大数据套件（TBDS）

产品文档

文档目录

产品简介
产品概述

最近更新时间: 2024-09-05 15:09:00

大数据套件（TBDS）是基于多年海量数据处理经验，对外提供的可靠、安全、易用的大数据处理平台。您可以借助 TBDS 在公有云、私有云、非云化环境，根据不同数据处理需求选择合适的大数据分析引擎和相应的实时数据开发、离线数据开发以及算法开发服务，来构建您的数据仓库、用户画像、精准推荐、风险管控等大数据应用服务。

主要功能

全链路数据开发

易用、安全、稳定、高性能的全链路大数据开发引擎。提供拖拽式的可视化数据开发 IDE，为用户的大数据集成、存储、计算环节提供完整而稳定的企业级解决方案。用户能借助大数据套件获取到强大的大数据开发能力，聚焦于进行企业的业务创新。

强大的数据分析与探索挖掘引擎

一站式数据分析、探索、挖掘平台。包含基于维度建模的多维分析、交互式探索分析、机器学习、深度学习、可视化敏捷报表门户等功能，向用户提供了强大的数据分析与数据挖掘能力，助力用户发现大数据的价值。

开箱即用的数据治理工具

开箱即用的数据治理工具。面向企业数据治理需求，提供完善的数据元信息管理功能。支持细到字段级别的数据权限管控，包含库表数据字典、数据血缘跟踪与溯源、热点数据分析等特色功能，以帮助企业客户提高海量数据资产的管理效率。

一站式运维管理平台

一站式的可视化运维管理平台。包含一键式集群部署、增量部署、丰富的可视化运维工具、完善的面向多租户的计算资源管控体系和用户权限管理体系，为客户提供企业级的大数据平台运维管理能力支撑。

产品优势

最近更新时间: 2024-09-05 15:09:00

技术开放

存储标准兼容开源 Hadoop 标准，可平滑迁移构建在 Hadoop 上的大数据平台。

- 支持多驱动接入、完美兼容社区标准。
- 支持 SQL2003 标准的内存迭代运算引擎 SparkSQL。
- 支持选购 PLSQL 语法驱动，以获得90%的 Oracle 语法支持。

安全可靠

- 数据节点分布式部署，可选多份备份。
- 所有系统控制节点主从热备，故障可秒级切换。经过95%的业务考验，可用性达99.999%。
- 支持数据加密传输、存储。
- 全平台单点登录，统一策略管控中心。
- 基于角色的数据管控体系，支持列级粒度权限控制。
- 完善的访问审计及预警模型。

性能卓越

高性能数据接入引擎，内部业务日接入五万亿条数据。性能全面超越社区方案，数据处理能力提升30%左右。支持上千维度、千亿规模数据的秒级交互式多维分析。

简单易用

支持一键式部署，只需选择您适合的服务即可快速完成部署。支持数据接入、处理、存储、分析、机器学习的拖拽式全链路大数据开发和开箱即用的数据治理工具集，数据开发者只需专注于业务开发。

场景丰富

在金融、政务、公安、零售、传统企业等领域积累了丰富的业务应用案例，特别是在 PB 级离线、近线、实时数仓企业级用户画像、金融实时风控、精准推荐、物联网大数据等场景我们都有成熟的解决方案。

生态完善

与各行业优秀的大数据开发、数据治理服务及应用商深度合作，为政务、金融、公安等行业客户提供"基础软件平台+应用+服务"的专业一体化大数据解决方案。

应用场景

最近更新时间: 2024-09-05 15:09:00

大数据套件 TBDS 适用于企业从 GB 到 TB、PB 级的大数据处理场景，包括但不限于以下场景：

数据仓库建设

大数据套件完整覆盖数据抽取、转换、加载、建模、分析、报表呈现、数据治理等数仓建设环节，用户可借助 TBDS 大数据套件在公有云、私有云、非云化环境快速建设 TB 到 PB 级的企业数据仓库和数据集市，搭建专属的大数据应用。

通过大数据处理套件，用户可降低基于企业数据仓库的数据应用开发周期和开发成本，以及数据仓库、数据处理、数据应用的运维成本。

实时流式数据处理

用户可基于大数据处理套件快速开发本行业在实时流式场景下的大数据处理、分析的应用程序，以实现对企业实时业务的风险监控与告警，以占据大数据时代的优势地位。

流式数据处理可用于金融行业的风险管控、物联网的海量传感器数据处理、工业生产线的实时故障预警、病人特征数据实时分析、实时交通流量分析、互联网实时流量分析等应用场景。

离线数据处理

大数据套件基于 Hadoop 体系的 MapReduce、Hive、Pig、Spark 技术，向企业用户提供了强大的数据离线批处理能力，用户可以便捷的使用大数据套件对企业数据进行抽取、转换、加载等离线数据处理加工。

通过离线数据处理引擎，用户可迅速的对企业所积累的数据进行 ETL 处理，快速发掘海量历史数据的商业价值和社会价值。

数据分析与探索挖掘

通过大数据套件所提供的强大数据分析与探索挖掘能力，用户可快速对企业在 PB 级规模下的大数据进行可视化的数据分析探索，在纷繁复杂的商业数据中快速获取数据洞察力，占领商业先机。

用户还可通过大数据套件所提供的强大机器学习能力对企业数据进行深度挖掘，进一步发掘海量数据中蕴藏的无限价值。

操作指南
查看集群列表

最近更新时间: 2024-09-05 15:09:00

1. 登录TBDS租户端控制台。

2. 在左侧导航栏单击【集群列表】，在【集群管理】页显示所有集群信息。

新建集群

最近更新时间: 2024-09-05 15:09:00

1. 登录TBDS租户端控制台。

2. 在左侧导航栏单击对应类型的【集群列表】，在【集群管理】页显示所有集群信息。

3. 单击【新建】，显示【新建集群】页面。

4. 在【集群信息】页签中填写【集群名称】，选择【地域】、【可用区】、【架构】、【TBDS版本】、【集群网络】、【操作系统】，填写【集群描述】，单击【下一步】。

5. 在【选择数量】页签中分别单击【portal节点】、【master节点】、【worker节点】页签填写对应的参数，单击【下一步】。

6. 在【信息确认】页签中确信信息无误后，单击【完成】。

常见问题
套件运维简介

最近更新时间: 2024-09-05 15:09:00

大数据套件（以下简称TBDS）是基于多年海量数据处理经验，对外提供的可靠、安全、易用的大数据处理平台。用户可以按需部署大数据处理服务实现数据处理需求，例如：报表展示，数据提取、分析，客户画像等大数据应用。

