目录

创建API分组

操作场景

创建API前，需要先创建API分组。API分组相当于API的集合，API提供者以API分组为单位，管理分组内的所有API。

操作步骤

1. 登录管理控制台，进入到【API网关】服务管理界面。
2. 选择菜单“开放API > API分组”，进入到API分组信息页面。
3. 单击“创建分组”，弹出“创建分组”对话框中填写分组名称和描述。
4. 完成分组信息填写后，单击“确定”，创建API分组。创建分组成功后，在“API分组”页面的列表中显示新创建的API分组。

- API分组创建后，系统为分组自动分配一个子域名，此子域名。此子域名可用于内部测试以及对外开放使用。
- 对外开放API时，暂不支持为API分组绑定您自己的独立域名。

环境变量管理

环境变量介绍

当前API支持发布至三个环境：测试（dev）、预发布（pre_release）、线上（release），同时API分组管理支持对环境变量进行管理，实现同一个API，在不同环境中调用不同的后端服务。

例如：API测试环境，后端服务对应到您的测试环境资源，从而可以保证在同一套API配置的情况下，供您的测试人员进行测试使用。

在API网关上进行环境管理的时候，您需要做两部分工作：

1. API的后端配置：通过设置 API分组 的 环境变量 ，为API分组的测试、预发、线上环境分别定义不同值，从而当调用API时，API网关可以调用到不同的后端地址。
2. API的前端调用：需要client端显式的说明需要调用哪个环境。目前API网关支持在 Header 中增加入参 X-KSCAPIGW-ENV 的信息。

下文将举例介绍后端服务类型为HTTP的情况下， 如何使用在Header 中增加入参 X-KSCAPIGW-ENV方式，从而实现环境变量管理。

操作说明

1.前期准备

准备2个后端服务，使用http访问时，分别会返回{“env”:”test env”}和{“env”:”relese env”}

2.API后端配置

1）在API网关控制台中创建API分组，进入环境变量管理。

2)环境变量管理中，在线上环境和测试环境中分别创建一个同名的变量(本例中为 backend-host)，但值的内容不同，分别1中准备好的2个后端服务地址。

3）在此分组下创建API，略过API的其他定义，重点在 定义API后端服务 的页面中，在 后端服务地址 的位置填写变量名称，填写 #backend-host# 。

4）完成其他API配置后发布到线上环境和测试环境中。

使用X-KSCAPIGW-ENV访问环境

1）线上环境调用。直接发起 API 调用，即调用线上环境。

2）预发环境调用。调用预发环境的API，则在调用API时，在 Header 中增加入参 X-KSCAPIGW-ENV: pre_release， 即可访问预发环境的 API。

3）测试环境调用。调用测试环境的 API，则在调用 API 时，在 Header 中增加入参 X-KSCAPIGW-ENV: dev， 即可访问测试环境的 API。

创建API

操作说明

- 创建 API 需要填写API的定义信息：基本信息、定义API请求、定义后端服务和定义返回结果。
- API 创建成功后需要对API进行调试。经测试证明 API可用后，可发布上线供用户使用。

在 API 网关控制台中 API 列表 页面，单击 创建 API，即进入 API 的创建和定义流程。

设置基本信息

1）API名称：API 名称标识需在所属分组内具有唯一性

2）API 分组：分组是 API 的管理单元。创建 API 之前，您需要先创建分组。

3）类型

- 公开 类型的 API：所有用户均可以在 API 网关控制台，已发布 API 页面看到 API 的部分信息。公开 类型的 API 会跟随 API 分组上架到云市场，供用户购买和调用。
- 私有 类型的 API：不能上架到云市场中售卖。如果有用户想要调用您的私有类型的 API ，需要您主动操作授权，否则用户无渠道获取 API 信息。

4）安全认证

- APP认证：要求请求者调用该 API 时，需通过对 APP 的身份认证。由API网关服务负责接口请求的安全认证。
- IAM认证：表示借助IAM服务进行安全认证。
- 无认证：即任何人知晓该 API 的请求定义后，均可发起请求，网关不对其做身份验证，均会将请求转发至您的后端服务。（强烈不建议使用此模式。） **推荐使用APP认证方式**。

