. 开发者资源

腾讯云 API 平台

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

API Inspector

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

SDK

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

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

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

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

错误码	描述
AuthFailure	CAM签名/鉴权错误。
FailedOperation.BaaSStopServing	为带来更好的服务与体验，长安链体验网络维护升级中，请稍后再试。
FailedOperation.InvalidOperation	用户非法操作。
FailedOperation.UserInBlackList	您因违反用户协议，目前无法使用长安链体验网络。
FailedOperation.UserNoJoinDemoCluster	用户未加入体验网络。
InternalError.ServiceError	服务异常，请重试。
InternalError.ServicePanic	交易服务异常，请重试。
InternalError.UnknownError	交易服务未知错误，请重试。
InvalidParameterValue.IllegalValue	请求参数错误。

调用长安链体验网络合约查询

最近更新时间：2025-04-25 01:54:51

1. 接口描述

接口请求域名：tbaas.tencentcloudapi.com。

调用长安链体验网络合约查询

默认接口请求频率限制：1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：QueryChainMakerDemoContract。
Version	是	String	公共参数 ，本接口取值：2018-04-16。
Region	是	String	公共参数 ，详见产品支持的 地域列表 ，本接口仅支持其中的：ap-beijing。
ClusterId	是	String	网络ID，可在区块链网络详情或列表中获得 示例值：chainmaker-demo
ChainId	是	String	业务链ID，可在网络概览页获取 示例值：chain_test
ContractName	是	String	合约名称，可在合约管理中获取 示例值：fact
FuncName	是	String	合约方法名 示例值：send
FuncParam	否	String	合约方法入参，json格式字符串，key/value都是string类型的map 示例值：{"name":"test"}

3. 输出参数

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

4. 示例

示例1 调用长安链体验网络合约查询

调用长安链体验网络合约查询

输入示例

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

<公共请求参数>

```
{
  "ChainId": "chain_test",
  "ClusterId": "chainmaker-demo",
  "ContractName": "fact",
  "FuncParam": "{\"file_hash\":\"1234567\"}",
  "FuncName": "find_by_file_hash"
}
```

输出示例

```
{
  "Response": {
    "RequestId": "fac6ef5a-7637-4a75-a2d3-0f7d87b5b8b5",
    "Result": {
      "Code": 0,
      "CodeMessage": "OK",
      "Message": "",
      "TxId": "3ae794a658af4a219f849f5891d7a390eae524de4d614339ace2411059aaa19c",
      "GasUsed": 6100540,
      "Result": "b2s="
    }
  }
}
```

5. 开发者资源

腾讯云 API 平台

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

API Inspector

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

SDK

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

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

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

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

错误码	描述
AuthFailure	CAM签名/鉴权错误。

错误码	描述
FailedOperation.BaaSStopServing	为带来更好的服务与体验，长安链体验网络维护升级中，请稍后再试。
FailedOperation.InvalidOperation	用户非法操作。
FailedOperation.UserInBlackList	您因违反用户协议，目前无法使用长安链体验网络。
FailedOperation.UserNoJoinDemoCluster	用户未加入体验网络。
InternalError.ServiceError	服务异常，请重试。
InternalError.ServicePanic	交易服务异常，请重试。
InternalError.UnknownError	交易服务未知错误，请重试。
InvalidParameterValue.IllegalValue	请求参数错误。

通过交易ID查询长安链体验网络交易

最近更新时间：2025-04-25 01:54:51

1. 接口描述

接口请求域名：tbaas.tencentcloudapi.com。

通过交易ID查询长安链体验网络交易

默认接口请求频率限制：1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：QueryChainMakerDemoTransaction。
Version	是	String	公共参数 ，本接口取值：2018-04-16。
Region	是	String	公共参数 ，详见产品支持的 地域列表 ，本接口仅支持其中的：ap-beijing。
ClusterId	是	String	网络ID，可在区块链网络详情或列表中获取 示例值：chainmaker-demo
ChainId	是	String	业务链ID，可在网络概览页获取 示例值：chain_test
TxID	是	String	交易ID，通过调用合约的返回值获取 示例值：HHHHHHHHHHHHHHHH

3. 输出参数

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

4. 示例

示例1 通过交易ID查询长安链体验网络交易

QueryChainMakerDemoTransaction

输入示例

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

{
  "ChainId": "chain_test",
  "ClusterId": "chainmaker-demo",
```

```
"TxId": "f6cf450f9b0b496f9ee6d332e739b8b182fe1e3b34ed446ab068a151f1ee8cbe"
}
```

输出示例

```
{
  "Response": {
    "RequestId": "d25c24d9-4bb9-44bc-b084-5a25af34bb14",
    "Result": {
      "Code": 0,
      "CodeMessage": "",
      "Message": "",
      "TxId": "f6cf450f9b0b496f9ee6d332e739b8b182fe1e3b34ed446ab068a151f1ee8cbe",
      "Timestamp": 1627267173,
      "GasUsed": 230871,
      "BlockHeight": 2,
      "ContractEvent": ""
    }
  }
}
```

5. 开发者资源

腾讯云 API 平台

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

API Inspector

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

SDK

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

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

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

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

错误码	描述
AuthFailure	CAM签名/鉴权错误。
FailedOperation.BaaSStopServing	为带来更好的服务与体验，长安链体验网络维护升级中，请稍后再试。
FailedOperation.InvalidOperation	用户非法操作。
FailedOperation.UserInBlackList	您因违反用户协议，目前无法使用长安链体验网络。
FailedOperation.UserNoJoinDemoCluster	用户未加入体验网络。

错误码	描述
InternalError.ServiceError	服务异常，请重试。
InternalError.ServicePanic	交易服务异常，请重试。
InternalError.UnknownError	交易服务未知错误，请重试。
InvalidParameterValue.IllegalValue	请求参数错误。

长安链相关接口

批量申请长安链用户签名证书

最近更新时间：2025-04-25 01:54:51

1. 接口描述

接口请求域名：tbaas.tencentcloudapi.com。

批量申请长安链用户签名证书

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer

<> 点击调试

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

2. 输入参数

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

参数名称	必选	类型	描述
Action	是	String	公共参数 ，本接口取值：ApplyChainMakerBatchUserCert。
Version	是	String	公共参数 ，本接口取值：2018-04-16。
Region	是	String	公共参数 ，详见产品支持的 地域列表 ，本接口仅支持其中的：ap-beijing。
ClusterId	是	String	网络ID，可在区块链网络详情或列表中获取 示例值：chainmaker-xxxxxxxxx
SignUserCsrList.N	是	Array of SignCertCsr	证书标识和证书请求文件，可参考TBaaS证书生成相关文档生成证书请求文件

3. 输出参数

参数名称	类型	描述
SignUserCrtList	Array of String	成功生成的用户证书的base64编码字符串列表，与SignUserCsrList一一对应 示例值：["Y2VydDE="]
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 批量申请长安链用户签名证书

输入示例

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

{
  "ClusterId": "chainmaker-itehp6pqt3",
  "SignUserCsrList": [
    {
      "CertMark": "user1",
```

输出示例

5. 开发者资源

腾讯云 API 平台

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

API Inspector

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

SDK

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

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

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

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

错误码	描述
AuthFailure	CAM签名/鉴权错误。
FailedOperation.BaaSStopServing	为带来更好的服务与体验，长安链体验网络维护升级中，请稍后再试。
FailedOperation.InvalidOperation	用户非法操作。
FailedOperation.UserInBlackList	您因违反用户协议，目前无法使用长安链体验网络。
FailedOperation.UserNoJoinDemoCluster	用户未加入体验网络。
InternalError.ServiceError	服务异常，请重试。
InternalError.ServicePanic	交易服务异常，请重试。
InternalError.UnknownError	交易服务未知错误，请重试。
InvalidParameterValue.IllegalValue	请求参数错误。

调用长安链合约执行交易

最近更新时间：2025-04-25 01:54:50

1. 接口描述

接口请求域名：tbaas.tencentcloudapi.com。

调用长安链合约执行交易

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：InvokeChainMakerContract。
Version	是	String	公共参数 ，本接口取值：2018-04-16。
Region	是	String	公共参数 ，详见产品支持的 地域列表 ，本接口仅支持其中的：ap-beijing。
ClusterId	是	String	网络ID，可在区块链网络详情或列表中获取 示例值：251005746bc987bojjue8
ChainId	是	String	业务链ID，可在网络概览页获取 示例值：chain-test
ContractName	是	String	合约名称，可在合约管理中获取 示例值：fact
FuncName	是	String	合约方法名 示例值：send
FuncParam	否	String	合约方法入参，json格式字符串，key/value都是string类型的map 示例值：{"name":"test"}
AsyncFlag	否	Integer	是否异步执行，1为是，否则为0；如果异步执行，可使用返回值中的交易TxID查询执行结果 示例值：1

3. 输出参数

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

4. 示例

示例1 调用长安链合约执行交易

调用长安链合约执行交易

输入示例

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

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

{
  "ChainId": "chain_dcwtt",
  "FuncParam": "{\\\"time\\\":\\\"2021-07-26\\\",\\\"file_hash\\\":\\\"123456\\\",\\\"file_name\\\":\\\"test.jpg\\\"}",
  "ClusterId": "chainmaker-j1juwsmndn",
  "FuncName": "save",
  "AsyncFlag": "0",
  "ContractName": "fact"
}
```

输出示例

```
{
  "Response": {
    "RequestId": "2974e7ac-72e2-49c3-ae39-bc2bc6a7330b",
    "Result": {
      "Code": 0,
      "CodeMessage": "OK",
      "Message": "",
      "TxId": "963b8210f10e45e6ab6fe5cd37ba7f6719db5948a38c494ca9137fa2b8c9033d",
      "GasUsed": 9573483,
      "Result": "b2s="
    }
  }
}
```

5. 开发者资源

腾讯云 API 平台

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

API Inspector

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

SDK

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

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

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

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

错误码	描述
AuthFailure.UnauthorizedOperation	用户无权限访问。
FailedOperation.BcosService	Bcos网络异常。
FailedOperation.CompileDeployedContract	Bcos不能编译已经部署的合约。
FailedOperation.CompileDeployingContract	Bcos不能编译正在部署的合约。
FailedOperation.ContractEditedByOtherAgency	Bcos只能由该合约所属机构执行操作。
FailedOperation.DeleteDeployedContract	Bcos合约已部署，无法删除。
FailedOperation.DeleteDeployingContract	Bcos合约正在部署，无法删除。
FailedOperation.DeployContractNotCompile	Bcos不能部署未编译的合约。
FailedOperation.FrontRequestFail	Bcos前置服务调用失败。
FailedOperation.InvalidGroupPk	Bcos无效的群组编号。
FailedOperation.InvalidKeyUser	Bcos无效的私钥用户信息。
FailedOperation.NewContract	Bcos新建合约失败。
FailedOperation.NotDeployedContract	Bcos合约未部署。
InternalError.DBError	Bcos数据库操作异常，请重试。
InternalError.InvalidContractParam	Bcos无效的合约参数。
InternalError.MethodTypeNotSupport	Bcos不支持的请求类型。
InternalError.ServerError	内部错误，请稍后重试或者联系技术人员
InternalError.ServiceError	服务异常，请重试。
InvalidParameter.AccountParamError	Bcos账号参数错误。
InvalidParameter.AgencyInvalid	Bcos无效的机构信息。
InvalidParameter.AgencyNetParamInvalid	Bcos新增机构网络关联信息入参错误。
InvalidParameter.AllianceIdOfNetEmpty	Bcos所属联盟编号不能为空。
InvalidParameter.ContractIdInvalid	Bcos无效的合约编号。
InvalidParameter.DataHadExist	Bcos数据已存在，请勿重复添加。
InvalidParameter.EmptyParam	Bcos无效参数。
InvalidParameter.ExpireTimeInvalid	Bcos无效的失效时间。
InvalidParameter.FrontConnFail	Bcos前置的IP或端口无效。
InvalidParameter.FrontIpInvalid	Bcos IP格式错误。
InvalidParameter.FrontParamError	Bcos新增前置服务参数错误。
InvalidParameter.FrontRequestFail	Bcos前置服务调用失败。
InvalidParameter.InvalidContractArg	Bcos无效的合约参数。
InvalidParameter.NetIdInvalid	Bcos无效的网络编号。
InvalidParameter.NetParamError	Bcos网络参数错误。
InvalidParameter.NewKeyUserParamError	Bcos新增私钥用户参数错误。
InvalidParameter.NoInfoToDelete	Bcos无可删信息,请确认后重试。
InvalidParameter.NotFoundValidFront	Bcos没有有效的前置服务信息。
InvalidParameter.ReDeployedContract	Bcos不能重复部署合约。

错误码	描述
InvalidParameter.ReDeployingContract	Bcos合约正在部署中，请勿重复操作。
InvalidParameter.RoleInvalid	Bcos无效的角色。
InvalidParameterValue.IllegalValue	请求参数错误。
MissingParameter.EmptyParam	Bcos缺少参数。
ResourceNotFound.EmptyData	Bcos查询不到数据。

查询长安链指定高度区块的交易

最近更新时间：2025-04-25 01:54:50

1. 接口描述

接口请求域名：tbaas.tencentcloudapi.com。

查询长安链指定高度区块的交易

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：QueryChainMakerBlockTransaction。
Version	是	String	公共参数 ，本接口取值：2018-04-16。
Region	是	String	公共参数 ，详见产品支持的 地域列表 ，本接口仅支持其中的: ap-beijing。
ClusterId	是	String	网络ID，可在区块链网络详情或列表中获取 示例值：251005746bc987bojjue8
ChainId	是	String	业务链ID，可在网络概览页获取 示例值：chain-test
BlockHeight	是	Integer	区块高度 示例值：1

3. 输出参数

参数名称	类型	描述
Result	Array of ChainMakerTransactionResult	区块交易
BlockHeight	Integer	区块高度 示例值：1
TxCount	Integer	交易数量 示例值：1
BlockTimestamp	Integer	区块时间戳，单位是秒 示例值：1625572262
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 查询长安链指定高度区块的交易

查询长安链指定高度区块的交易

输入示例

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

<公共请求参数>