TBDS 为客户提供一站式运维管理平台，无需登录服务器或执行命令，通过运维管理界面即可完成如组件部署、配置变更、指标监控、日志诊断等日常运维工作。

服务和节点管理

最近更新时间: 2024-09-05 15:09:00

组件部署概览

1. 登录 TBDS Portal，单击【运维中心】模块，进入运维中心，如下：

2. 进入组件部署概览页面，可查看到当前集群包含的主机总数、组件总数以及正常/停止组件数。

3. 通过切换主机和服务标签页，可分别进入主机列表页和组件列表页。

组件列表页

1. 单击【本集群包含服务】链接进入服务列表页，可以查看当前集群中运行的所有组件。

2. 单击左侧组件菜单中的不同组件可切换到不同组件的详细信息页。 每个组件的详细信息页都提供了组件的启动、停止、重启功能，以及服务概览信息、配置信息、分布主机信息。

增加服务

1. 在服务列表页，单击【增加服务】，进入增加服务向导，添加新服务：

2. 在增加服务向导，页面中展示了套件中可增加的服务、版本号以及服务简介。我们以ntp服务为例执行安装操作：

3. 勾选【服务】，单击【下一步】，进入规划部署页面：

4. 在右侧服务节点列表中选择将要部署的节点IP，单击【下一步】，进入自定义服务配置页：

5. 此时可以调整服务配置，我们使用默认配置，单击【下一步】，

此时启动服务安装，查看当前安装进度并等待服务安装完成：

6. 待安装成功，进度显示100%后，单击【完成】，至此服务安装完成。

之后进入服务列表页，可以查看到新增加的ntp服务已经处于运行中。

卸载服务

进入服务列表页，通过右侧【卸载】，可以快速卸载服务，我们以ntp服务为例。

> 卸载服务前必须停止当前服务才能查看到卸载按钮。

1. 单击【停止】停止服务：

2. 此时弹出对话框，单击【确认】，停止服务：

3. 待服务完成停止，单击【卸载】，在弹出的确认框单击【确定】：

此时页面返回卸载成功，服务即成功卸载。

服务扩缩容

通过组件部署可以快速完成服务的扩缩容。 套件支持包括Lhotse runner、Hbase Region，Kafka Broker、ES Server等服务的节点扩容操作，标准化的操作流程以及便捷的操作方式可以帮助客户快速完成容量规划和实施。

服务扩容

以扩容Lhotse Runner为例。

1. 进入Lhotse Runner服务信息页，确认当前runner节点数，切换到【主机】标签页，查看详细的主机列表：

2. 在主机列表中，单击LHOTSE_RUNNER展开下拉列表，查看主机IP信息，回到组件部署概览页，通过左侧【查看主机列表】链接进入主机列表：

3. 选择新的runner节点需要扩容到的目标服务器IP，并单击进入。

   此时在主机的摘要信息中，可以看到当前主机所安装的组件列表，单击【添加组件】，下拉菜单中选择"Lhotse Runner"组件：

4. 弹出确认框，确认无误后单击【确定】：

5. 等待其执行完成，Lhotse Runner的新节点即安装完成：

6. 此时回到新节点主机的组件摘要页，可以看到runner已经安装完成，处于停止状态，单击【启动】即可启动新扩容的runner。

服务缩容

缩容和扩容流程正好相反，仅需要在主机的组件摘要页先停止需要缩容的服务，然后执行【删除】即可：

单击【删除】之后，弹出确认框，单击确定即可完成组件卸载。

新增节点

通过新增节点功能可以将一台新的节点添加到当前集群中。

新增的节点可以用于部署新服务或者扩容现有服务，实现容量和服务能力的水平扩展。

1. 新节点需要先经过初始化之后，再由界面添加进集群，初始化方式如下：

```
./tbds-bootstrap.sh pushkey ip \#此处ip为新节点ip

./tbds-bootstrap.sh init ip

./tbds-bootstrap.sh postinit ip
```

2. 初始化之后在组件部署概览页面，单击左侧【本集群包含主机】链接进入主机列表：

3. 单击【新增主机】：

4. 此时进入【新增主机】向导页，新增机器流程包括主机注册、主机分配、服务配置、安装启动等步骤，在填写主机页面，填入新增机器的IP 账号及密码信息，其中主机名为可选项，如未指定主机名，则系统会按照默认规则自动设置主机名。

5. 填好以上信息后，单击【注册】，开始注册主机，待主机注册成功以后，继续单击【下一步】即可，后续步骤于服务安装步骤类似。

监控告警和日志诊断

最近更新时间: 2024-09-05 15:09:00

TBDS套件内提供了完善的服务指标监控和日志诊断功能，通过Portal即可完成监控和诊断的操作，套件运维操作的规划化和自动化使得客户在构建大数据能力的过程中得到很好的保障，运营能力得以加强。

服务监控

1. 登录Portal后，通过【运维中心】模块进入运维管理页面，单击【监控告警】标签，即可打开监控导览页：

2. 进入到监控页面，左右角分别提供了【硬件监控】、【进程指标】以及服务组件指标的页面入口，套件内已内置了常见的监控指标panel。

   i. 单击【硬件指标】链接，进入硬件指标页，通过选择【主机】下拉框中的主机，可切换查看不同主机的CPU使用率、磁盘使用量、负载、网络使用等指标信息：

   ii. 单击【进程指标】链接，进入进程指标页，和【硬件指标】页类似，支持切换主机以查看相应的监控指标数据：

3. 同时监控系统还提供了客户自定义监控面板的能力，单击导览页右上角【+】加号按钮，进入增加面板页面，需要注意的是套件中限制了只能由账户 admin 有权限操作监控面板自定义。

   i. 其中【标题】填写新建指标面板的自定义标题，我们以FLUME为例，新建一个面板，单击下方【新建panel】按钮，新建一个指标展示panel：

   ii. 在【指标】标签页，数据来源选择opentsdb，Metric中选择指标名称：

iii. 在【坐标轴&图例】标签页中可以设置坐标轴单位、图例展示位置以及图示展示选项：

iv. 在【基本信息】标签页，可以自定义panel标题，panel展示尺寸：

v. 在【格式】标签页，可以设置绘制线条的样式以及多系列值的展示样式：

vi. 在【时间范围】标签页，可以设置时间展示区间和时间偏移：

当完成上述自定义参数设置后，单击右上角保存按钮，当前定制的面板和panel即可保存。

日志诊断

1. 登录Portal后，单击【运维中心】中的【文件管理】标签页，进入日志诊断页。 日志诊断系统搜集了套件内组件的运行日志，并提供集中展示和快速检索。

2. 单击右上角【更多筛选】按钮展开筛选菜单，第一个输入框内可以输入日志关键字，时间选择器中可以选择查找的日志时间范围，主机中可以填写指定的主机，日志级别可筛选不同的日志等级，服务和组件中可快速筛选指定服务类型和组件类型。