5）描述：API 功能描述。

设置API前端请求定义

这部分是定义用户如何请求您的 API ，包括协议、请求Path、HTTP Method、入参请求模式、和入参定义。

1）域名：系统自动为您分配的对应分组的子域名

2）协议：分三种HTTP、HTTPS、HTTP&HTTPS。传输重要或敏感数据时推荐使用HTTPS。

3）请求Path：Path 指相对于服务 Host， API 的请求路径。请求 Path 可以与后端服务实际 Path 不同，您可以随意撰写合法的有明确语义的 Path 给用户使用。您可以在请求 Path 中配置动态参数，即要求用户在 Path 中传入参数，同时您的后端又可以不在 Path 中接收参数，而是映射为在 Query、Header 等位置接收。

4）匹配模式

- 绝对匹配：调用的请求Path固定为创建时填写的API请求Path。
- 前缀匹配：调用的请求Path将以创建时填写的API请求Path为前缀，支持接口定义多个不同Path。
- 例如，请求路径为/test/aa，使用前缀匹配时，通过/test/aa/cc可以访问，但是通过/test/aacc无法访问。

5）Method：支持GET、POST、DELETE、PUT、PATCH、HEAD、OPTIONS、ANY。其中ANY表示该API支持任意请求方法。

6）支持CORS：是否开启跨域访问CORS（cross-origin resource sharing）。CORS允许浏览器向跨域服务器，发出XMLHttpRequest请求，从而克服了AJAX只能同源使用的限制。目前支持简单跨域请求。

7）入参定义

- 参数名：展示给用户的参数名称。
- 参数位置：参数在请求中的位置，包含path、 head、query、body。如果您在 Path 中配置了动态参数，必须存在参数位置为path的同名参数。
- 类型：字段的类型，支持：string、int、boolean。
- 必填：指此参数是否为必填。当选择为 是 时，API 网关会校验用户的请求中是否包含了此参数。若不包含此参数，则拒绝该请求。
- 默认值：当 必填 为 否 时生效。在用户请求中不包含此参数时，API 网关自动添加默认值发送给后端服务。
- 示例：指参数的填写示例。
- 描述：参数的用途描述及使用注意事项等。

8）请求body 当body非表单选项，比如二进制、json时。可以填写如下信息：

- body内容描述：用于描述body的数据格式
- 模型：可以选择此分组下的模型

定义后端服务信息

这部分主要是定义一些参数的前后端映射，即 API 后端服务的配置，包括后端服务地址、后端Path、后端超时时间、参数映射、常量参数、系统参数。用户请求到达 API 网关后，API 网关会根据您的后端配置，映射为对应的后端服务的请求形式，请求后端服务。

1)后端服务类型：目前支持 HTTP/HTTPS和MOCK两种类型。

2)后端服务地址:格式：“主机:端口”，主机为IP地址/域名，未指定端口时，HTTP协议默认使用80端口，HTTPS协议默认使用443端口。端口范围：1 ~ 65535。如果需要创建变量标识，则填写“#变量名#”，如#ipaddress#。

3)后端请求Path：Path 是您的 API 服务在您后端服务器上的实际请求路径，可以包含路径参数，以{路径参数}形式表示，比如/getUserInfo/{userId}。

如果请求路径中含有环境变量，则使用#变量名#的方式将环境变量定义到请求路径中，如/#path#。

4）后端认证：支持无认证和密钥认证。推荐使用密钥认证。

5）后端请求超时/读超时：指 API 请求到达 API 网关后，API 网关调用 API 后端服务的响应时间，即由 API 网关请求后端服务开始直至 API 网关收到后端返回结果的时长。单位为毫秒。设置值不能超过 30 秒。如果响应时间超过该值，API 网关会放弃请求后端服务，并给用户返回相应的错误信息。

6）mock返回结果：当为mock模式时，无1-5项，只需填写此项。Mock一般用于开发调试验证。在项目初始阶段，后端服务没有搭建好API联调环境，可以使用Mock模式，将预期结果固定返回给API调用方，方便调用方进行项目开发。

7）配置后端服务参数：可根据后端服务实际的参数名称和参数位置修改映射关系；包括参数名称和参数位置。

8）配置常量参数：您可以配置常量参数。您配置常量参数对您的用户不可见，但是 API 网关会在中转请求时，将这些参数加入到请求中的指定位置，再传递至后端服务，实现您的后端的一些业务需求。比如，您需要 API 网关每次向后端服务发送请求都带有一个关键词 ksygateway，您就可以把 ksygateway 配置为常量参数，并指定接收的位置。