```
{
  "BlockHeight": 2,
  "ChainId": "chain_dcwtt",
  "ClusterId": "chainmaker-j1juwsmndn"
}
```

输出示例

```
{
  "Response": {
    "RequestId": "0c64831e-a310-47c7-bf6c-a625c5cfde17",
    "BlockHeight": 2,
    "TxCount": 1,
    "BlockTimestamp": 1627267176,
    "Result": [
      {
        "Code": 0,
        "CodeMessage": "",
        "Message": "",
        "TxId": "f6cf450f9b0b496f9ee6d332e739b8b182fe1e3b34ed446ab068a151f1ee8cbe",
        "Timestamp": 1627267173,
        "GasUsed": 230871,
        "BlockHeight": 2,
        "ContractEvent": ""
      }
    ]
  }
}
```

5. 开发者资源

腾讯云 API 平台

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

API Inspector

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

SDK

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

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

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

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

错误码	描述
AuthFailure.UnauthorizedOperation	用户无权限访问。
FailedOperation.BcosService	Bcos网络异常。
FailedOperation.CompileDeployedContract	Bcos不能编译已经部署的合约。
FailedOperation.CompileDeployingContract	Bcos不能编译正在部署的合约。
FailedOperation.ContractEditedByOtherAgency	Bcos只能由该合约所属机构执行操作。
FailedOperation.DeleteDeployedContract	Bcos合约已部署，无法删除。
FailedOperation.DeleteDeployingContract	Bcos合约正在部署，无法删除。
FailedOperation.DeployContractNotCompile	Bcos不能部署未编译的合约。
FailedOperation.FrontRequestFail	Bcos前置服务调用失败。
FailedOperation.InvalidGroupPk	Bcos无效的群组编号。
FailedOperation.InvalidKeyUser	Bcos无效的私钥用户信息。
FailedOperation.NewContract	Bcos新建合约失败。
FailedOperation.NotDeployedContract	Bcos合约未部署。
InternalError.DBError	Bcos数据库操作异常，请重试。
InternalError.InvalidContractParam	Bcos无效的合约参数。
InternalError.MethodTypeNotSupport	Bcos不支持的请求类型。
InternalError.ServerError	内部错误，请稍后重试或者联系技术人员
InternalError.ServiceError	服务异常，请重试。
InternalError.ServicePanic	交易服务异常，请重试。
InternalError.UnknownError	交易服务未知错误，请重试。
InvalidParameter.AccountParamError	Bcos账号参数错误。
InvalidParameter.AgencyInvalid	Bcos无效的机构信息。
InvalidParameter.AgencyNetParamInvalid	Bcos新增机构网络关联信息入参错误。
InvalidParameter.AllianceIdOfNetEmpty	Bcos所属联盟编号不能为空。
InvalidParameter.ContractIdInvalid	Bcos无效的合约编号。
InvalidParameter.DataHadExist	Bcos数据已存在，请勿重复添加。
InvalidParameter.EmptyParam	Bcos无效参数。
InvalidParameter.ExpireTimeInvalid	Bcos无效的失效时间。
InvalidParameter.FrontConnFail	Bcos前置的IP或端口无效。
InvalidParameter.FrontIpInvalid	Bcos IP格式错误。
InvalidParameter.FrontParamError	Bcos新增前置服务参数错误。
InvalidParameter.FrontRequestFail	Bcos前置服务调用失败。

错误码	描述
InvalidParameter.InvalidContractArg	Bcos无效的合约参数。
InvalidParameter.NetIdInvalid	Bcos无效的网络编号。
InvalidParameter.NetParamError	Bcos网络参数错误。
InvalidParameter.NewKeyUserParamError	Bcos新增私钥用户参数错误。
InvalidParameter.NoInfoToDelete	Bcos无可删信息,请确认后重试。
InvalidParameter.NotFoundValidFront	Bcos没有有效的前置服务信息。
InvalidParameter.ReDeployedContract	Bcos不能重复部署合约。
InvalidParameter.ReDeployingContract	Bcos合约正在部署中，请勿重复操作。
InvalidParameter.RoleInvalid	Bcos无效的角色。
InvalidParameterValue.IllegalValue	请求参数错误。
MissingParameter.EmptyParam	Bcos缺少参数。
ResourceNotFound.EmptyData	Bcos查询不到数据。

调用长安链合约查询

最近更新时间：2025-04-25 01:54:50

1. 接口描述

接口请求域名：tbaas.tencentcloudapi.com。

调用长安链合约查询

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：QueryChainMakerContract。
Version	是	String	公共参数 ，本接口取值：2018-04-16。
Region	是	String	公共参数 ，详见产品支持的 地域列表 ，本接口仅支持其中的：ap-beijing。
ClusterId	是	String	网络ID，可在区块链网络详情或列表中获取 示例值：251005746bc987bojjue8
ChainId	是	String	业务链ID，可在网络概览页获取 示例值：chain-test
ContractName	是	String	合约名称，可在合约管理中获取 示例值：fact
FuncName	是	String	合约方法名 示例值：send
FuncParam	否	String	合约方法入参，json格式字符串，key/value都是string类型的map 示例值：{"name":"test"}

3. 输出参数

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

4. 示例

示例1 调用长安链合约查询

调用长安链合约查询

输入示例

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

<公共请求参数>

```
{
  "ChainId": "chain_dcwtt",
  "ClusterId": "chainmaker-j1juwsmndn",
  "ContractName": "fact",
  "FuncParam": "{\"file_hash\":\"1234567\"}",
  "FuncName": "find_by_file_hash"
}
```

输出示例

```
{
  "Response": {
    "RequestId": "fac6ef5a-7637-4a75-a2d3-0f7d87b5b8b5",
    "Result": {
      "Code": 0,
      "CodeMessage": "OK",
      "Message": "",
      "TxId": "3ae794a658af4a219f849f5891d7a390eae524de4d614339ace2411059aaa19c",
      "GasUsed": 6100540,
      "Result": "b2s="
    }
  }
}
```

5. 开发者资源

腾讯云 API 平台

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

API Inspector

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

SDK

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

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

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

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

错误码	描述
AuthFailure.UnauthorizedOperation	用户无权限访问。

错误码	描述
FailedOperation.BcosService	Bcos网络异常。
FailedOperation.CompileDeployedContract	Bcos不能编译已经部署的合约。
FailedOperation.CompileDeployingContract	Bcos不能编译正在部署的合约。
FailedOperation.ContractEditedByOtherAgency	Bcos只能由该合约所属机构执行操作。
FailedOperation.DeleteDeployedContract	Bcos合约已部署，无法删除。
FailedOperation.DeleteDeployingContract	Bcos合约正在部署，无法删除。
FailedOperation.DeployContractNotCompile	Bcos不能部署未编译的合约。
FailedOperation.FrontRequestFail	Bcos前置服务调用失败。
FailedOperation.InvalidGroupPk	Bcos无效的群组编号。
FailedOperation.InvalidKeyUser	Bcos无效的私钥用户信息。
FailedOperation.NewContract	Bcos新建合约失败。
FailedOperation.NotDeployedContract	Bcos合约未部署。
InternalError.DBError	Bcos数据库操作异常，请重试。
InternalError.InvalidContractParam	Bcos无效的合约参数。
InternalError.MethodTypeNotSupport	Bcos不支持的请求类型。
InternalError.ServerError	内部错误，请稍后重试或者联系技术人员
InternalError.ServiceError	服务异常，请重试。
InvalidParameter.AccountParamError	Bcos账号参数错误。
InvalidParameter.AgencyInvalid	Bcos无效的机构信息。
InvalidParameter.AgencyNetParamInvalid	Bcos新增机构网络关联信息入参错误。
InvalidParameter.AllianceIdOfNetEmpty	Bcos所属联盟编号不能为空。
InvalidParameter.ContractIdInvalid	Bcos无效的合约编号。
InvalidParameter.DataHadExist	Bcos数据已存在，请勿重复添加。
InvalidParameter.EmptyParam	Bcos无效参数。
InvalidParameter.ExpireTimeInvalid	Bcos无效的失效时间。
InvalidParameter.FrontConnFail	Bcos前置的IP或端口无效。
InvalidParameter.FrontIpInvalid	Bcos IP格式错误。
InvalidParameter.FrontParamError	Bcos新增前置服务参数错误。
InvalidParameter.FrontRequestFail	Bcos前置服务调用失败。
InvalidParameter.InvalidContractArg	Bcos无效的合约参数。
InvalidParameter.NetIdInvalid	Bcos无效的网络编号。
InvalidParameter.NetParamError	Bcos网络参数错误。
InvalidParameter.NewKeyUserParamError	Bcos新增私钥用户参数错误。
InvalidParameter.NoInfoToDelete	Bcos无可删信息,请确认后重试。
InvalidParameter.NotFoundValidFront	Bcos没有有效的前置服务信息。
InvalidParameter.ReDeployedContract	Bcos不能重复部署合约。
InvalidParameter.ReDeployingContract	Bcos合约正在部署中，请勿重复操作。

错误码	描述
InvalidParameter.RoleInvalid	Bcos无效的角色。
InvalidParameterValue.IllegalValue	请求参数错误。
MissingParameter.EmptyParam	Bcos缺少参数。
ResourceNotFound.EmptyData	Bcos查询不到数据。

通过交易ID查询长安链交易

最近更新时间：2025-04-25 01:54:50

1. 接口描述

接口请求域名：tbaas.tencentcloudapi.com。

通过交易ID查询长安链交易

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：QueryChainMakerTransaction。
Version	是	String	公共参数 ，本接口取值：2018-04-16。
Region	是	String	公共参数 ，详见产品支持的 地域列表 ，本接口仅支持其中的：ap-beijing。
ClusterId	是	String	网络ID，可在区块链网络详情或列表中获取 示例值：251005746bc987bojjue8
ChainId	是	String	业务链ID，可在网络概览页获取 示例值：chain-test
TxID	是	String	交易ID，通过调用合约的返回值获取 示例值：HHHHHHHHHHHHHHHH

3. 输出参数

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

4. 示例

示例1 通过交易ID查询

通过交易ID查询

输入示例

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

{
  "ChainId": "chain_dcwtt",
  "ClusterId": "chainmaker-j1juwsmndn",
```

```
"TxId": "f6cf450f9b0b496f9ee6d332e739b8b182fe1e3b34ed446ab068a151f1ee8cbe"
}
```