   如下图所示，我们筛选了一段时间内的hbase服务日志中包含'hstore completed'关键字的日志，单击【查找】按钮即可找到相应的日志条目：

服务告警

进入【运维中心】，打开【监控告警】标签页，单击左侧【】按钮进入集群告警处理页面。 套件提供了对机器指标、组件状态、服务指标等的统一监控和告警能力，当监控到某项指标达到指定告警规则的阈值后，即可触发发送告警信息到指定人员。告警又可分为指标告警和服务告警。

指标告警

针对不同的指标定义告警规则，当监控到具体指标的值满足告警规则定义的触发方式时，即产生告警记录。单击【创建告警】，进入指标告警规则创建菜单：

告警规则可选择打开或者关闭，通知方式支持邮件和短信，指标支持进程指标、硬件指标、服务指标，告警触发条件可选阈值、同比或者环比，检测周期支持1分钟，5分钟，30分钟，1小时。最后单击创建保存即可。

服务告警

针对服务和组件的运行状态的告警，监控服务运行健康状态和组件存活情况。套件中默认已经内置针对所有服务的服务告警。

单击【编辑】可以对告警接收人，告警通知方式，监控频率等进行修改。

告警记录查看

单击【告警记录】按钮可以查看系统产生的历史告警信息，支持对告警信息按照接收人、指标名、服务类型以及告警时间进行筛选。当发生告警后，请及时作出响应，将对应的指标或者服务恢复到正常水平，系统监控到服务水平恢复后，告警会自动解除。

访问权限管理与审计

最近更新时间: 2024-09-05 15:09:00

套件中提供了针对hdfs、hbase、hive等服务的访问权限控制及日志审计功能，可实现按照项目、用户组、用户等维度的权限管控。运维中心提供了访问权限策略的管理以及审计日志查看功能。

进入运维中心，切换到【访问管理】标签页即可进入访问管理页面。

访问策略管理

在访问管理页面，单击左侧【】按钮进入访问策略管理页面，不同的服务拥有独立的策略，单击页面中的服务链接进入对应服务的策略管理页面。例如hdfs的页面中可以查看到如下策略：

审计日志查看

当访问管理策略生效后，任何针对策略所覆盖到的资源的读写操作都会产生审计日志，单击左侧【】按钮打开审计日志查看页面。可以查看到服务资源的访问记录以及访问策略的变更记录。

链接归集

最近更新时间: 2024-09-05 15:09:00

链接归集提供用户访问大数据套件TBDS内部组件的部分web和展示内部组件注册过服务发现的端口列表。组件如hdfs NN的web访问需要先授权，授权之后才能访问。

链接访问

1. 打开TBDS portal，输入正确的用户名密码，进入portal主页面。

2. 依次单击进入【运维中心】 > 【系统运维】 > 【链接管理】，进入链接管理页面，默认停留在链接页面。

3. 查看链接列表，确定需要访问的链接项，单击对应链接条目右侧的【申请链接权限】。

4. 申请之后需要等待管理员的审批，审批通过之后才能继续访问。下面说明管理员审批步骤。

   i. 管理员登录TBDS portal，进入portal主页面。

   ii. 单击页面右上角的【个人中心】，进入【个人中心】。一次单击【审批单】 > 【链接申请】，就可以看到普通用户刚刚提交的链接访问申请。单击申请单右侧的【通过】，则授权成功。

   iii. 审批通过的单据可以通过以下方式查看。

5. 普通用户这个时候再次登录链接归集页面就可以看到刚才想要访问的地址的具体url，至此后台也在管理员审批的同时赋予了普通用户访问此链接的权限。

6. 根据链接归集页面上的提示配置普通用户所在机器的主机-ip映射。以windows为例，则需配置C:\Windows\System32\drivers\etc\hosts，在该文件里加上想要访问的链接的host项。

7. 在浏览器输入链接地址，并输入用户名密码就可以打开具体web页面了。用户名密码同登录portal所用的用户名密码。

访问端口列表

直接进入portal的链接归集页面，单击右侧的【端口列表】即可。

备份管理

最近更新时间: 2024-09-05 15:09:00

快照和归档是大数据套件提供的保护数据安全的功能。目前快照支持 hdfs 和 hbase，归档支持 hdfs。

- HDFS 快照是一个只读的基于时间点文件系统拷贝。快照可以是整个文件系统的也可以是一部分。常用来作为数据备份，防止用户错误和容灾。快照会存储在 snapshottable 的目录下。snapshottable 下存储的 snapshots 最多为65535个。没有限制 snapshottable 目录的数量。管理员可以设置任何的目录成为 snapshottable。如果 snapshottable 里面存着快照，那么文件夹不能删除或者改名。

- hbase 快照就是一份元信息的合集，允许管理员恢复到表的先前状态。快照不是表的复制而是一个文件名称列表，因而不会复制数据。完全快照恢复是指恢复到之前的"表结构"以及当时的数据，快照之后发生的数据不会恢复。

快照

1. 打开大数据套件TBDS的portal，输入用户名密码登录。依次进入【运维中心】 > 【系统运维】 > 【备份】。默认进入数据快照主页面。

2. 单击【新建快照策略】，进入编辑快照策略详情页面，可以填写具体策略配置信息。其中执行周期分为立即执行和按日月周等周期性执行，可以根据业务需求自行选定。

3. 保存好快照策略之后，可以查看和修改快照的执行情况。

- 将鼠标放在快照策略列表的某条策略条目右侧的【操作】上，可以查看历史执行情况，禁用该策略，重新定义策略规则等操作。

- 如果单击进入【执行历史】，则可以看到快照每次历史执行的详细信息，包括执行开始时间，持续时间，以及快照生成的文件名。执行历史条目的右侧有""操作"链接，可以查看本次执行的日志，删除快照文件，从本次快照还原文件，等操作。