9）配置系统参数：指 API 网关的系统参数。默认系统参数不会传递给您，但是如果您需要获取系统参数，您可以在 API 里配置接收位置和名称。具体内容如下表：

注：您需确保您录入的所有参数的参数名称全局唯一，包括 Path 中的动态参数、Headers 参数、Query 参数、Body 参数（非二进制）、常量参数、系统参数。如果您同时在 Headers 和 Query 里各有一个名为 name 的参数，将会导致错误。

定义返回结果

录入正确示例、失败返回结果示例、和错误码定义。

录入后单击完成，即完成API服务的创建。

后续操作

您创建完API后，可以通过调试API，验证服务是否正常。

调试API

操作场景

API创建后需要验证服务是否正常，管理控制台提供调试功能，您可以添加HTTP头部参数与body体参数，调试API接口。

- 后端路径中含有环境变量的API，不支持调试。
- API绑定签名密钥时，不支持调试。
- 如果API已绑定流控策略，在调试API时，流控策略无效。

前提条件

- 已创建API分组和分组内的API。
- 已搭建完成后端服务。

操作说明

（1）登录API网关控制台，点击菜单 **开放API**> API管理，进入到API管理信息页面。

（2）通过以下任意一种方法，进入API调试页面。

- 在待调试的API所在行，在操作列点击 **调试**按钮。
- 单击“**API名称**”，进入API详情页面。在页面右上角单击“**调试**”。

（3）添加请求参数后，单击“发送请求”。 右侧返回结果回显区域打印API调用的Response信息。

- 调用成功时，返回HTTP状态码为“200”和Response信息。
- 调试失败时，返回HTTP状态码为4xx或5xx，具体错误信息请参见错误代码表。

后续步骤

完成以上定义后和初步调试后，您就完成了 API 的创建。您可以发布 API 到测试、预发、线上环境，继续调试或供用户使用。还可以为 API 绑定 签名密钥、访问控制 和 流量控制 等安全配置。

发布API

操作场景

创建完成的API，支持发布到不同的环境。API只有在发布到环境后，才支持被调用。API网关支持查看API发布历史（版本号、发布说明、发布时间和发布环境），并支持回滚到不同的API历史版本。

已发布的API，在修改信息后，需要重新发布才能将修改后的信息同步到环境中。

前提条件

- 已创建API分组和分组内的API。
- 已搭建完成后端服务。

操作说明

1）单击“开放API > API管理”，进入到API管理信息页面。

2）通过以下任意一种方法，进入“发布API”页面。

- 在待发布的API所在行，单击“发布”。
- 单击“API名称”，进入API详情页面。在右上角单击“发布”。
- 如果需要批量发布API，则勾选待发布的API，单击“批量发布”。

3）在弹出的发布弹窗里面选择API需要发布到的环境，并填写发布说明。如果API在选择的环境中已发布，再次发布即为覆盖该环境的API。

4）单击“发布”，完成API发布。

下线API

操作场景

- 已发布的API因为其他原因需要暂停对外提供服务，可以暂时将API从相关环境中下线。
- 该操作将导致此API在指定的环境无法被访问，请确保已经告知使用此API的用户。

前提条件

- 已创建API分组和分组内的API。
- API已发布到该环境。

操作说明

1）单击“开放API > API管理”，进入到API管理信息页面。

2）通过以下任意一种方法，进入“下线API”页面。

- 在待发布的API所在行，单击“下线”。

- 单击“API名称”，进入API详情页面。在右上角单击“下线”。
- 如果需要批量发布API，则勾选待发布的API，单击“批量下线”。

3）在弹出的发布弹窗里面选择API需要下线的环境。

4）单击“确定”，完成API下线。

授权API

操作场景

API在创建后，通过指定授权给某些应用，让指定应用能够调用API。

说明：

- 仅在API发布到环境后，才支持被调用。
- 仅在API为APP认证时，才支持通过应用调用API。

前提条件

- 已创建API分组和分组内的API。
- API已发布至对应环境。
- 已创建应用。

操作步骤

1）单击“开放API > API管理”，进入到API管理信息页面。

2）通过以下任意一种方法，进入“授权应用”页面。

- 在待授权的API所在行，单击“授权”，进入“授权API”页面。单击“添加应用”，弹出“授权应用”对话框。
- 通过API详情页面进入，进入“授权信息”tab页面。单击“添加应用”，弹出“授权应用”对话框。

3）选择API授权环境、授权时间，查询并勾选应用后，单击“授权”