输出示例

```
{
  "Response": {
    "RequestId": "d25c24d9-4bb9-44bc-b084-5a25af34bb14",
    "Result": {
      "Code": 0,
      "CodeMessage": "",
      "Message": "",
      "TxId": "f6cf450f9b0b496f9ee6d332e739b8b182fe1e3b34ed446ab068a151f1ee8cbe",
      "Timestamp": 1627267173,
      "GasUsed": 230871,
      "BlockHeight": 2,
      "ContractEvent": ""
    }
  }
}
```

5. 开发者资源

腾讯云 API 平台

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

API Inspector

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

SDK

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

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

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

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

错误码	描述
AuthFailure.UnauthorizedOperation	用户无权限访问。
FailedOperation.BcosService	Bcos网络异常。
FailedOperation.CompileDeployedContract	Bcos不能编译已经部署的合约。
FailedOperation.CompileDeployingContract	Bcos不能编译正在部署的合约。
FailedOperation.ContractEditedByOtherAgency	Bcos只能由该合约所属机构执行操作。

错误码	描述
FailedOperation.DeleteDeployedContract	Bcos合约已部署，无法删除。
FailedOperation.DeleteDeployingContract	Bcos合约正在部署，无法删除。
FailedOperation.DeployContractNotCompile	Bcos不能部署未编译的合约。
FailedOperation.FrontRequestFail	Bcos前置服务调用失败。
FailedOperation.InvalidGroupPk	Bcos无效的群组编号。
FailedOperation.InvalidKeyUser	Bcos无效的私钥用户信息。
FailedOperation.NewContract	Bcos新建合约失败。
FailedOperation.NotDeployedContract	Bcos合约未部署。
InternalError.DBError	Bcos数据库操作异常，请重试。
InternalError.InvalidContractParam	Bcos无效的合约参数。
InternalError.MethodTypeNotSupport	Bcos不支持的请求类型。
InternalError.ServerError	内部错误，请稍后重试或者联系技术人员
InternalError.ServiceError	服务异常，请重试。
InvalidParameter.AccountParamError	Bcos账号参数错误。
InvalidParameter.AgencyInvalid	Bcos无效的机构信息。
InvalidParameter.AgencyNetParamInvalid	Bcos新增机构网络关联信息入参错误。
InvalidParameter.AlliancedOfNetEmpty	Bcos所属联盟编号不能为空。
InvalidParameter.ContractIdInvalid	Bcos无效的合约编号。
InvalidParameter.DataHadExist	Bcos数据已存在，请勿重复添加。
InvalidParameter.EmptyParam	Bcos无效参数。
InvalidParameter.ExpireTimeInvalid	Bcos无效的失效时间。
InvalidParameter.FrontConnFail	Bcos前置的IP或端口无效。
InvalidParameter.FrontIpInvalid	Bcos IP格式错误。
InvalidParameter.FrontParamError	Bcos新增前置服务参数错误。
InvalidParameter.FrontRequestFail	Bcos前置服务调用失败。
InvalidParameter.InvalidContractArg	Bcos无效的合约参数。
InvalidParameter.NetIdInvalid	Bcos无效的网络编号。
InvalidParameter.NetParamError	Bcos网络参数错误。
InvalidParameter.NewKeyUserParamError	Bcos新增私钥用户参数错误。
InvalidParameter.NoInfoToDelete	Bcos无可删信息,请确认后重试。
InvalidParameter.NotFoundValidFront	Bcos没有有效的前置服务信息。
InvalidParameter.ReDeployedContract	Bcos不能重复部署合约。
InvalidParameter.ReDeployingContract	Bcos合约正在部署中，请勿重复操作。
InvalidParameter.RoleInvalid	Bcos无效的角色。
InvalidParameterValue.IllegalValue	请求参数错误。
MissingParameter.EmptyParam	Bcos缺少参数。
ResourceNotFound.EmptyData	Bcos查询不到数据。

Hyperledger Fabric 2.3 相关接口

获取Fabric某区块的详细信息

最近更新时间：2025-04-25 01:54:52

1. 接口描述

接口请求域名：tbaas.tencentcloudapi.com。

获取Fabric某区块的详细信息

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer

<> 点击调试

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

2. 输入参数

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

参数名称	必选	类型	描述
Action	是	String	公共参数 ，本接口取值：DescribeFabricBlock。
Version	是	String	公共参数 ，本接口取值：2018-04-16。
Region	是	String	公共参数 ，详见产品支持的 地域列表 ，本接口仅支持其中的：ap-guangzhou, ap-hongkong。
ClusterId	是	String	网络ID，可在区块链网络详情或列表中获取 示例值：fabric-65z42qi150
ChannelId	是	String	通道ID，可在通道列表或通道详情获取 示例值：channel-9xej4d
BlockHeight	是	Integer	区块高度，从0开始 示例值：4

3. 输出参数

参数名称	类型	描述
BlockHeight	Integer	区块高度 示例值：1024
BlockHash	String	区块Hash 示例值：90af8cff95cfc2b9caeafc9d53bc04a699ef47e381143d66e5d03548deab1aa5
PreBlockHash	String	前置区块Hash 示例值：1d40d9c1fc8929c0b5ba4048f6a756543695664976cec7c8ac76e587587dfa01
TxCount	Integer	区块中交易数量 示例值：100
TransactionList	Array of Transaction	区块中交易列表
CreateTimestamp	String	创建时间戳 示例值：2023-08-05 07:44:04
ProposerOrg	String	提案组织 示例值：testorg
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 DescribeFabricBlock

DescribeFabricBlock

输入示例

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

```
{
  "ClusterId": "fabric-65z42qi150",
  "ChannelId": "channel-9xej4d",
  "BlockHeight": 4
}
```

输出示例

```
{
  "Response": {
    "BlockHash": "e0421ecc20340425a2cf01808469b2523651f8100da92a0287b3769115e57370",
    "BlockHeight": 4,
    "CreateTimestamp": "2023-08-05 07:44:04",
    "PreBlockHash": "f69b2b5a1884e5bbb077a3025378e2210b83bbada23b5c17045b0bf537a53bd0",
    "ProposerOrg": "区块链测试机构pettychen",
    "RequestId": "a6288649-01c0-4d8f-bdd4-ba02163d0721",
    "TransactionList": [
      {
        "BlockHeight": 4,
        "ChaincodeName": "_lifecycle",
        "CreateTime": "2023-08-05 15:44:04",
        "Sender": "区块链测试机构",
        "TxId": "03389f4a5af1883347ecdb15471e70abf3a8e6cf5b6648dc789674af7930e229"
      }
    ],
    "TxCount": 1
  }
}
```

5. 开发者资源

腾讯云 API 平台

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

API Inspector

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

SDK

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

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

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

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

错误码	描述
FailedOperation.FabricBlockNoExist	区块不存在。
FailedOperation.InvalidAuth	用户无权限访问。
InternalError.ServerError	内部错误，请稍后重试或者联系技术人员

获取Fabric交易的详细信息

最近更新时间：2025-04-25 01:54:52

1. 接口描述

接口请求域名：tbaas.tencentcloudapi.com。

获取Fabric交易的详细信息

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer

</> 点击调试

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

2. 输入参数

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

参数名称	必选	类型	描述
Action	是	String	公共参数 ，本接口取值：DescribeFabricTransaction。
Version	是	String	公共参数 ，本接口取值：2018-04-16。
Region	是	String	公共参数 ，详见产品支持的 地域列表 ，本接口仅支持其中的：ap-guangzhou, ap-hongkong。
ClusterId	是	String	网络ID，可在区块链网络详情或列表中获取 示例值：fabric-65z42qi150
ChannelId	是	String	通道ID，可在通道列表或通道详情获取 示例值：channel-9xej4d
TxId	是	String	交易ID 示例值：03389f4a5af1883347ecdb15471e70abf3a8e6cf5b6648dc789674af7930e229

3. 输出参数

参数名称	类型	描述
TxId	String	交易ID 示例值：03389f4a5af1883347ecdb15471e70abf3a8e6cf5b6648dc789674af7930e229
TxHash	String	交易Hash 示例值：8e5b978d33cfbc0510b4e60406f8481f8ec00e390131d5b6ea6d38604e1bcb46
TxStatus	String	交易状态 示例值：VALID
JoinOrgList	Array of String	参与的组织列表 示例值：["org-xxxx.fabric-ian2bqtylk"]
Sender	String	交易发送者 示例值：org-xxxx.fabric-ian2bqtylk
CreateTime	String	创建时间 示例值：2023-07-07 10:10:10
BlockHeight	Integer	区块高度 示例值：1024
ChaincodeName	String	交易所属合约 示例值：fact
TransactionData	String	交易数据，base64编码，解码后为json化的字符串 示例值：eyJjaGFpbmNvZGVfc3BIYyl6eyJ0aW1lb3V0ljowLCJ0eXBlljoiVU5ERUZJTkVEIn19

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 DescribeFabricTransaction

DescribeFabricTransaction

输入示例

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

{
  "ClusterId": "fabric-65z42qi150",
  "ChannelId": "channel-9xej4d",
  "TxId": "03389f4a5af1883347ecdb15471e70abf3a8e6cf5b6648dc789674af7930e229"
}
```

输出示例

```
{
  "Response": {
    "BlockHeight": 4,
    "ChaincodeName": "_lifecycle",
    "CreateTime": "2023-08-05 15:44:04",
    "JoinOrgList": [
      "org-xxxx.fabric-ian2bqtylk"
    ],
    "RequestId": "b5221353-8f20-4547-8260-c550b4b0175c",
    "Sender": "org-xxxx.fabric-ian2bqtylk",
    "TransactionData": "eyJjaGFpbmNvZGVfc3BlYyI6eyJjaGFpbmNvZGVfaWQiOnsibmFtZSI6I19saWZlY3ljbjGUiLCJwYXRoIjoiIiwidmVyc2lvbiI6IiJ9LCJpbmB1dCI6eyJhcmdzIjpbIkJNvbW1pdENoYWluY29kZURlZmluaXRpb24iLCJedTAwMDhcdTAwMDFcdTAwMTJcdTAwMDNteWNcdTAwMWFcdTAwMDR2MS4wXCJedTAwMDRlc2NjKlxiMDAwNHZzY2MyMVxuL1xiMDAwMlxiMDAwOFxiMDAwMlxiMDAwNlxiMDAwOFxiMDAwMVxiMDAwMlxiMDAwMlxiMDAwOFxiMDAwMFxiMDAwYSNcdTAwMTIhXG5cdTAwMWZvcmtctMjUxMDA1NzQ2LmZhYnJpYy02NXo0MnFpMTUwO1xiMDAwMCI6JkZWNvcmtF0aW9ucyI6e30sIm1zX2luaXQiOmZhbHN1fSwidGltZW91dCI6MCwidHlwZSI6I1VOREVGSU5FRFCJ9fQ==",
    "TxHash": "8e5b978d33cfbc0510b4e60406f8481f8ec00e390131d5b6ea6d38604e1bcb46",
    "TxId": "03389f4a5af1883347ecdb15471e70abf3a8e6cf5b6648dc789674af7930e229",
    "TxStatus": "VALID"
  }
}
```

5. 开发者资源

腾讯云 API 平台

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

API Inspector

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

SDK

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

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

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

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

错误码	描述
FailedOperation.FabricTransactionNotExist	交易不存在。
FailedOperation.InvalidAuth	用户无权限访问。
InternalServerError.ServerError	内部错误，请稍后重试或者联系技术人员

调用Fabric用户合约执行交易

最近更新时间：2025-04-25 01:54:51

1. 接口描述

接口请求域名：tbaas.tencentcloudapi.com。

调用Fabric用户合约执行交易

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：InvokeFabricChaincode。
Version	是	String	公共参数 ，本接口取值：2018-04-16。
Region	是	String	公共参数 ，详见产品支持的 地域列表 ，本接口仅支持其中的：ap-guangzhou, ap-hongkong。
ClusterId	是	String	网络ID，可在区块链网络详情获取 示例值：fabric-65z42qi150
ChannelId	是	String	通道ID，可在通道列表或通道详情获取 示例值：channel-9xej4d
ChaincodeName	是	String	合约名称，可在合约列表或合约详情获取 示例值：fact
FuncName	是	String	合约方法 示例值：Set
FuncParam.N	否	Array of String	合约方法入参 示例值：["key","value"]
WithAsyncResult	否	Boolean	是否异步执行，如果异步执行，可使用返回值中的交易TxID查询执行结果 示例值：false

3. 输出参数

参数名称	类型	描述
TxId	String	交易ID 示例值：03389f4a5af1883347ecdb15471e70abf3a8e6cf5b6648dc789674af7930e229
TxStatus	String	交易状态 示例值：VALID
TxResult	String	交易结果 示例值：value
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 InvokeFabricChaincode

InvokeFabricChaincode

输入示例

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

{
  "ClusterId": "fabric-65z42qi150",
  "ChannelId": "channel-9xej4d",
  "ChaincodeName": "fact",
  "FuncName": "Set",
  "FuncParam": [
    "D",
    "40"
  ],
  "WithAsyncResult": false
}
```