归档

1. 同快照一样，首先进入备份页面，单击右侧的【数据归档策略】，进入归档主页面。

2. 在该页面右侧单击【新建归档策略】，进入归档策略配置编辑界面。同快照一样，归档也可以定期（周期）执行，确保数据安全。

3. 新建好归档策略保存好后，可以对策略进行修改重新编辑以及查看归档策略的执行情况，分别如下 2 图所示。

系统设置

最近更新时间: 2024-09-05 15:09:00

进入运维中心，单击【系统设置】标签页进入系统设置页面。在系统设置中，提供了系统基本信息修改、公告管理、通知渠道管理以及产品信息查看等功能。其中：

- 系统信息：支持系统名称、LOGO的修改；
- 公告管理：添加的公告信息可以指定时段、指定项目进行展示，已经添加的公告可编辑、停止或者删除；公告信息将展示在首页醒目位置。
- 通知渠道：告警信息通过此处配置的通知渠道发送，包括邮件渠道和短信渠道。
- 产品信息：可查看当前系统所使用的license以及到期时间，最大节点数等信息。

API文档
腾讯大数据套件（tbds）
版本（2020-01-16）
API概览

最近更新时间: 2024-09-11 15:04:50

API版本

V3

腾讯大数据套件

接口名称	接口功能
BmsCreateCluster	bms创建集群
BmsDescribeInstancesByName	根据实例名称获取bms实例信息
ClusterExists	判断集群名是否存在
CreateCluster	创建集群
DescribeClusterDetail	获取集群详情
DescribeClusterNodes	获取集群节点
DestroyCluster	销毁集群
GetClusters	获得集群列表
GetProgress	获得集群部署进度
TestApplyBms	bms创建实例接口测试

调用方式
接口签名v1

最近更新时间: 2024-09-11 15:04:50

tcecloud 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tcecloud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，点击【新建】即可以创建一对SecretId/SecretKey

注意：开发商帐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。以下是生成签名串的详细过程：

假设用户的 SecretId 和 SecretKey 分别是：

- SecretId: AKIDz8krbsJ5yKBZQpn74WFkmLPx3EXAMPLE
- SecretKey: Gu5t9xGARNpq86cd98joQYCN3EXAMPLE

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	AKIDz8krbsJ5yKBZQpn74WFkmLPx3EXAMPLE
Timestamp	当前时间戳	1465185768
Nonce	随机正整数	11886
Region	实例所在区域	ap-guangzhou

参数名称	中文	参数值
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

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' : 'AKIDz8krbsJ5yKBZQpn74WFkmLPx3EXAMPLE',
'Timestamp' : 1465185768,
'Version': '2017-03-12',
}
```

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

2.2. 拼接请求字符串

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

然后将格式化后的各个参数用"&"拼接在一起，最终生成的请求字符串为:

```
Action=DescribeInstances&InstanceIds.0=ins-09dx96dg&Limit=20&Nonce=11886&Offset=0&Region=ap-guangzhou&SecretId=AKIDz8krbsJ5yKBZQpn74WFkmLPx3EXAMPLE&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.finance.cloud.tencent.com。实际的请求域名根据接口所属模块的不同而不同，详见各接口说明。
3. 请求路径: 当前版本云API的请求路径固定为 / 。
4. 请求字符串: 即上一步生成的请求字符串。

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

示例的拼接结果为：

```
GETcvm.finance.cloud.tencent.com/?Action=DescribeInstances&InstanceIds.0=ins-09dx96dg&Limit=20&Nonce=11886&Offset=0&Region=ap-guangzhou&SecretId=AKIDz8krbsJ5yKBZQpn74WFkmLPx3EXAMPLE&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 = 'Gu5t9xGARNpq86cd98joQYCN3EXAMPLE';
$srcStr = 'GETcvm.finance.cloud.tencent.com/?Action=DescribeInstances&InstanceIds.0=ins-09dx96dg&Limit=20&Nonce=11886&Offset=0&Region=ap-guangzhou&SecretId=AKIDz8krbsJ5yKBZQpn74WFkmLPx3EXAMPLE&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:

```
EliP9YW3pW28FpsEdkXt/+WcGeI=
```

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 EliP9YW3pW28FpsEdkXt/+WcGeI= ，最终得到的签名串请求参数（Signature）为：EliP9YW3pW28FpsEdkXt%2f%2bWcGeI%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 http 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

错误代码	错误描述
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tcecloud 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
- JavaScript
- .NET

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： http://imgcache.finance.cloud.tencent.com:80cvm.finance.cloud.tencent.com/?Action=DescribeInstances&InstanceIds.0=ins-09dx96dg&Limit=20&Nonce=11886&Offset=0&Region=ap-guangzhou&SecretId=AKIDz8krbsJ5yKBZQpn74WFkmLPx3EXAMPLE&Signature=EliP9YW3pW28FpsEdkXt%2F%2BWcGeI%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，即使是旧版的 API ，由于存在细节差异也会导致签名计算错误，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 TceCloudAPIDemo {
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.finance.cloud.tencent.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("http://imgcache.finance.cloud.tencent.com:80cvm.finance.cloud.tencent.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
params.put("SecretId", "AKIDz8krbsJ5yKBZQpn74WFkmLPx3EXAMPLE"); // 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
params.put("Signature", sign(getStringToSign(params), "Gu5t9xGARNpq86cd98joQYCN3EXAMPLE", "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 time

import requests
```

```
secret_id = "AKIDz8krbsJ5yKBZQpn74WFkmLPx3EXAMPLE"
secret_key = "Gu5t9xGARNpq86cd98joQYCN3EXAMPLE"

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.finance.cloud.tencent.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("http://imgcache.finance.cloud.tencent.com:80" + endpoint, params=data)
# print(resp.url)
```

接口签名v3

最近更新时间: 2024-09-11 15:04:50

tcecloud 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tcecloud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，点击【新建】即可以创建一对SecretId/SecretKey

注意：开发商帐号最多可以拥有两对 SecretId / SecretKey。

2. TC3-HMAC-SHA256 签名方法

注意：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 格式默认所有业务接口均支持，multipart 格式只有特定业务接口支持，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。

下面以云服务器查询广州区实例列表作为例子，分步骤介绍签名的计算过程。我们仅用到了查询实例列表的两个参数：Limit 和 Offset，使用 GET 方法调用。

假设用户的 SecretId 和 SecretKey 分别是：AKIDz8krbsJ5yKBZQpn74WFkmLPx3EXAMPLE 和 Gu5t9xGARNpq86cd98joQYCN3EXAMPLE