4）授权成功后，可以在“授权信息”/“授权API”中查看已授权的应用。

后续操作

您将API授权给指定应用后，可以通过不同语言的SDK调用此API。

开启跨域访问

什么是跨域访问

浏览器出于安全性考虑，会限制从页面脚本内发起的跨域访问（CORS）请求，此时页面只能访问同源的资源，而CORS允许浏览器向跨域服务器，发送XMLHttpRequest请求，从而实现跨域访问。

浏览器将CORS请求分为两类：简单请求和非简单请求。

当请求同时满足下面三个条件时，CORS验证机制会使用简单模式进行处理。

1.请求方法是下列之一： GET HEAD POST

2.请求头中的Content-Type请求头的值是下列之一： application/x-www-form-urlencoded multipart/form-data text/plain

3.Fetch规范定义了CORS安全头的集合（跨域请求中自定义的头属于安全头的集合）该集合为： Accept Accept-Language Content-Language Content-Type （需要注意额外的限制） DPR Downlink Save-Data Viewport-Width Width

不满足以上2个条件的，都为非简单请求。对于非简单请求，在正式通信之前，浏览器会增加一次HTTP查询请求，称为预检请求。浏览器询问服务器，当前页面所在的源是否在服务器的许可名单之中，以及可以使用哪些HTTP请求方法和头信息字段。预检通过后，浏览器向服务器发送简单请求。

在API网关实现简单模式CORS跨域请求

API网关默认所有API允许跨域访问，因此如果用户的API后端服务的应答中不做特殊返回，API网关会返回允许所有域跨域访问的相关头，下面是一个示例：

客户端的API请求

```
GET /simple HTTP/1.1
Host: ksyun.com
orgin: http://www.ksyun.com
content-type: application/x-www-form-urlencoded; charset=utf-8
accept: application/json; charset=utf-8
date: Mon, 18 Sep 2019 12:50:23 GMT
```

后端服务应答

```
HTTP/1.1 200 OK
Date: Mon, 18 Sep 2019 12:50:23  GMT
Content-Type: application/json; charset=UTF-8
Content-Length: 12
{"200","OK"}
```

API网关应答

```
HTTP/1.1 200 OK
Date: Mon, 18 Sep 2019 12:50:23  GMT
Access-Control-Allow-Origin: *
x-kscapigw-request-id: 104735BD-8968-458F-9929-DBFA43F324C6
Content-Type: application/json; charset=UTF-8
```

```
Content-Length: 12
{"200","OK"}
```

从上面三个报文可以看出，API网关会对用户的后端服务应答做一定修改，增加一个跨域头：

```
Access-Control-Allow-Origin: *
```

这个跨域头的意思是，本API允许所有域的请求访问。

如果用户需要定制针对简单请求的应答的跨域头，只需要在后端服务应答中，增加Access-Control-Allow-Origin这个跨域头即可，后端服务应答中的头会默认覆盖掉API网关自己增加的头。下面是一个例子，这个例子中的API只允许http://www.ksyun.com 这一个域访问：

客户端的API请求

```
GET /simple HTTP/1.1
Host: www.ksyun.com
orgin: http://www.ksyun.com
content-type: application/x-www-form-urlencoded; charset=utf-8
accept: application/json; charset=utf-8
date: Mon, 18 Sep 2019 12:50:23  GMT
```

后端服务应答

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin:http://www.kysun.com
Date: Mon, 18 Sep 2019 12:50:23  GMT
Content-Type: application/json; charset=UTF-8
Content-Length: 12
{"200","OK"}
```

API网关应答

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: http://www.ksyun.com
x-kscapigw-request-id: 104735BD-8968-458F-9929-DBFA43F324C6
Date: Mon, 18 Sep2019 12:50:23  GMT
Content-Type: application/json; charset=UTF-8
Content-Length: 12
{"200","OK"}
```

Mock模式配置

在项目开发过程中，往往是多个合作方一同开发，多个合作方相互依赖，而这种依赖在项目过程中会造成相互制约，理解误差也会影响开发进度，甚至影响项目的工期。所以在开发过程中，一般都会使用 Mock 来模拟最初预定的返回结果，来降低理解偏差，从而提升开发效率。