输出示例

```
{
  "Response": {
    "RequestId": "bab100f3-be40-415c-b29b-3a44c77b25fe",
    "TxId": "0595a33e308170b035e0d97c1a79274ac55badf048c0c8fc569719e14a2990c5",
    "TxResult": "",
    "TxStatus": "VALID"
  }
}
```

5. 开发者资源

腾讯云 API 平台

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

API Inspector

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

SDK

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

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

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

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

错误码	描述
FailedOperation.FabricChaincodeInvokeFailed	合约调用失败。
FailedOperation.FabricChaincodeNotExist	合约不存在。
FailedOperation.InvalidAuth	用户无权限访问。
InternalError.ServerError	内部错误，请稍后重试或者联系技术人员

调用Fabric用户合约查询

最近更新时间：2025-04-25 01:54:51

1. 接口描述

接口请求域名：tbaas.tencentcloudapi.com。

调用Fabric用户合约查询

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：QueryFabricChaincode。
Version	是	String	公共参数 ，本接口取值：2018-04-16。
Region	是	String	公共参数 ，详见产品支持的 地域列表 ，本接口仅支持其中的：ap-guangzhou, ap-hongkong。
ClusterId	是	String	网络ID，可在区块链网络详情获取 示例值：fabric-65z42qi150
ChannelId	是	String	通道ID，可在通道列表或通道详情获取 示例值：channel-9xej4d
ChaincodeName	是	String	合约名称，可在合约列表或合约详情获取 示例值：fact
FuncName	是	String	合约方法 示例值：Get
FuncParam.N	否	Array of String	合约方法入参 示例值：["key"]

3. 输出参数

参数名称	类型	描述
TxId	String	交易ID 示例值：03389f4a5af1883347ecdb15471e70abf3a8e6cf5b6648dc789674af7930e229
TxStatus	String	交易状态 示例值：VALID
TxResult	String	交易结果 示例值：value
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 QueryFabricChaincode

QueryFabricChaincode

输入示例

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

```
{
  "ClusterId": "fabric-65z42qi150",
  "ChannelId": "channel-9xej4d",
  "ChaincodeName": "fact",
  "FuncName": "Get",
  "FuncParam": [
    "A"
  ]
}
```

输出示例

```
{
  "Response": {
    "RequestId": "e431d04c-2a5c-43b4-a480-f127b3f46eb6",
    "TxId": "030acb112f9f2aa2ce537fea9d94d2f3c9ec6cf8e05aa62f5d3746167825dbe8",
    "TxResult": "10",
    "TxStatus": "VALID"
  }
}
```

5. 开发者资源

腾讯云 API 平台

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

API Inspector

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

SDK

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

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

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

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

错误码	描述
FailedOperation.FabricChaincodeNoExist	合约不存在。
FailedOperation.FabricChaincodeQueryFailed	合约查询失败。
FailedOperation.InvalidAuth	用户无权限访问。
InternalError.ServerError	内部错误，请稍后重试或者联系技术人员

Hyperledger Fabric 1.4 相关接口

申请用户证书

最近更新时间：2025-04-25 01:54:53

1. 接口描述

接口请求域名：tbaas.tencentcloudapi.com。

申请用户证书

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer

<> 点击调试

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

2. 输入参数

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

参数名称	必选	类型	描述
Action	是	String	公共参数 ，本接口取值：ApplyUserCert。
Version	是	String	公共参数 ，本接口取值：2018-04-16。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
Module	是	String	模块名，固定字段：cert_mng 示例值：cert_mng
Operation	是	String	操作名，固定字段：cert_apply_for_user 示例值：cert_apply_for_user
ClusterId	是	String	区块链网络ID，可在区块链网络详情或列表中获取 示例值：251005746bckuobc41mpu
GroupName	是	String	申请证书的组织名称，可以在组织管理列表中获取当前组织的名称 示例值：youtuOrg
UserIdentity	是	String	用户证书标识，用于标识用户证书，要求由纯小写字母组成，长度小于10 示例值：usercontent
Applicant	是	String	证书申请实体，使用腾讯云账号实名认证的名称 示例值：优图SDK
IdentityNum	是	String	证件号码。如果腾讯云账号对应的实名认证类型为企业认证，填入“0”；如果腾讯云账号对应的实名认证类型为个人认证，填入个人身份证号码 示例值：3452***321
CsrData	是	String	csr p10证书文件。需要用户根据文档生成证书的CSR文件 示例值：-----BEGINCERTIFICATEREQUEST-----
Notes	否	String	证书备注信息 示例值：user

3. 输出参数

参数名称	类型	描述
CertId	Integer	证书ID 示例值：172634
CertDn	String	证书DN 示例值：C=CN,ST=ShenZhen,L=4ShenZhen,O=org1,OU=client,CN=kylouse@org1.bcdk3eis17qe@client

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 用户证书申请

输入示例

```
https://tbaas.tencentcloudapi.com/?Action=ApplyUserCert
&Module=cert_mng
&Operation=cert_apply_for_user
&ClusterId=251005746bckuobc41mpu
&GroupName=youtuOrg
&UserIdentity=kyloz
&Applicant=优图SDK
&IdentityNum=3503*****2342
&CsrData=-----BEGINCERTIFICATEREQUEST-----
MIIBSTCB8AIBADCBjTELMakGA1UEBhMCQ04xEjAQBgNVBAgMCUd1YW5nWmhvdTER
MA8GA1UEBwwIU2h1blpoZW4xPDASBgNVBAoMC2V4YW1wbGUuY29tMQ8wDQYDVQQL
DAZjbG11bnQxMDAuBgNVBAMMJ0FkbWluLm9yZzJAb3JnMi5leGFtcGx1LmNvbSx0
eXB1PWNsaWVudDBZMBMGByqGSM49AgEGCCqGSM49AwEHA0IABMIrWeLZjivbTLwy
z+JMfmZjPEJ8x4H8a/CnqBIwcCaJjg5cMqOi+VyFHkCM9Sv4MFOqPAK7rE62m9XF
s4B8hfKgADAKBgqghkJOQQDAGNIADBFAiEAtL/h0DHfRJ1170U6OGpCkN34jX0a
GGGSHbFu2KxLUHsCIAFxXa6UsQOf93vT0FQ1deBodbVCbNhM8/jtVqfZ/lhB
-----ENDCERTIFICATEREQUEST-----
&Notes=fact
&<公共请求参数>
```

输出示例

```
{
  "Response": {
    "CertDn": "C=CN, ST=ShenZhen, L=4ShenZhen, O=org1, OU=client, CN=kylouse@org1.bcdk3eis17qe@client",
    "CertId": 172634,
    "RequestId": "16824370-6058-4f38-a32d-c19b1eaeafae"
  }
}
```

5. 开发者资源

腾讯云 API 平台

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

API Inspector

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

SDK

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

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

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

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

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

错误码	描述
AuthFailure.UnauthorizedOperation	用户无权限访问。
FailedOperation.CaCheckCsr	检查CSR文件非法。
FailedOperation.CaDbOption	数据库操作错误。
FailedOperation.CaExsit	证书已经存在。
FailedOperation.CaGenkey	产生密钥对失败。
FailedOperation.Calnit	初始化CA失败。
FailedOperation.CalInputParam	输入参数非法。
FailedOperation.CaNoExist	证书不存在。
FailedOperation.CaRevoke	注销证书失败。
FailedOperation.CaRootNonExist	根证书不存在。
FailedOperation.CaService	CA内部错误。
FailedOperation.CaSignCert	签发证书失败。
FailedOperation.CaYunApiApplyCert	CFCA云API申请证书失败。
FailedOperation.CaYunApiCommon	CFCA云API错误。
FailedOperation.GroupIllegal	用户非法操作。
FailedOperation.UserAuthType	获取用户认证类型出错。
InternalServerError.FlaskException	Flask内部错误。
InternalServerError.NoDefineError	错误码未定义。
InternalServerError.ServerException	服务器异常。
InvalidParameterValue.IllegalFormat	参数格式不正确。
InvalidParameterValue.IllegalValue	请求参数错误。
InvalidParameterValue.ParameterEmpty	参数值为空。

下载用户证书

最近更新时间：2025-05-27 01:58:19

1. 接口描述

接口请求域名：tbaas.tencentcloudapi.com。

下载用户证书

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer

</> 点击调试

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

2. 输入参数

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

参数名称	必选	类型	描述
Action	是	String	公共参数 ，本接口取值：DownloadUserCert。
Version	是	String	公共参数 ，本接口取值：2018-04-16。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
Module	是	String	模块名，固定字段：cert_mng 示例值：cert_mng
Operation	是	String	操作名，固定字段：cert_download_for_user 示例值：cert_download_for_user
CertId	是	Integer	证书ID，可以在证书详情页面获取 示例值：171781
CertDn	是	String	证书DN，可以在证书详情页面获取 示例值：name
ClusterId	是	String	区块链网络ID，可在区块链网络详情或列表中获取 示例值：251005746bckuobc41mpu
GroupName	是	String	下载证书的组织名称，可以在组织管理列表中获取当前组织的名称 示例值：youtuOrg

3. 输出参数

参数名称	类型	描述
CertName	String	证书名称 示例值：kyloz@youtuorg.bckuobc41mpu@client.pem
Cert	String	证书内容 示例值：-----BEGIN CERTIFICATE----- MIIC4DCCAoagAwIBAgIMx6T1QUi9YviE6cpGfQyIQIJ+SAp/6u5Ehgzjglgaf9eX823ZRWLg+S7CdWSk/oVVeSoelyUDY -----END CERTIFICATE-----
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 RequestId。

4. 示例

示例1 用户证书下载

输入示例

```
https://tbaas.tencentcloudapi.com/?Action=DownloadUserCert
&Module=cert_mng
&Operation=cert_download_for_user
&CertId=171781
&CertDn=C=CN,ST=ShenZhen,L=4ShenZhen,O=youtuOrg,OU=client,CN=kyloz@youtuorg,bckuobc41mpu@client
&ClusterId=251005746bckuobc41mpu
&GroupName=youtuOrg
&<公共请求参数>
```