2.1. 拼接规范请求串

按如下格式拼接规范请求串（CanonicalRequest）：

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

- HTTPRequestMethod：HTTP 请求方法（GET、POST ），本示例中为 GET；

- CanonicalURI：URI 参数，API 3.0 固定为正斜杠（/）；
- CanonicalQueryString：发起 HTTP 请求 URL 中的查询字符串，对于 POST 请求，固定为空字符串，对于 GET 请求，则为 URL 中问号（?）后面的字符串内容，本示例取值为：Limit=10&Offset=0。注意：CanonicalQueryString 需要经过 URL 编码。
- CanonicalHeaders：参与签名的头部信息，至少包含 host 和 content-type 两个头部，也可加入自定义的头部参与签名以提高自身请求的唯一性和安全性。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）的字典排序进行拼接。此例中为：`content-type:application/x-www-form-urlencoded\nhost:cvm.finance.cloud.tencent.com\n`
- 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）按照字典排序进行拼接，并且以分号（;）分隔。此例中为：`content-type;host`
- HashedRequestPayload：请求正文的哈希值，计算方法为 Lowercase(HexEncode(Hash.SHA256(RequestPayload)))，对 HTTP 请求整个正文 payload 做 SHA256 哈希，然后十六进制编码，最后编码串转换成小写字母。注意：对于 GET 请求，RequestPayload 固定为空字符串，对于 POST 请求，RequestPayload 即为 HTTP 请求正文 payload。

根据以上规则，示例中得到的规范请求串如下（为了展示清晰，\n 换行符通过另起打印新的一行替代）：

```
GET
/
Limit=10&Offset=0
content-type:application/x-www-form-urlencoded
host:cvm.finance.cloud.tencent.com

content-type;host
e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
```

2.2. 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
```

- 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 X-TC-Timestamp 取值，如上示例请求为 1539084154；
- 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，例如 cvm。如上示例请求，取值为 2018-10-09/cvm/tc3_request；
- HashedCanonicalRequest：前述步骤拼接所得规范请求串的哈希值，计算方法为 Lowercase(HexEncode(Hash.SHA256(CanonicalRequest)))。

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，可能导致运行一段时间后，请求必定失败（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（为了展示清晰，\n 换行符通过另起打印新的一行替代）：

```
TC3-HMAC-SHA256
1539084154
2018-10-09/cvm/tc3_request
91c9c192c14460df6c1ffc69e34e6c5e90708de2a6d282cccf957dbf1aa7f3a7
```

2.3. 计算签名

1）计算派生签名密钥，伪代码如下

```
SecretKey = "Gu5t9xGARNpq86cd98joQYCN3EXAMPLE"
SecretDate = HMAC_SHA256("TC3" + SecretKey, Date)
SecretService = HMAC_SHA256(SecretDate, Service)
SecretSigning = HMAC_SHA256(SecretService, "tc3_request")
```

- SecretKey：原始的 SecretKey；
- Date：即 Credential 中的 Date 字段信息，如上示例，为2018-10-09；
- Service：即 Credential 中的 Service 字段信息，如上示例，为 cvm；

2）计算签名，伪代码如下

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

- SecretSigning：即以上计算得到的派生签名密钥；
- StringToSign：即步骤2计算得到的待签名字符串；

2.4. 拼接 Authorization

按如下格式拼接 Authorization：

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

- Algorithm：签名方法，固定为 TC3-HMAC-SHA256；
- SecretId：密钥对中的 SecretId；
- CredentialScope：见上文，凭证范围；
- SignedHeaders：见上文，参与签名的头部信息；
- Signature：签名值

根据以上规则，示例中得到的值为：

```
TC3-HMAC-SHA256 Credential=AKIDEXAMPLE/Date/service/tc3_request, SignedHeaders=content-type;host, Signature=5da7a33f6993f0614b047e5df4582db9e9bf4672ba50567dba16c6ccf174c474
```

最终完整的调用信息如下：

```
http://imgcache.finance.cloud.tencent.com:80cvm.finance.cloud.tencent.com/?Limit=10&Offset=0

Authorization: TC3-HMAC-SHA256 Credential=AKIDz8krbsJ5yKBZQpn74WFkmLPx3EXAMPLE/2018-10-09/cvm/tc3_request, SignedHeaders=content-type;host, Signature=5da7a33f6993f0614b047e5df4582db9e9bf4672ba50567dba16c6ccf174c474
Content-Type: application/x-www-form-urlencoded
Host: cvm.finance.cloud.tencent.com
X-TC-Action: DescribeInstances
X-TC-Version: 2017-03-12
X-TC-Timestamp: 1539084154
X-TC-Region: ap-guangzhou
```

3. 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）

4. 签名演示

Java

```
import java.io.BufferedReader;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.net.URL;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.Map;
import java.util.TimeZone;
import java.util.TreeMap;
import javax.crypto.Mac;
import javax.crypto.spec.SecretKeySpec;
import javax.net.ssl.HttpsURLConnection;
import javax.xml.bind.DatatypeConverter;

import org.apache.commons.codec.digest.DigestUtils;

public class TceCloudAPITC3Demo {
private final static String CHARSET = "UTF-8";
```

```
private final static String ENDPOINT = "cvm.finance.cloud.tencent.com";
private final static String PATH = "/";
private final static String SECRET_ID = "AKIDz8krbsJ5yKBZQpn74WFkmLPx3EXAMPLE";
private final static String SECRET_KEY = "Gu5t9xGARNpq86cd98joQYCN3EXAMPLE";
private final static String CT_X_WWW_FORM_URLENCODED = "application/x-www-form-urlencoded";
private final static String CT_JSON = "application/json";
private final static String CT_FORM_DATA = "multipart/form-data";

public static byte[] sign256(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(CHARSET));
}

public static void main(String[] args) throws Exception {
String service = "cvm";
String host = "cvm.finance.cloud.tencent.com";
String region = "ap-guangzhou";
String action = "DescribeInstances";
String version = "2017-03-12";
String algorithm = "TC3-HMAC-SHA256";
String timestamp = "1539084154";
//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 = "GET";
String canonicalUri = "/";
String canonicalQueryString = "Limit=10&Offset=0";
String canonicalHeaders = "content-type:application/x-www-form-urlencoded\n" + "host:" + host + "\n";
String signedHeaders = "content-type;host";
String hashedRequestPayload = DigestUtils.sha256Hex("");
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 = DigestUtils.sha256Hex(canonicalRequest.getBytes(CHARSET));
String stringToSign = algorithm + "\n" + timestamp + "\n" + credentialScope + "\n" + hashedCanonicalRequest;
System.out.println(stringToSign);