API 网关也支持 Mock 模式的简单配置。

配置 Mock

1）在 API 编辑页面——后端基础定义，选择后端服务类型为Mock。

2）填写 Mock 返回结果。Mock 返回结果，可以填写您真实的返回结果。目前支持是 Json格式作为 Mock 返回结果。

保存后 Mock 设置成功，请根据实际需要 发布 到测试或线上环境进行测试，也可以在 API 调试页面进行调试。

创建流控策略

简介

为了保护API开放者的后端服务不因过于频繁的调用导致负载过高，API网关提供了流量控制的功能来限制API的调用。

- 流量控制策略与API独立管理，流量控制策略配置完成后可绑定到不同的API上，对多个API同时生效。
- 流量控制策略可以配置对 API、用户、应用三个对象的流控值，流控的单位可以是分钟、小时、天。
- 同一个环境中，一个API只能被一个流控策略绑定，但一个流控策略可以绑定多个API。
- 已有的流量控制策略上，可以额外配置特殊用户和特殊应用（APP），这些特例也是针对当前策略已绑定的API生效。

创建流控策略

1. 登录管理控制台，进入到【API网关】服务管理界面。
2. 选择菜单“开放API > 流量控制”，进入到流量控制页面。
3. 单击“新建流控策略”，弹出对话框如下。

API流量限制 该策略绑定的API在对应环境的单位时间内被调用的次数不能超过设定值，单位时间可选秒、分钟、小时、天，如5000次/分钟。

应用流量限制 每个APP对该策略绑定的任何一个API在对应环境的单位时间内的调用次数不能超过设定值。如10000次/小时。

用户流量限制 每个金山云账号对该策略绑定的任何一个 API在对应环境的单位时间内的调用次数不能超过设定值。如 1000000 万次/天。

在一个流控策略里面，这三个值可以同时设置。请注意，用户流量限制应不大于 API 流量限制，APP 流量限制应不大于用户流量限制。即 APP 流量限制 <= 用户流量限制 <= API 流量限制。

4. 完成信息填写后，单击“确定”，创建流控策略。创建成功后在流控策略页面的列表中显示新创建的流控策略

添加特殊应用或用户

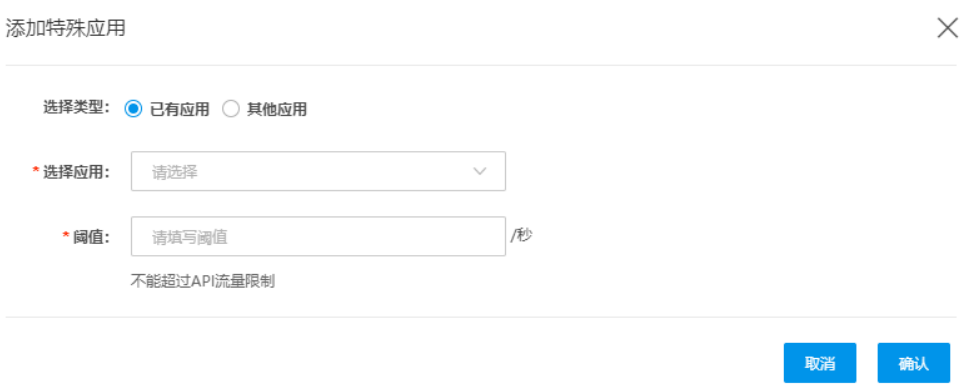
操作场景

如果需要为某个应用/用户设置特定的流控值，则通过添加特殊应用/用户可以实现。

添加特殊应用

1. 单击“开放API > 流量控制”，进入到流量控制信息页面。
2. 单击待添加特殊应用的流控策略的名称，进入流控详情页面。
3. 进入“特殊应用”tab页面，点击“添加特殊应用”

4. 在弹出“添加特殊应用”对话框，按照以下两种方式添加特殊应用。

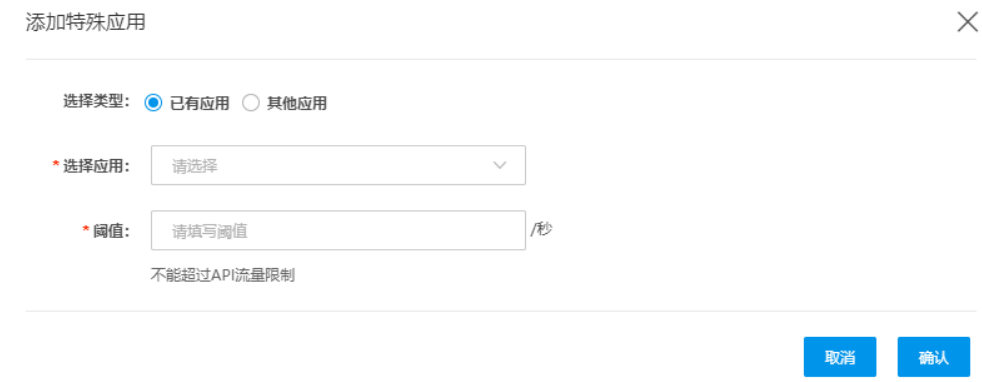


- 添加已有应用：单击“已有应用”，选择已有应用，输入阈值。
- 添加其他应用：单击“其他应用”，输入其他用户的应用ID和阈值。

添加特殊用户

1. 单击“开放API > 流量控制”，进入到流量控制信息页面。
2. 单击待添加特殊应用的流控策略的名称，进入流控详情页面。
3. 进入“特殊金山云用户”tab页面，点击“添加金山云用户”