输出示例

```
{
  "Response": {
    "Cert": "-----BEGIN CERTIFICATE-----\nMIIC4DCCAoagAwIBAgIBBzAKBggqhkJOPQQAjBpMQswCQYDVQQGEwJDTjERMA8G\n4xETAPBgNVBACMFNoZW5aaGVuMREwDwYDVQQDAh5b3V0\ndU9yZzEhMB8GA1UEAwYY2EueW91dHVvcmcuYmNrdW9iYzQxbXB1MB4XDTE5MTAx\nNV0XDTIyMDkzMDA2NDY0NVowGyUxZzAJBgNVBAYTAkNOMREwDwYDVQQTI\nnEwhTaGVuWmhlbjESMBAGA1UEBxMJNFNoZW5aaGVuMREwDwYDVQQKEWh5b3V0dU9y\n\nZzEPMA0GA1UECzMzY2pZW50MSswKQYDVQDFCJreWxvekB5b3V0dW9yZy5iY2t1\nnb2JjNDFTcHVAY2xpZW50MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQg\nAEWitZ4tm0\nkN9tMvDLP4kx+ZmM8QnzHgfr8KeoEjBwJomODlwyo6L5XIUEQIz1K/gwU6o8Arus\nnTrab1cWzgHyF8qOCAQAwgf0wCQYDVROTBAlwADARBglg\nhkgBhvhCAQEEBAMCBkAw\nnHQYDVRO0BBYEFNeW9zH7j/rxDk1NBz4OB5aWVY66MIGbBgNVHSMegZMwgZCAFDdP\nn92Wahmf0Pabh45gYaozILIVooW2kazBpMQ\nswCQYDVQQGEwJDTjERMA8GA1UECAwI\nnU2hlblpoZW4xETAPBgNVBACMFNoZW5aaGVuMREwDwYDVQQDAh5b3V0dU9yZzEh\n\nMB8GA1UEAwYY2EueW91dHVvcmcuYmNrdW9iYzQxbXB1ggkA+Nat6Mct1AowCwYD\nnVR0PBAQDAgWgMBMGA1UdJQQMMAoGCCsGAQUFBwMBMAoGCCqGSM49BAMCA0gAMEUC\n\nnIQDquRCXmx6T1Q\nUi9YviE6cpGfQyIQIj+SAp/6u5EhgZjgIgaF9eX823ZRWLG+S7\nnCdwSk/oVVeSoelyUDYqTvYYqXtI=\n-----END CERTIFICATE-----\n",
    "CertName": "kyloz@youtuorg.bckuobc41mpu@client.pem",
    "RequestId": "5f6192d9-fbcd-4d1d-9b52-61006f33543b"
  }
}
```

5. 开发者资源

腾讯云 API 平台

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

API Inspector

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

SDK

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

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

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

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

错误码	描述
AuthFailure.UnauthorizedOperation	用户无权限访问。
FailedOperation.CaCheckCsr	检查CSR文件非法。
FailedOperation.CaDbOption	数据库操作错误。
FailedOperation.CaExsit	证书已经存在。
FailedOperation.CaGenkey	产生密钥对失败。
FailedOperation.Calnit	初始化CA失败。
FailedOperation.CaInputParam	输入参数非法。
FailedOperation.CaNoExist	证书不存在。
FailedOperation.CaRevoke	注销证书失败。
FailedOperation.CaRootNonExist	根证书不存在。
FailedOperation.CaService	CA内部错误。
FailedOperation.CaSignCert	签发证书失败。
FailedOperation.CaYunApiApplyCert	CFCA云API申请证书失败。
FailedOperation.CaYunApiCommon	CFCA云API错误。
FailedOperation.GroupIllegal	用户非法操作。
InternalError.FlaskException	Flask内部错误。
InternalError.NoDefineError	错误码未定义。
InternalError.ServerException	服务器异常。
InvalidParameterValue.IllegalFormat	参数格式不正确。
InvalidParameterValue.IllegalValue	请求参数错误。
InvalidParameterValue.ParameterEmpty	参数值为空。

查询fabric的区块列表

最近更新时间：2025-04-25 01:54:53

1. 接口描述

接口请求域名：tbaas.tencentcloudapi.com。

查看当前网络下的所有区块列表，分页展示

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：GetBlockList。
Version	是	String	公共参数 ，本接口取值：2018-04-16。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
Module	是	String	模块名称，固定字段：block 示例值：block
Operation	是	String	操作名称，固定字段：block_list 示例值：block_list
ChannelId	是	Integer	通道ID，固定字段：0 示例值：12
GroupId	是	Integer	组织ID，固定字段：0 示例值：32
ChannelName	是	String	需要查询的通道名称，可在通道详情或列表中获取 示例值：factch
GroupName	是	String	调用接口的组织名称，可以在组织管理列表中获取当前组织的名称 示例值：liulanOrg
ClusterId	是	String	区块链网络ID，可在区块链网络详情或列表中获取 示例值：251005746bc0f03q8u93j
Offset	否	Integer	需要获取的起始交易偏移 示例值：0
Limit	否	Integer	需要获取的交易数量 示例值：10

3. 输出参数

参数名称	类型	描述
TotalCount	Integer	区块数量 示例值：6
BlockList	Array of Block	区块列表
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 查询所有区块列表

输入示例

```
https://tbaas.tencentcloudapi.com/?Action=GetBlockList
&Module=block
&Operation=block_list
&ChannelId=0
&GroupId=0
&ChannelName=kylotst
&GroupName=liulanOrg
&ClusterId=251005746bc0f03q8u93j
&Offset=0
&Limit=10
&<公共请求参数>
```

输出示例

```
{
  "Response": {
    "BlockList": [
      {
        "BlockId": 5,
        "BlockNum": 5,
        "DataHash": "92f1c3b1eb0eb1bc1825f1acd3474e7679b2a97cdf8d6155ed6a0e6c1458e479",
        "PreHash": "92f1c3b1eb0eb1bc1825f1acd3474e7679b2a97cdf8d6155ed6a0e6c1458e479",
        "TxCount": 1
      },
      {
        "BlockId": 4,
        "BlockNum": 4,
        "DataHash": "17e381e65605963c9211abc5b72d80cf3a6a4955ff7b61c70b406b98c90ded6f",
        "PreHash": "17e381e65605963c9211abc5b72d80cf3a6a4955ff7b61c70b406b98c90ded6f",
        "TxCount": 1
      },
      {
        "BlockId": 3,
        "BlockNum": 3,
        "DataHash": "32b80e93141467eda367f05a2428a6c03369405ae10349db7fad0762e9b56cc2",
        "PreHash": "32b80e93141467eda367f05a2428a6c03369405ae10349db7fad0762e9b56cc2",
        "TxCount": 1
      },
      {
        "BlockId": 2,
        "BlockNum": 2,
        "DataHash": "8be5094bf40c5b7e410cc3e8aa33354da2aa78db862e17cfd5fa783dddc1ee3a",
        "PreHash": "8be5094bf40c5b7e410cc3e8aa33354da2aa78db862e17cfd5fa783dddc1ee3a",
        "TxCount": 1
      },
      {
        "BlockId": 1,
        "BlockNum": 1,
        "DataHash": "af319fe2ff9e5830af0f07200adf965156446a205137e61d87f1db0b8f43cc7e",
        "PreHash": "af319fe2ff9e5830af0f07200adf965156446a205137e61d87f1db0b8f43cc7e",
        "TxCount": 1
      }
    ]
  }
}
```



```
},
{
  "BlockId": 0,
  "BlockNum": 0,
  "DataHash": "96f22d00947478c1b16c2aec5e7079c761b168d304cadf19a14a032ee8f64a23",
  "PreHash": "96f22d00947478c1b16c2aec5e7079c761b168d304cadf19a14a032ee8f64a23",
  "TxCount": 1
}
],
"RequestId": "018328c8-9c24-4104-a0ad-a7a31c033278",
"TotalCount": 6
}
}
```

5. 开发者资源

腾讯云 API 平台

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

API Inspector

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

SDK

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

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

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

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

错误码	描述
AuthFailure.UnauthorizedOperation	用户无权限访问。
FailedOperation.FabricBlockQuery	区块列表查询失败。
FailedOperation.FabricRequestParams	参数验证失败。
FailedOperation.GroupIllegal	用户非法操作。
FailedOperation.NoChannelPeer	节点没有加入通道。
FailedOperation.NoObject	对象不存在。
FailedOperation.TimeOutUrl	后台服务请求超时。
InternalError.FlaskException	Flask 内部错误。
InternalError.NoDefineError	错误码未定义。
InternalError.ServerException	服务器异常。

错误码	描述
InvalidParameterValue.IllegalFormat	参数格式不正确。
InvalidParameterValue.IllegalValue	请求参数错误。
InvalidParameterValue.ParameterEmpty	参数值为空。

获取区块内交易列表

最近更新时间：2025-04-25 01:54:53

1. 接口描述

接口请求域名：tbaas.tencentcloudapi.com。

获取区块内交易列表

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：GetBlockTransactionListForUser。
Version	是	String	公共参数 ，本接口取值：2018-04-16。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
Module	是	String	模块名，固定字段：transaction 示例值：transaction
Operation	是	String	操作名，固定字段：block_transaction_list_for_user 示例值：block_transaction_list_for_user
ClusterId	是	String	区块链网络ID，可在区块链网络详情或列表中获取 示例值：251005746bcdk3eis17qe
GroupName	是	String	参与交易的组织名称，可以在组织管理列表中获取当前组织的名称 示例值：org1
ChannelName	是	String	业务所属通道名称，可在通道详情或列表中获取 示例值：fronttest3
BlockId	是	Integer	区块ID，通过GetInvokeTx接口可以获取交易所在的区块ID 示例值：2
Offset	否	Integer	查询的交易列表起始偏移地址 示例值：0
Limit	否	Integer	查询的交易列表数量 示例值：10

3. 输出参数

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

4. 示例

示例1 获取区块内的交易列表

输入示例

```
https://tbaas.tencentcloudapi.com/?Action=GetBlockTransactionListForUser
&Module=transaction
&Operation=block_transaction_list_for_user
&ClusterId=251005746bcdk3eis17qe
&GroupName=org1
&ChannelName=fronttest3
&BlockId=2
&Offset=0
&Limit=10
&<公共请求参数>
```

输出示例

```
{
  "Response": {
    "RequestId": "4697d721-ef88-4dba-96b2-5c9143773a6f",
    "TotalCount": 1,
    "TransactionList": [
      {
        "BlockHeight": 3,
        "BlockId": 2,
        "CreateOrgName": "org1",
        "CreateTime": "2020-01-06 22:18:15",
        "TransactionHash": "b5e398182b07bbde523e63804ee24690323bd5e190db99da079a00af2d71b652",
        "TransactionId": "0af5b0ae129b0435aa4ec8547c7500f428673dee363331f2f5f8d61d49e43e68",
        "TransactionStatus": "VALID",
        "TransactionType": "ENDORSE_TRANSACTION"
      }
    ]
  }
}
```

5. 开发者资源

腾讯云 API 平台

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

API Inspector

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

SDK

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

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

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

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

错误码	描述
AuthFailure.UnauthorizedOperation	用户无权限访问。
FailedOperation.DatabaseException	数据库操作异常。
FailedOperation.FabricBlockDetail	区块详情查询失败。
FailedOperation.FabricTransactionDetail	交易详情查询失败。
FailedOperation.FabricTransactionQuery	交易列表查询失败。
FailedOperation.GroupIllegal	用户非法操作。
FailedOperation.NoChannelPeer	节点没有加入通道。
FailedOperation.NoObject	对象不存在。
FailedOperation.NoPeer	没有可用节点。
FailedOperation.StatusNoMatch	操作状态不匹配。
FailedOperation.TimeOutUrl	后台服务请求超时。
InternalError.FlaskException	Flask内部错误。
InternalError.NoDefineError	错误码未定义。
InternalError.ServerException	服务器异常。
InvalidParameterValue	参数取值错误。
InvalidParameterValue.IllegalFormat	参数格式不正确。
InvalidParameterValue.IllegalValue	请求参数错误。
InvalidParameterValue.ParameterEmpty	参数值为空。

获取区块链网络概要

最近更新时间：2025-04-25 01:54:53

1. 接口描述

接口请求域名：tbaas.tencentcloudapi.com。

获取区块链网络概要

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：GetClusterSummary。
Version	是	String	公共参数 ，本接口取值：2018-04-16。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
Module	是	String	模块名称，固定字段：cluster_mng 示例值：cluster_mng
Operation	是	String	操作名称，固定字段：cluster_summary 示例值：cluster_summary
ClusterId	是	String	区块链网络ID，可在区块链网络详情或列表中获取 示例值：251005746bc0f03q8u93j
GroupId	是	Integer	组织ID，固定字段：0 示例值：12
GroupName	是	String	调用接口的组织名称，可以在组织管理列表中获取当前组织的名称 示例值：liulanOrg