// ************ 步骤 3：计算签名 ************
byte[] secretDate = sign256(("TC3" + SECRET_KEY).getBytes(CHARSET), date);
byte[] secretService = sign256(secretDate, service);
byte[] secretSigning = sign256(secretService, "tc3_request");
String signature = DatatypeConverter.printHexBinary(sign256(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("Host", host);
headers.put("Content-Type", CT_X_WWW_FORM_URLENCODED);
headers.put("X-TC-Action", action);
headers.put("X-TC-Timestamp", timestamp);
headers.put("X-TC-Version", version);
headers.put("X-TC-Region", region);
}
}
```

Python

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

# 密钥参数
secret_id = "AKIDz8krbsJ5yKBZQpn74WFkmLPx3EXAMPLE"
secret_key = "Gu5t9xGARNpq86cd98joQYCN3EXAMPLE"

service = "cvm"
host = "cvm.finance.cloud.tencent.com"
endpoint = "http://imgcache.finance.cloud.tencent.com:80" + host
region = "ap-guangzhou"
action = "DescribeInstances"
version = "2017-03-12"
algorithm = "TC3-HMAC-SHA256"
timestamp = 1539084154
date = datetime.utcfromtimestamp(timestamp).strftime("%Y-%m-%d")
params = {"Limit": 10, "Offset": 0}

# ************* 步骤 1：拼接规范请求串 *************
http_request_method = "GET"
canonical_uri = "/"
canonical_querystring = "Limit=10&Offset=0"
ct = "x-www-form-urlencoded"
payload = ""
if http_request_method == "POST":
canonical_querystring = ""
ct = "json"
payload = json.dumps(params)
canonical_headers = "content-type:application/%s\nhost:%s\n" % (ct, host)
signed_headers = "content-type;host"
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)

# 公共参数添加到请求头部
headers = {
"Authorization": authorization,
"Host": host,
"Content-Type": "application/%s" % ct,
"X-TC-Action": action,
"X-TC-Timestamp": str(timestamp),
"X-TC-Version": version,
"X-TC-Region": region,
}
```

请求结构

最近更新时间: 2024-09-11 15:04:50

1. 服务地址

地域（Region）是指物理的数据中心的地理区域。tcecloud交付验证不同地域之间完全隔离，保证不同地域间最大程度的稳定性和容错性。为了降低访问时延、提高下载速度，建议您选择最靠近您客户的地域。

您可以通过 API接口 **查询地域列表** 查看完成的地域列表。

2. 通信协议

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

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

4. 字符编码

均使用UTF-8编码。

返回结果

最近更新时间: 2024-09-11 15:04:50

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，可以联系我们，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，可以联系我们，并提供该 ID 来解决问题。

公共错误码 (TODO: 重复信息, 是否真的需要?)

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，下表列出了公共错误码。

错误码	错误描述
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。
AuthFailure.SignatureExpire	签名过期。
AuthFailure.SignatureFailure	签名错误。
AuthFailure.TokenFailure	token 错误。
AuthFailure.UnauthorizedOperation	请求未 CAM 授权。
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。
InvalidParameterValue	参数取值错误。
LimitExceeded	超过配额限制。
MissingParameter	缺少参数错误。
NoSuchVersion	接口版本不存在。
RequestLimitExceeded	请求的次数超过了频率限制。
ResourceInUse	资源被占用。
ResourceInsufficient	资源不足。
ResourceNotFound	资源不存在。
ResourceUnavailable	资源不可用。
UnauthorizedOperation	未授权操作。
UnknownParameter	未知参数错误。
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。

公共参数

最近更新时间: 2024-09-11 15:04:50

公共参数是用于标识用户和接口鉴权目的的参数，如非必要，在每个接口单独的接口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。

签名方法 v3

使用 TC3-HMAC-SHA256 签名方法时，公共参数需要统一放到 HTTP Header 请求头部中，如下：

参数名称	类型	必选	描述
X-TC-Action	String	是	操作的接口名称。取值参考接口文档中输入参数公共参数 Action 的说明。例如云服务器的查询实例列表接口，取值为 DescribeInstances。
X-TC-Region	String	是	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。
X-TC-Timestamp	Integer	是	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。
X-TC-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。
Authorization	String	是	HTTP 标准身份认证头部字段，例如： TC3-HMAC-SHA256 Credential=AKIDEXAMPLE/Date/service/tc3_request, SignedHeaders=content-type;host, Signature=fe5f80f77d5fa3beca038a248ff027d0445342fe2855ddc963176630326f1024 其中， - TC3-HMAC-SHA256：签名方法，目前固定取该值； - Credential：签名凭证，AKIDEXAMPLE 是 SecretId；Date 是 UTC 标准时间的日期，取值需要和公共参数 X-TC-Timestamp 换算的 UTC 标准时间日期一致；service为产品名，必须与调用的产品域名一致，例如cvm； - 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。
X-TC-Token	String	否	临时证书所用的 Token ，需要结合临时密钥一起使用。临时密钥和 Token 需要到访问管理服务调用接口获取。长期密钥不需要 Token。

签名方法 v1

使用 HmacSHA1 和 HmacSHA256 签名方法时，公共参数需要统一放到请求串中，如下

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。

参数名称	类型	必选	描述
Region	String	是	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。具体计算方法参见接口鉴权文档。
Version	String	是	操作的 API 的版本。取值参考接口文档中入参公共参数 Version 的说明。例如云服务器的版本 2017-03-12。
SignatureMethod	String	否	签名方式，目前支持 HmacSHA256 和 HmacSHA1。只有指定此参数为 HmacSHA256 时，才使用 HmacSHA256 算法验证签名，其他情况均使用 HmacSHA1 验证签名。
Token	String	否	临时证书所用的 Token ，需要结合临时密钥一起使用。临时密钥和 Token 需要到访问管理服务调用接口获取。长期密钥不需要 Token 。

地域列表

地域（Region）是指物理的数据中心的地理区域。tcecloud交付验证不同地域之间完全隔离，保证不同地域间最大程度的稳定性和容错性。为了降低访问时延、提高下载速度，建议您选择最靠近您客户的地域。

您可以通过 API接口 **查询地域列表** 查看完成的地域列表。

腾讯大数据套件
bms创建集群

最近更新时间: 2024-09-11 15:04:50

1. 接口描述

接口请求域名： tbds.api3.finance.cloud.tencent.com。

创建TBDS集群，申请BMS机器，并在此机器上部署TBDS集群.