4. 在弹出“添加金山云用户”对话框，填写金山云用户ID和阈值进行提交。



添加访问控制策略

简介

访问控制策略是API网关提供的API安全防护组件之一，主要用来控制访问API的IP地址和帐户，您可以通过设置IP地址或帐户的黑白名单来允许/拒绝某个IP地址或帐户访问API。

- 访问控制策略与API独立管理，访问控制策略配置完成后可绑定到不同的API上，对多个API同时生效。
- 同一个环境中，一个API只能被一个流控策略绑定，但一个流控策略可以绑定多个API。

创建流控策略

1. 登录管理控制台，进入到【API网关】服务管理界面。
2. 选择菜单“开放API > 访问控制”，进入到访问控制页面。
3. 单击“新建访问控制策略”，弹出对话框如下。

限制类型

- IP地址：允许/禁止访问API的IP地址。
- 账号名：允许/禁止访问API的金山云账号。

动作

允许：白名单，允许访问的IP地址或金山云账户

禁止：黑名单，禁止访问的IP地址或金山云账户

注：可以在账号管理>账号及安全页面查看对应金山云的账号ID

4. 完成信息填写后，单击“确定”，完成访问控制策略创建。

创建签名密钥

什么是签名密钥

签名密钥用于后端服务验证API网关的身份，在API网关请求后端服务时，保障后端服务的安全。。 签名密钥是由的一对 Key 和 Secret组成，您将密钥绑定到 API 之后，由网关抛向您服务后端的该 API 的请求均会加上签名信息。您需要在后端做对称计算来解析签名信息，从而验证网关的身份。

- 密钥只能被绑定到同一个 Region 下的API上。
- 一个 API的某个环境 仅能绑定一个密钥，密钥可以被替换和修改，可以与 API 绑定或者解绑。

新建密钥

1. 登录管理控制台，进入到【API网关】服务管理界面。
2. 选择菜单“开放API > 签名密钥”，进入到签名密钥页面。
3. 单击“新建密钥”，弹出对话框如下。

4. 点击确定，完成密钥创建。

签名密钥使用

域名

使用api定义中的后端请求地址

公共Header参数

名称	描述	是否参与计算签名
x-kscapigw-proxy-signature-ak	ak 用户在控制台建立的签名密钥Key	是
x-kscapigw-proxy-signature-nonce	请求调用者生成的uuid，为了避免重放	是
x-kscapigw-proxy-signature-timestamp	时间，UTC格式，例如：2020-03-13T17:18:36Z	是
x-kscapigw-proxy-signature-headers	参与计算签名的header，多个header使用英文逗号分隔	否
x-kscapigw-proxy-signature	签名	否

签名算法

1）根据请求参数(公共header参数和api参数，不包含公共header参数x-kscapigw-proxy-signature)构造规范化请求字符串：CanonicalizedQueryString api参数包含：所有query,path,body,header(x-kscapigw-proxy-signature-headers标识参与计算签名的header)参数

第一步：请求参数排序。排序规则以参数名按照字典排序

第二步：请求参数编码。使用UTF-8字符集对每个请求参数的名称和参数取值进行URLEncode，一般在URLEncode后需对三种字符替换：加号（+）替换成 %20、星号（*）替换成 %2A、 %7E 替换回波浪号（~）

第三步：每对URLEncode后的参数名称和参数值，用=进行连接。每对之间使用&进行连接。得到规范化请求字符串CanonicalizedQueryString。

2）计算签名。 sign = hash_hmac('sha256', CanonicalizedQueryString, sk)

sign值为签名算法返回的16进制格式小写字符串

签名样例：
88b203541ce8c757d7d554af2a25de036d3d9a636d91fb44d01bf82dc67a6941

计算签名时使用的sk为Key对应的密钥(Secret)，使用的哈希算法是：HMAC-SHA256。

传递签名

将计算的签名结果放到请求的header中，Key为：x-kscapigw-proxy-signature。