3. 输出参数

参数名称	类型	描述
TotalChannelCount	Integer	网络通道总数量 示例值：1
MyChannelCount	Integer	当前组织创建的通道数量 示例值：1
JoinChannelCount	Integer	当前组织加入的通道数量 示例值：1
TotalPeerCount	Integer	网络节点总数量 示例值：2
MyPeerCount	Integer	当前组织创建的节点数量 示例值：2
OrderCount	Integer	其他组织创建的节点数量 示例值：0
TotalGroupCount	Integer	网络组织总数量 示例值：1

参数名称	类型	描述
MyGroupCount	Integer	当前组织创建的组织数量 示例值：1
TotalChaincodeCount	Integer	网络智能合约总数量 示例值：1
RecentChaincodeCount	Integer	最近7天发起的智能合约数量 示例值：1
MyChaincodeCount	Integer	当前组织发起的智能合约数量 示例值：1
TotalCertCount	Integer	当前组织的证书总数量 示例值：8
TlsCertCount	Integer	颁发给当前组织的证书数量 示例值：4
PeerCertCount	Integer	网络背书节点证书数量 示例值：2
ClientCertCount	Integer	当前组织业务证书数量 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 GetClusterSummary

输入示例

```
https://tbaas.tencentcloudapi.com/?Action=GetClusterSummary
&Module=cluster_mng
&Operation=cluster_summary
&ClusterId=251005746bc0f03q8u93j
&GroupId=12
&GroupName=liulanOrg
&<公共请求参数>
```

输出示例

```
{
  "Response": {
    "MyPeerCount": 1,
    "ClientCertCount": 1,
    "TotalChaincodeCount": 1,
    "TotalChannelCount": 1,
    "TlsCertCount": 1,
    "OrderCount": 1,
    "RecentChaincodeCount": 1,
    "PeerCertCount": 1,
    "MyChaincodeCount": 1,
    "RequestId": "aeae4ed4-a4ce-4ffd-9826-1aae7669bb7c",
    "TotalCertCount": 1,
    "MyGroupCount": 1,
    "MyChannelCount": 1,
    "TotalGroupCount": 1,
    "JoinChannelCount": 1,
```

```
"TotalPeerCount": 1
}
```

5. 开发者资源

腾讯云 API 平台

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

API Inspector

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

SDK

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

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

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

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

错误码	描述
AuthFailure.UnauthorizedOperation	用户无权限访问。
FailedOperation.GroupIllegal	用户非法操作。
FailedOperation.NoObject	对象不存在。
FailedOperation.TimeOutUrl	后台服务请求超时。
InternalServerError.FlaskException	Flask内部错误。
InternalServerError.NoDefineError	错误码未定义。
InternalServerError.ServerException	服务器异常。
InvalidParameterValue.IllegalFormat	参数格式不正确。
InvalidParameterValue.IllegalValue	请求参数错误。
InvalidParameterValue.ParameterEmpty	参数值为空。

获取fabric最新交易列表

最近更新时间：2025-04-25 01:54:52

1. 接口描述

接口请求域名：tbaas.tencentcloudapi.com。

获取fabric最新交易列表

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：GetLatestTransactionList。
Version	是	String	公共参数 ，本接口取值：2018-04-16。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
Module	是	String	模块名称，固定字段：transaction 示例值：transaction
Operation	是	String	操作名称，固定字段：latest_transaction_list 示例值：latest_transaction_list
GroupId	是	Integer	组织ID，固定字段：0 示例值：0
ChannelId	是	Integer	通道ID，固定字段：0 示例值：0
LatestBlockNumber	是	Integer	获取的最新交易的区块数量，取值范围1~5 示例值：5
GroupName	是	String	调用接口的组织名称，可以在组织管理列表中获得当前组织的名称 示例值：liulanOrg
ChannelName	是	String	需要查询的通道名称，可在通道详情或列表中获得 示例值：kylotst
ClusterId	是	String	区块链网络ID，可在区块链网络详情或列表中获得 示例值：251005746bc0f03q8u93j
Offset	否	Integer	需要获取的起始交易偏移 示例值：0
Limit	否	Integer	需要获取的交易数量 示例值：10

3. 输出参数

参数名称	类型	描述
TotalCount	Integer	交易总数量 示例值：5
TransactionList	Array of TransactionItem	交易列表

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

4. 示例

示例1 GetLatestTransactionList

获取fabric最新交易列表

输入示例

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

{
  "Module": "transaction",
  "Operation": "latest_transaction_list",
  "GroupId": 0,
  "ChannelId": 0,
  "Offset": 0,
  "Limit": 5,
  "LatestBlockNumber": 5,
  "GroupName": "junyutest1",
  "ChannelName": "testch1",
  "ClusterId": "251005746bc8h0o128ocy"
}
```

输出示例

```
{
  "Response": {
    "RequestId": "9b8d5384-d3ff-49ca-9795-3b9bb4205c4c",
    "TotalCount": 1,
    "TransactionList": [
      {
        "BlockHeight": 1,
        "BlockId": 0,
        "CreateOrgName": "junyutest1",
        "CreateTime": "2023-06-15 10:49:00",
        "TransactionHash": "",
        "TransactionId": "ef6b21d46c88ce74f95eda336d553f80735131587857dbb04f9b00623eb38f65",
        "TransactionStatus": "VALID",
        "TransactionType": "CONFIG"
      }
    ]
  }
}
```

5. 开发者资源

腾讯云 API 平台

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

API Inspector

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

SDK

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

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

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

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

错误码	描述
AuthFailure.UnauthorizedOperation	用户无权限访问。
FailedOperation.FabricBlockDetail	区块详情查询失败。
FailedOperation.FabricTransactionQuery	交易列表查询失败。
FailedOperation.GroupIllegal	用户非法操作。
FailedOperation.NoChannelGroup	组织没有加入通道。
FailedOperation.NoChannelPeer	节点没有加入通道。
FailedOperation.NoObject	对象不存在。
InternalServerError.FlaskException	Flask内部错误。
InternalServerError.NoDefineError	错误码未定义。
InternalServerError.ServerException	服务器异常。
InvalidParameterValue.IllegalFormat	参数格式不正确。
InvalidParameterValue.IllegalValue	请求参数错误。
InvalidParameterValue.ParameterEmpty	参数值为空。

获取交易的详情

最近更新时间：2025-04-25 01:54:52

1. 接口描述

接口请求域名：tbaas.tencentcloudapi.com。

获取交易的详情

默认接口请求频率限制：5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：GetTransactionDetailForUser。
Version	是	String	公共参数 ，本接口取值：2018-04-16。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
Module	是	String	模块名，固定字段：transaction 示例值：transaction
Operation	是	String	操作名，固定字段：transaction_detail_for_user 示例值：transaction_detail_for_user
ClusterId	是	String	区块链网络ID，可在区块链网络详情或列表中获取 示例值：251005746bcd6xfka7pi
GroupName	是	String	参与交易的组织名称，可以在组织管理列表中获取当前组织的名称 示例值：org2
ChannelName	是	String	业务所属通道名称，可在通道详情或列表中获取 示例值：factch
BlockId	是	Integer	区块ID，通过GetInvokeTx接口可以获取交易所在的区块ID 示例值：1
TransactionId	是	String	交易ID，需要查询的详情的交易ID 示例值：42c562517707bf53f08fe6e5a1cf5f50c784352ff0d8166e77514ebe518bd6c0

3. 输出参数

参数名称	类型	描述
TransactionId	String	交易ID 示例值：42c562517707bf53f08fe6e5a1cf5f50c784352ff0d8166e77514ebe518bd6c0
TransactionHash	String	交易hash 示例值：42c562517707bf53f08fe6e5a1cf5f50c784352ff0d8166e77514ebe518bd6c0
CreateOrgName	String	创建交易的组织名 示例值：org2
TransactionType	String	交易类型（普通交易和配置交易） 示例值：CONFIG
TransactionStatus	String	交易状态 示例值：VALID

参数名称	类型	描述
CreateTime	String	交易创建时间 示例值：2019-10-28 18:48:28
TransactionData	String	交易数据 示例值：eyJjaGFpbmNvZGVfc3BIYyI6eyJ0aW1lb3V0IjowLCJ0eXBlljoiVU5ERUZJTkVEIn19
BlockId	Integer	交易所在区块号 示例值：1
BlockHash	String	交易所在区块哈希 示例值：a0f1bd5819e1e664873c3d903d005f9f24661986c2c4c541a185f613fdeaff52
BlockHeight	Integer	交易所在区块高度 示例值：1
ChannelName	String	通道名称 示例值：factch
ContractName	String	交易所在合约名称 示例值：fact
EndorserOrgList	Array of EndorserGroup	背书组织列表
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 获取交易详情

输入示例

```
https://tbaas.tencentcloudapi.com/?Action=GetTransactionDetailForUser
&Module=transaction
&Operation=transaction_detail_for_user
&ClusterId=251005746bcd6xjfk7pi
&GroupName=org2
&ChannelName=tyler
&BlockId=0
&TransactionId=42c562517707bf53f08fe6e5a1cf5f50c784352ff0d8166e77514ebe518bd6c0
&<公共请求参数>
```

输出示例

```
{
  "Response": {
    "BlockHash": "a0f1bd5819e1e664873c3d903d005f9f24661986c2c4c541a185f613fdeaff52",
    "BlockHeight": 1,
    "BlockId": 0,
    "ChannelName": "tyler",
    "ContractName": "fact",
    "CreateOrgName": "org2",
    "CreateTime": "2019-10-28 18:48:28",
    "EndorserOrgList": [
      {
        "EndorserGroupName": "org2",
        "EndorserPeerList": [
          "Admin.org2.bcd6xjfk7pi"
        ]
      }
    ]
  }
}
```

```
}
],
"RequestId": "a13c4aed-ba4a-4357-850a-66707d7d5e74",
"TransactionData": "{\\\"chaincode_spec\\\":{\\\"timeout\\\":0,\\\"type\\\":\\\"UNDEFINED\\\"}}",
"TransactionHash": "42c562517707bf53f08fe6e5a1cf5f50c784352ff0d8166e77514ebe518bd6c0",
"TransactionId": "42c562517707bf53f08fe6e5a1cf5f50c784352ff0d8166e77514ebe518bd6c0",
"TransactionStatus": "VALID",
"TransactionType": "CONFIG"
}
```

5. 开发者资源

腾讯云 API 平台

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

API Inspector

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

SDK

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

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

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

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

错误码	描述
AuthFailure.UnauthorizedOperation	用户无权限访问。
FailedOperation.DatabaseException	数据库操作异常。
FailedOperation.FabricBlockDetail	区块详情查询失败。
FailedOperation.FabricTransactionDetail	交易详情查询失败。
FailedOperation.GroupIllegal	用户非法操作。
FailedOperation.NoObject	对象不存在。
FailedOperation.TimeOutUrl	后台服务请求超时。
InternalServerError.FailUrl	服务器异常，请重试。
InternalServerError.FlaskException	Flask内部错误。
InternalServerError.NoDefineError	错误码未定义。
InternalServerError.ServerException	服务器异常。
InternalServerError.TransactionService	交易服务内部错误。

错误码	描述
InvalidParameterValue.IllegalFormat	参数格式不正确。
InvalidParameterValue.IllegalValue	请求参数错误。
InvalidParameterValue.ParameterEmpty	参数值为空。

新增交易

最近更新时间：2025-04-25 01:54:52

1. 接口描述

接口请求域名：tbaas.tencentcloudapi.com。

新增交易

默认接口请求频率限制：50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：Invoke。
Version	是	String	公共参数 ，本接口取值：2018-04-16。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
Module	是	String	模块名，固定字段：transaction 示例值：transaction
Operation	是	String	操作名，固定字段：invoke 示例值：invoke
ClusterId	是	String	区块链网络ID，可在区块链网络详情或列表中获取 示例值：251005746envnew
ChaincodeName	是	String	业务所属智能合约名称，可在智能合约详情或列表中获取 示例值：cc050301
ChannelName	是	String	业务所属通道名称，可在通道详情或列表中获取 示例值：ch042103
Peers.N	是	Array of PeerSet	对该笔交易进行背书的节点列表（包括节点名称和节点所属组织名称，详见数据结构一节），可以在通道详情中获取该通道上的节点名称及其所属组织名称 示例值：peer0.neworg02.envnew
FuncName	是	String	该笔交易需要调用的智能合约中的函数名称 示例值：createCar
GroupName	是	String	调用合约的组织名称，可以在组织管理列表中获取当前组织的名称 示例值：NewOrg02
Args.N	否	Array of String	被调用的函数参数列表，参数列表大小总和和要求小于2M 示例值：CAR92
AsyncFlag	否	Integer	同步调用标识，可选参数，值为0或者不传表示使用同步方法调用，调用后会等待交易执行后再返回执行结果；值为1时表示使用异步方式调用Invoke，执行后会立即返回交易对应的Txid，后续需要通过GetInvokeTx这个API查询该交易的执行结果。（对于逻辑较为简单的交易，可以使用同步模式；对于逻辑较为复杂的交易，建议使用异步模式，否则容易导致API因等待时间过长，返回等待超时） 示例值：0

3. 输出参数

参数名称	类型	描述
Events	String	交易执行结果 示例值：myOrgpeer0.myorg.envnew:VALID

参数名称	类型	描述
TxId	String	交易ID 示例值：280e9f1436c3ce045af4f3c7060ff217583585d41faf1f1daa99387419bac07c
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 创建汽车信息——同步模式

输入示例