默认接口请求频率限制：20次/秒。

接口更新时间：2021-12-07 10:25:24。

接口只验签名不鉴权。

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	允许NULL	类型	描述
Action	是	否	String	公共参数，本接口取值：BmsCreateCluster
Version	是	否	String	公共参数，本接口取值：2020-01-16
Region	是	否	String	公共参数，本接口不需要传递此参数。
Name	是	否	String	集群的名称
Creator	否	否	String	创建人
CvmImageId	是	否	String	Docker镜像ID
ZoneId	是	否	String	实例所属可用区ID
VpcId	是	否	String	私有网络ID
SubnetId	是	否	String	Bms子网ID
Desc	是	否	String	集群描述
MasterInstanceInfo	是	否	BmsInstanceInfo	master 节点参数
PortalInstanceInfo	是	否	InstanceInfo	portal 节点参数
WorkerInstanceInfo	是	否	BmsInstanceInfo	worker 节点参数
CvmSubnetId	是	否	String	cvm的子网id
OperatingSystem	是	否	String	操作系统名称，例如：Centos 7.5 for x86_64

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，每次请求都会返回。定位问题时需要提供该次请求的 RequestId。

4. 错误码

该接口暂无业务逻辑相关的错误码，其他错误码详见公共错误码。

根据实例名称获取bms实例信息

最近更新时间: 2024-09-11 15:04:50

1. 接口描述

接口请求域名： tbds.api3.finance.cloud.tencent.com。

根据实例名称获取bms实例信息

默认接口请求频率限制：20次/秒。

接口更新时间：2021-11-15 16:12:56。

接口只验签名不鉴权。

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	允许NULL	类型	描述
Action	是	否	String	公共参数，本接口取值：BmsDescribeInstancesByName
Version	是	否	String	公共参数，本接口取值：2020-01-16
Region	是	否	String	公共参数，本接口不需要传递此参数。
InstanceNames	否	否	Array of String	实例名称数组

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，每次请求都会返回。定位问题时需要提供该次请求的 RequestId。

4. 错误码

该接口暂无业务逻辑相关的错误码，其他错误码详见公共错误码。

判断集群名是否存在

最近更新时间: 2024-09-11 15:04:50

1. 接口描述

接口请求域名： tbds.api3.finance.cloud.tencent.com。

无

默认接口请求频率限制：20次/秒。

接口更新时间：2020-03-23 21:43:06。

接口只验签名不鉴权。

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	允许NULL	类型	描述
Action	是	否	String	公共参数，本接口取值：ClusterExists
Version	是	否	String	公共参数，本接口取值：2020-01-16
Region	是	否	String	公共参数，本接口不需要传递此参数。
Name	是	否	String	集群名称

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，每次请求都会返回。定位问题时需要提供该次请求的 RequestId。

4. 错误码

该接口暂无业务逻辑相关的错误码，其他错误码详见公共错误码。

创建集群

最近更新时间: 2024-09-11 15:04:50

1. 接口描述

接口请求域名： tbds.api3.finance.cloud.tencent.com。

创建TBDS集群，申请CVM机器，并在此机器上部署TBDS集群

默认接口请求频率限制：20次/秒。

接口更新时间：2020-12-01 20:03:21。

接口只验签名不鉴权。

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	允许NULL	类型	描述
Action	是	否	String	公共参数，本接口取值：CreateCluster
Version	是	否	String	公共参数，本接口取值：2020-01-16
Region	是	否	String	公共参数，本接口不需要传递此参数。
Name	是	否	String	集群的名称
Creator	否	否	String	创建人
ImageId	是	否	String	Docker镜像ID
ZoneId	是	否	String	实例所属可用区ID
VpcId	是	否	String	私有网络ID
SubnetId	是	否	String	子网ID
Desc	是	否	String	集群描述
MasterInstanceInfo	否	否	InstanceInfo	master 节点参数
PortalInstanceInfo	是	否	InstanceInfo	portal 节点参数
WorkerInstanceInfo	否	否	InstanceInfo	worker 节点参数
DisasterRecoverGroupIds	否	否	Array of String	置放群组id，仅支持指定一个。

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，每次请求都会返回。定位问题时需要提供该次请求的 RequestId。

4. 错误码

该接口暂无业务逻辑相关的错误码，其他错误码详见公共错误码。

获取集群详情

最近更新时间: 2024-09-11 15:04:50

1. 接口描述

接口请求域名： tbds.api3.finance.cloud.tencent.com。

获取集群详情

默认接口请求频率限制：20次/秒。

接口更新时间：2020-09-07 19:44:50。

接口只验签名不鉴权。

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	允许NULL	类型	描述
Action	是	否	String	公共参数，本接口取值：DescribeClusterDetail
Version	是	否	String	公共参数，本接口取值：2020-01-16
Region	是	否	String	公共参数，本接口不需要传递此参数。
ClusterId	是	否	String	集群id

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，每次请求都会返回。定位问题时需要提供该次请求的 RequestId。

4. 错误码

该接口暂无业务逻辑相关的错误码，其他错误码详见公共错误码。

获取集群节点

最近更新时间: 2024-09-11 15:04:50

1. 接口描述

接口请求域名： tbds.api3.finance.cloud.tencent.com。

获取集群节点.

默认接口请求频率限制：20次/秒。

接口更新时间：2020-09-09 11:24:54。

接口只验签名不鉴权。

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	允许NULL	类型	描述
Action	是	否	String	公共参数，本接口取值：DescribeClusterNodes
Version	是	否	String	公共参数，本接口取值：2020-01-16
Region	是	否	String	公共参数，本接口不需要传递此参数。
ClusterId	是	否	String	集群id
PageIndex	否	否	Uint64	从0开始的页面index
PageSize	否	否	Uint64	页面大小

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，每次请求都会返回。定位问题时需要提供该次请求的 RequestId。

4. 错误码

该接口暂无业务逻辑相关的错误码，其他错误码详见公共错误码。

销毁集群

最近更新时间: 2024-09-11 15:04:50

1. 接口描述

接口请求域名： tbds.api3.finance.cloud.tencent.com。

销毁集群

默认接口请求频率限制：20次/秒。

接口更新时间：2020-09-07 15:49:00。

接口只验签名不鉴权。

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	允许NULL	类型	描述
Action	是	否	String	公共参数，本接口取值：DestroyCluster
Version	是	否	String	公共参数，本接口取值：2020-01-16
Region	是	否	String	公共参数，本接口不需要传递此参数。
ClusterId	是	否	String	集群id

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，每次请求都会返回。定位问题时需要提供该次请求的 RequestId。

4. 错误码

该接口暂无业务逻辑相关的错误码，其他错误码详见公共错误码。

获得集群列表

最近更新时间: 2024-09-11 15:04:50

1. 接口描述

接口请求域名： tbds.api3.finance.cloud.tencent.com。

无

默认接口请求频率限制：20次/秒。

接口更新时间：2021-12-08 10:41:45。

接口只验签名不鉴权。

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	允许NULL	类型	描述
Action	是	否	String	公共参数，本接口取值：GetClusters
Version	是	否	String	公共参数，本接口取值：2020-01-16
Region	是	否	String	公共参数，本接口不需要传递此参数。
PageIndex	否	否	Uint64	从0开始的页面index
PageSize	否	否	Uint64	页面大小
Keyword	否	否	String	关键字，支持|分隔多个关键字
Type	否	否	String	类型，填充“cvm”或“bms”

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，每次请求都会返回。定位问题时需要提供该次请求的 RequestId。

4. 错误码

该接口暂无业务逻辑相关的错误码，其他错误码详见公共错误码。

获得集群部署进度

最近更新时间: 2024-09-11 15:04:50

1. 接口描述

接口请求域名： tbds.api3.finance.cloud.tencent.com。

无

默认接口请求频率限制：20次/秒。

接口更新时间：2020-03-02 10:56:35。

接口只验签名不鉴权。

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	允许NULL	类型	描述
Action	是	否	String	公共参数，本接口取值：GetProgress
Version	是	否	String	公共参数，本接口取值：2020-01-16
Region	是	否	String	公共参数，本接口不需要传递此参数。
ClusterId	是	否	String	集群ID

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，每次请求都会返回。定位问题时需要提供该次请求的 RequestId。

4. 错误码

该接口暂无业务逻辑相关的错误码，其他错误码详见公共错误码。

bms创建实例接口测试

最近更新时间: 2024-09-11 15:04:50

1. 接口描述

接口请求域名： tbds.api3.finance.cloud.tencent.com。

bms创建实例接口测试

默认接口请求频率限制：20次/秒。

接口更新时间：2021-11-23 14:35:40。

接口只验签名不鉴权。

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	允许NULL	类型	描述
Action	是	否	String	公共参数，本接口取值：TestApplyBms
Version	是	否	String	公共参数，本接口取值：2020-01-16
Region	是	否	String	公共参数，本接口不需要传递此参数。
Name	是	否	String	集群的名称
Creator	否	否	String	创建人
ZoneId	是	否	String	实例所属可用区ID
VpcId	是	否	String	私有网络ID
SubnetId	是	否	String	子网ID
Desc	是	否	String	集群描述
Password	否	否	String	主机密码
InstanceName	否	否	String	实例名称
HostName	否	否	String	主机名称
PrivateIpAddresses	否	否	Array of String	服务器内网ip数组
InternetAccessible	否	否	InternetAccessible	公网带宽信息
InstanceCount	否	否	Uint64	实例个数
FlavorId	否	否	String	套餐id
OperatingSystemType	否	否	String	bms,系统类型，例如：Linux

参数名称	必选	允许NULL	类型	描述
OperatingSystem	否	否	String	bms,系统名称
EnhancedService	否	否	Bool	bms,开启主机安全服务
RaidType	否	否	String	RAID类型存储类型

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，每次请求都会返回。定位问题时需要提供该次请求的 RequestId。

4. 错误码

该接口暂无业务逻辑相关的错误码，其他错误码详见公共错误码。

数据结构

最近更新时间: 2024-09-11 15:04:50

SSLVpnServer

SSL VPN服务端实例。

被如下接口引用：

名称	必选	允许NULL	类型	描述
VpcId	是	否	String	vpcid
VpnSslServerId	是	否	String	ssl vpn server id
VpnGatewayId	是	否	String	vpngw id
SSLVpnServerName	是	否	String	SSLVpnServerName
LocalAddress	是	否	String	本段地址
RemoteAddress	是	否	String	对端地址
MaxConnection	是	否	Int64	连接数
WanIp	是	否	String	WanIp

DataDisk

数据盘对象

被如下接口引用：BmsCreateCluster、CreateCluster

名称	必选	允许NULL	类型	描述
DiskType	是	否	String	数据盘类型
DiskSize	是	否	Uint64	数据盘大小，单位：GB。最小调整步长为10G，不同数据盘类型取值范围不同，具体限制详见：CVM实例配置。默认值为0，表示不购买数据盘。更多限制详见产品文档
DeleteWithInstance	是	否	Bool	数据盘是否随子机销毁。TRUE：子机销毁时，销毁数据盘，只支持按小时后付费云盘。FALSE：子机销毁时，保留数据盘。默认取值：TRUE

BmsInstanceInfo

bms 实例信息

被如下接口引用：BmsCreateCluster

名称	必选	允许NULL	类型	描述
FlavorId	是	否	String	bms机型
InstanceCount	是	否	Uint64	要申请创建的BMS实例个数
RaidType	是	否	String	RAID类型存储类型

InstanceInfo

CVM实例信息

被如下接口引用：BmsCreateCluster、CreateCluster

名称	必选	允许NULL	类型	描述
InstanceType	是	否	String	CVM实例类型
InstanceCount	是	否	Uint64	要申请创建的CVM实例个数
DiskType	是	否	String	系统盘类型
SystemDiskSize	是	否	Uint64	系统盘大小，单位：GB。默认值为 50
LocalStorageDiskCount	否	否	Uint64	本地磁盘的个数
DataDisks	否	否	Array of DataDisk	数据盘
DisasterRecoverGroupIds	否	否	Array of String	置放群组id，仅支持指定一个。

InternetAccessible

公网带宽信息

被如下接口引用：TestApplyBms

名称	必选	允许NULL	类型	描述
PublicIpAssigned	否	否	Bool	是否分配公网ip
InternetMaxBandwidthOut	否	否	String	带宽大小
InternetServiceProvider	否	否	String	外网供应商

错误码

最近更新时间: 2024-09-11 15:04:50

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
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接口鉴权文档检查签名计算过程。
AuthFailure.TokenFailure	token 错误。
AuthFailure.UnauthorizedOperation	请求未 CAM 授权。
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。
InvalidParameterValue	参数取值错误。

错误码	说明
LimitExceeded	超过配额限制。
MissingParameter	缺少参数错误。
NoSuchVersion	接口版本不存在。
RequestLimitExceeded	请求的次数超过了频率限制。
ResourceInUse	资源被占用。
ResourceInsufficient	资源不足。
ResourceNotFound	资源不存在。
ResourceUnavailable	资源不可用。
UnauthorizedOperation	未授权操作。
UnknownParameter	未知参数错误。
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