```
https://tbaas.tencentcloudapi.com/?Action=Invoke
&Module=transaction
&Operation=invoke
&ClusterId=251005746envnew
&Peers.0.PeerName=peer0.neworg02.envnew
&Peers.0.OrgName=NewOrg02
&ChannelName=ch042103
&ChaincodeName=cc050301
&FuncName=createCar
&Args.0=CAR92
&Args.1=Chevy
&Args.2=Volt
&Args.3=Black
&Args.4=Nick
&GroupName=NewOrg02
&<公共请求参数>
```

输出示例

```
{
  "Response": {
    "Events": "myOrgpeer0.myorg.envnew:VALID",
    "RequestId": "0b82b65e-7100-49f1-9f29-e934a8833711",
    "TxId": "0366ab8f31c9f8aa6b9fc9506fa841e55d1ecd492b3ecc373c0f66ca49f33ea1"
  }
}
```

示例2 创建汽车信息——异步模式

输入示例

```
https://tbaas.tencentcloudapi.com/?Action=Invoke
&Module=transaction
&Operation=invoke
&ClusterId=251005746envnew
&Peers.0.PeerName=peer0.neworg02.envnew
&Peers.0.OrgName=NewOrg02
&ChannelName=ch042103
&ChaincodeName=cc050301
&FuncName=createCar
&Args.0=CAR92
&Args.1=Chevy
&Args.2=Volt
```

```
&Args.3=Black
&Args.4=Nick
&AsyncFlag=1
&GroupName=NewOrg02
&<公共请求参数>
```

输出示例

```
{
  "Response": {
    "Events": "",
    "RequestId": "51ad1047-30c0-4397-b672-37e8ffd65d9f",
    "TxId": "280e9f1436c3ce045af4f3c7060ff217583585d41faf1f1daa99387419bac07c"
  }
}
```

5. 开发者资源

腾讯云 API 平台

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

API Inspector

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

SDK

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

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

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

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

错误码	描述
AuthFailure.UnauthorizedOperation	用户无权限访问。
FailedOperation.DatabaseException	数据库操作异常。
FailedOperation.FabricCommit	提交失败。
FailedOperation.FabricEventHub	event hub失败。
FailedOperation.FabricProposal	背书失败。
FailedOperation.FabricRequestParams	fabric请求参数错误。
FailedOperation.FabricRequstParams	参数验证失败。
FailedOperation.FabricTxInvoke	fabric invoke交易错误。

错误码	描述
FailedOperation.FabricTxQuery	fabric query交易错误。
FailedOperation.FabricTxQueryNone	fabric query交易错误, 无正确。
FailedOperation.GroupIllegal	用户非法操作。
FailedOperation.ManageService	交易请求异常。
FailedOperation.NoChainCodeChannel	合约没有在通道初始化。
FailedOperation.NoChainCodeGroup	组织没有加入合约。
FailedOperation.NoChainCodePeer	合约没有在节点安装。
FailedOperation.NoChannelGroup	组织没有加入通道。
FailedOperation.NoChannelPeer	节点没有加入通道。
FailedOperation.NoObject	对象不存在。
FailedOperation.NoPeer	没有可用节点。
FailedOperation.TransactionTimeout	交易执行超时, 请稍后再试。
InternalError.FlaskException	Flask内部错误。
InternalError.ServicePanic	交易服务异常, 请重试。
InternalError.TransactionService	交易服务内部错误。
InternalError.UnknownError	交易服务未知错误, 请重试。
InvalidParameterValue.IllegalValue	请求参数错误。
InvalidParameterValue.ParameterEmpty	参数值为空。

查询交易

最近更新时间：2025-04-25 01:54:52

1. 接口描述

接口请求域名：tbaas.tencentcloudapi.com。

查询交易

默认接口请求频率限制：50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：Query。
Version	是	String	公共参数 ，本接口取值：2018-04-16。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
Module	是	String	模块名，固定字段：transaction 示例值：transaction
Operation	是	String	操作名，固定字段：query 示例值：query
ClusterId	是	String	区块链网络ID，可在区块链网络详情或列表中获取 示例值：251005746envnew
ChaincodeName	是	String	业务所属智能合约名称，可在智能合约详情或列表中获取 示例值：cc050301
ChannelName	是	String	业务所属通道名称，可在通道详情或列表中获取 示例值：ch042103
Peers.N	是	Array of PeerSet	执行该查询交易的节点列表（包括节点名称和节点所属组织名称，详见数据结构一节），可以在通道详情中获取该通道上的节点名称及其所属组织名称 示例值：peer0.neworg02.envnew
FuncName	是	String	该笔交易查询需要调用的智能合约中的函数名称 示例值：queryCar
GroupName	是	String	调用合约的组织名称，可以在组织管理列表中获取当前组织的名称 示例值：NewOrg02
Args.N	否	Array of String	被调用的函数参数列表 示例值：CAR92

3. 输出参数

参数名称	类型	描述
Data	Array of String	查询结果数据 示例值：{"make":"Chevy","model":"Volt","colour":"Black","ow
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 查询汽车信息

输入示例

```
https://tbaas.tencentcloudapi.com/?Action=Query
&Module=transaction
&Operation=query
&ClusterId=251005746envnew
&Peers.0.PeerName=peer0.neworg02.envnew
&Peers.0.OrgName=NewOrg02
&ChannelName=ch042103
&ChaincodeName=cc050301
&FuncName=queryCar
&Args.0=CAR92
&GroupName=NewOrg02
&<公共请求参数>
```

输出示例

```
{
  "Response": {
    "Data": [
      "{\"make\":\"Chevy\",\"model\":\"Volt\",\"colour\":\"Black\",\"owner\":\"Nick\"}"
    ],
    "RequestId": "3f6836c5-e889-431e-b932-47a1653c5f7b"
  }
}
```

5. 开发者资源

腾讯云 API 平台

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

API Inspector

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

SDK

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

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

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

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

错误码	描述
AuthFailure.UnauthorizedOperation	用户无权限访问。
FailedOperation.DatabaseException	数据库操作异常。
FailedOperation.FabricCommit	提交失败。
FailedOperation.FabricProposal	背书失败。
FailedOperation.FabricRequestParams	fabric请求参数错误。
FailedOperation.FabricRequestParams	参数验证失败。
FailedOperation.FabricTxInvoke	fabric invoke交易错误。
FailedOperation.FabricTxQuery	fabric query交易错误。
FailedOperation.FabricTxQueryNone	fabric query交易错误, 无正确。
FailedOperation.GroupIllegal	用户非法操作。
FailedOperation.ManageService	交易请求异常。
FailedOperation.NoChainCodeChannel	合约没有在通道初始化。
FailedOperation.NoChainCodeGroup	组织没有加入合约。
FailedOperation.NoChainCodePeer	合约没有在节点安装。
FailedOperation.NoChannelGroup	组织没有加入通道。
FailedOperation.NoChannelPeer	节点没有加入通道。
FailedOperation.NoObject	对象不存在。
FailedOperation.TransactionTimeout	交易执行超时, 请稍后再试。
InternalError.FlaskException	Flask内部错误。
InternalError.ServicePanic	交易服务异常, 请重试。
InternalError.TransactionService	交易服务内部错误。
InternalError.UnknownError	交易服务未知错误, 请重试。
InvalidParameterValue.IllegalValue	请求参数错误。
InvalidParameterValue.ParameterEmpty	参数值为空。

Invoke异步调用结果查询

最近更新时间：2025-05-21 01:29:55

1. 接口描述

接口请求域名：tbaas.tencentcloudapi.com。

Invoke异步调用结果查询

默认接口请求频率限制：50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：GetInvokeTx。
Version	是	String	公共参数 ，本接口取值：2018-04-16。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
Module	是	String	模块名，固定字段：transaction 示例值：transaction
Operation	是	String	操作名，固定字段：query_txid 示例值：query_txid
ClusterId	是	String	区块链网络ID，可在区块链网络详情或列表中获取 示例值：251005746envnew
ChannelName	是	String	业务所属通道名称，可在通道详情或列表中获取 示例值：ch042103
PeerName	是	String	执行该查询交易的节点名称，可以在通道详情中获取该通道上的节点名称及其所属组织名称 示例值：peer0.neworg02.envnew
PeerGroup	是	String	执行该查询交易的节点所属组织名称，可以在通道详情中获取该通道上的节点名称及其所属组织名称 示例值：NewOrg02
TxId	是	String	交易ID 示例值：280e9f1436c3ce045af4f3c7060ff217583585d41faf1f1daa99387419bac07c
GroupName	是	String	调用合约的组织名称，可以在组织管理列表中获取当前组织的名称 示例值：NewOrg02

3. 输出参数

参数名称	类型	描述
TxValidationCode	Integer	交易执行状态码 示例值：0
TxValidationMsg	String	交易执行消息 示例值：VALID
BlockId	Integer	交易所在区块ID 示例值：6
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 查询异步模式下Invoke接口的调用结果

查询异步模式下Invoke接口的调用结果

输入示例

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

{
  "PeerGroup": "NewOrg02",
  "ClusterId": "251005746envnew",
  "PeerName": "peer0.neworg02.envnew",
  "Module": "transaction",
  "TxId": "280e9f1436c3ce045af4f3c7060ff217583585d41faf1f1daa99387419bac07c",
  "GroupName": "NewOrg02",
  "Operation": "query_txid",
  "ChannelName": "ch042103"
}
```

输出示例

```
{
  "Response": {
    "BlockId": 6,
    "RequestId": "551b801e-6dbe-46be-aa46-f8cc3ff1cd09",
    "TxValidationCode": 0,
    "TxValidationMsg": "VALID"
  }
}
```

5. 开发者资源

腾讯云 API 平台

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

API Inspector

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

SDK

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

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

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

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

错误码	描述
AuthFailure.UnauthorizedOperation	用户无权限访问。
FailedOperation.DatabaseException	数据库操作异常。
FailedOperation.FabricRequestParams	fabric请求参数错误。
FailedOperation.FabricRequstParams	参数验证失败。
FailedOperation.FabricTxIdQuery	Txid 查询失败。
FailedOperation.Grouplllegal	用户非法操作。
FailedOperation.ManageService	交易请求异常。
FailedOperation.NoChannelPeer	节点没有加入通道。
FailedOperation.NoObject	对象不存在。
FailedOperation.TransactionTimeout	交易执行超时，请稍后再试。
InternalError.FlaskException	Flask内部错误。
InternalError.ServicePanic	交易服务异常，请重试。
InternalError.TransactionService	交易服务内部错误。
InternalError.UnknownError	交易服务未知错误，请重试。
InvalidParameterValue.IllegalValue	请求参数错误。
InvalidParameterValue.ParameterEmpty	参数值为空。

数据结构

最近更新时间：2025-05-16 01:58:10

Block

区块对象

被如下接口引用：GetBlockList。

名称	类型	描述
BlockNum	Integer	区块编号 示例值：1
DataHash	String	区块数据Hash数值 示例值：6a3f18885016cafea3380732d8d867cdb655799135c466f1cd4a0456291fed13
BlockId	Integer	区块ID，与区块编号一致 示例值：1
PreHash	String	前一个区块Hash 示例值：1d40d9c1fc8929c0b5ba4048f6a756543695664976cec7c8ac76e587587dfa01
TxCount	Integer	区块内的交易数量 示例值：1

ChainMakerContractResult

长安链合约执行结果

被如下接口引用：InvokeChainMakerContract, InvokeChainMakerDemoContract, QueryChainMakerContract, QueryChainMakerDemoContract。

名称	类型	描述
Code	Integer	交易结果码 示例值：0
CodeMessage	String	交易结果码含义 示例值：Success
TxId	String	交易ID 示例值：9d3c4eabca704b04b9317583a194c5b7163aeb3c44e5418f98f79a4a319efb92
GasUsed	Integer	Gas使用量 示例值：0
Message	String	合约返回消息 示例值：Success
Result	String	合约函数返回，base64编码 示例值：dGVzdHRlc3Q=

ChainMakerTransactionResult

长安链交易查询结果

被如下接口引用：QueryChainMakerBlockTransaction, QueryChainMakerDemoBlockTransaction, QueryChainMakerDemoTransaction, QueryChainMakerTransaction。

名称	类型	描述
Code	Integer	交易结果码 示例值：0
CodeMessage	String	交易结果码含义 示例值：OK

名称	类型	描述
TxId	String	交易ID 示例值：6ab24544b68643318dd06dc062525e682eeeb9f143294436998127d0f274164d
GasUsed	Integer	Gas使用量 示例值：0
BlockHeight	Integer	区块高度 示例值：1
ContractEvent	String	合约执行结果 示例值：[{"contract_name":"ChainMakerDemo","contract_version":"v1"}]
Message	String	合约返回信息 示例值：OK
Timestamp	Integer	交易时间，单位是秒 示例值：1709261981

EndorserGroup

背书组织及其节点列表

被如下接口引用：GetTransactionDetailForUser。

名称	类型	描述
EndorserGroupName	String	背书组织名称 示例值：org1
EndorserPeerList	Array of String	背书节点列表 示例值：["peer-org1","peer-org2"]

PeerSet

PeerSet

被如下接口引用：Invoke, Query。

名称	类型	必选	描述
PeerName	String	是	节点名称 示例值：peer1-org1.fabric-xdfsds
OrgName	String	是	组织名称 示例值：org1.fabric-xdfsds

SignCertCsr

用于申请用户签名证书的结构体

被如下接口引用：ApplyChainMakerBatchUserCert。

名称	类型	必选	描述
CertMark	String	是	用户签名证书的标识，会存在于用户申请的证书中 示例值：alice
SignCsrContent	String	是	用户申请签名证书所需要的证书请求文件的base64编码 示例值：-----BEGINCERTIFICATEREQUEST-----

Transaction

交易显示概述信息

被如下接口引用：DescribeFabricBlock。

名称	类型	描述
TxId	String	交易ID 示例值: af5b8ec2e660461f9a32ec3c9ec151688a1c83e6ebb94815a21e84fbc29cf7c2
ChaincodeName	String	合约名称 示例值: save
Sender	String	交易发送者 示例值: zhangsan
CreateTime	String	交易创建时间 示例值: 2023-07-07 10:10:10
BlockHeight	Integer	交易所在区块高度 示例值: 1024
TxIndex	Integer	交易在区块中的序号 示例值: 0

TransactionItem

交易列表项信息

被如下接口引用: GetBlockTransactionListForUser, GetLatestTransactionList。

名称	类型	描述
TransactionId	String	交易ID 示例值: 0af5b0ae129b0435aa4ec8547c7500f428673dee363331f2f5f8d61d49e43e68
TransactionHash	String	交易hash 示例值: b5e398182b07bbde523e63804ee24690323bd5e190db99da079a00af2d71b652
CreateOrgName	String	创建交易的组织名 示例值: org1
BlockId	Integer	交易所在区块号 示例值: 3
TransactionType	String	交易类型（普通交易和配置交易） 示例值: ENDORSER_TRANSACTION
CreateTime	String	交易创建时间 示例值: 2020-01-06 22:18:15
BlockHeight	Integer	交易所在区块高度 示例值: 3
TransactionStatus	String	交易状态 示例值: VALID

错误码

最近更新时间：2024-12-20 01:53:20

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
AuthFailure	CAM签名/鉴权错误。
FailedOperation.BaaSStopServing	为带来更好的服务与体验，长安链体验网络维护升级中，请稍后再试。
FailedOperation.BcosService	Bcos网络异常。
FailedOperation.CaCheckCsr	检查CSR文件非法。
FailedOperation.CaDbOption	数据库操作错误。
FailedOperation.CaExsit	证书已经存在。
FailedOperation.CaGenkey	产生密钥对失败。
FailedOperation.Calnit	初始化CA失败。
FailedOperation.CalInputParam	输入参数非法。
FailedOperation.CaNoExist	证书不存在。
FailedOperation.CaRevoke	注销证书失败。
FailedOperation.CaRootNonExist	根证书不存在。
FailedOperation.CaService	CA内部错误。
FailedOperation.CaSignCert	签发证书失败。
FailedOperation.CaYunApiApplyCert	CFCA云API申请证书失败。
FailedOperation.CaYunApiCommon	CFCA云API错误。

错误码	说明
FailedOperation.CompileDeployedContract	Bcos不能编译已经部署的合约。
FailedOperation.CompileDeployingContract	Bcos不能编译正在部署的合约。
FailedOperation.ContractEditedByOtherAgency	Bcos只能由该合约所属机构执行操作。
FailedOperation.DatabaseException	数据库操作异常。
FailedOperation.DeleteDeployedContract	Bcos合约已部署，无法删除。
FailedOperation.DeleteDeployingContract	Bcos合约正在部署，无法删除。
FailedOperation.DeployContractNotCompile	Bcos不能部署未编译的合约。
FailedOperation.FabricBlockDetail	区块详情查询失败。
FailedOperation.FabricBlockNoExist	区块不存在。
FailedOperation.FabricBlockQuery	区块列表查询失败。
FailedOperation.FabricChaincodeInvokeFailed	合约调用失败。
FailedOperation.FabricChaincodeNoExist	合约不存在。
FailedOperation.FabricChaincodeQueryFailed	合约查询失败。
FailedOperation.FabricCommit	提交失败。
FailedOperation.FabricEventHub	event hub失败。
FailedOperation.FabricProposal	背书失败。
FailedOperation.FabricRequestParams	fabric请求参数错误。
FailedOperation.FabricRequstParams	参数验证失败。
FailedOperation.FabricTransactionDetail	交易详情查询失败。
FailedOperation.FabricTransactionNoExist	交易不存在。
FailedOperation.FabricTransactionQuery	交易列表查询失败。
FailedOperation.FabricTxIdQuery	Txid 查询失败。
FailedOperation.FabricTxInvoke	fabric invoke交易错误。
FailedOperation.FabricTxQuery	fabric query交易错误。
FailedOperation.FabricTxQueryNone	fabric query交易错误, 无正确。
FailedOperation.FrontRequestFail	Bcos前置服务调用失败。
FailedOperation.Grouplllegal	用户非法操作。
FailedOperation.InvalidAuth	用户无权限访问。
FailedOperation.InvalidGroupPk	Bcos无效的群组编号。
FailedOperation.InvalidKeyUser	Bcos无效的私钥用户信息。
FailedOperation.InvalidOperation	用户非法操作。
FailedOperation.ManageService	交易请求异常。
FailedOperation.NewContract	Bcos新建合约失败。
FailedOperation.NoChainCodeChannel	合约没有在通道初始化。
FailedOperation.NoChainCodeGroup	组织没有加入合约。
FailedOperation.NoChainCodePeer	合约没有在节点安装。
FailedOperation.NoChannelGroup	组织没有加入通道。

错误码	说明
FailedOperation.NoChannelPeer	节点没有加入通道。
FailedOperation.NoObject	对象不存在。
FailedOperation.NoPeer	没有可用节点。
FailedOperation.NotDeployedContract	Bcos合约未部署。
FailedOperation.ServiceFailed	服务调用失败，请检查参数。
FailedOperation.StatusNoMatch	操作状态不匹配。
FailedOperation.TimeOutUrl	后台服务请求超时。
FailedOperation.TransactionTimeout	交易执行超时，请稍后再试。
FailedOperation.UserAuthType	获取用户认证类型出错。
FailedOperation.UserInBlackList	您因违反用户协议，目前无法使用长安链体验网络。
FailedOperation.UserNoJoinDemoCluster	用户未加入体验网络。
InternalError.DBError	Bcos数据库操作异常，请重试。
InternalError.FailUrl	服务器异常，请重试。
InternalError.FlaskException	Flask内部错误。
InternalError.InvalidContractParam	Bcos无效的合约参数。
InternalError.MethodTypeNotSupport	Bcos不支持的请求类型。
InternalError.NoDefineError	错误码未定义。
InternalError.ServerError	内部错误，请稍后重试或者联系技术人员
InternalError.ServerException	服务器异常。
InternalError.ServiceError	服务异常，请重试。
InternalError.ServicePanic	交易服务异常，请重试。
InternalError.TransactionService	交易服务内部错误。
InternalError.UnknownError	交易服务未知错误，请重试。
InvalidParameter.AccountParamError	Bcos账号参数错误。
InvalidParameter.AgencyInvalid	Bcos无效的机构信息。
InvalidParameter.AgencyNetParamInvalid	Bcos新增机构网络关联信息入参错误。
InvalidParameter.AllianceldOfNetEmpty	Bcos所属联盟编号不能为空。
InvalidParameter.ContractIdInvalid	Bcos无效的合约编号。
InvalidParameter.DataHadExist	Bcos数据已存在，请勿重复添加。
InvalidParameter.EmptyParam	Bcos无效参数。
InvalidParameter.ExpireTimeInvalid	Bcos无效的失效时间。
InvalidParameter.FrontConnFail	Bcos前置的IP或端口无效。
InvalidParameter.FrontIpInvalid	Bcos IP格式错误。
InvalidParameter.FrontParamError	Bcos新增前置服务参数错误。
InvalidParameter.FrontRequestFail	Bcos前置服务调用失败。
InvalidParameter.InputDataViolation	输入参数存在违规内容。
InvalidParameter.InvalidContractArg	Bcos无效的合约参数。

错误码	说明
InvalidParameter.NetIdInvalid	Bcos无效的网络编号。
InvalidParameter.NetParamError	Bcos网络参数错误。
InvalidParameter.NewKeyUserParamError	Bcos新增私钥用户参数错误。
InvalidParameter.NoInfoToDelete	Bcos无可删信息,请确认后重试。
InvalidParameter.NotFoundValidFront	Bcos没有有效的前置服务信息。
InvalidParameter.ReDeployedContract	Bcos不能重复部署合约。
InvalidParameter.ReDeployingContract	Bcos合约正在部署中,请勿重复操作。
InvalidParameter.RoleInvalid	Bcos无效的角色。
InvalidParameterValue.IllegalFormat	参数格式不正确。
InvalidParameterValue.IllegalValue	请求参数错误。
InvalidParameterValue.ParameterEmpty	参数值为空。
MissingParameter.EmptyParam	Bcos缺少参数。
OperationDenied.NotOwner	只有所有者可以操作。
ResourceNotFound.EmptyData	Bcos查询不到数据。