腾讯微服务平台

服务治理

产品文档

腾讯云

文档目录

服务治理
服务基本操作

最近更新时间：2018-10-11 10:01:00

服务是微服务平台管理的基本单元。服务管理包括服务的生命周期管理和服务治理两部分。
服务基本操作包括创建服务和删除服务。

创建服务

1. 登录 TSF 控制台。
2. 单击左侧导航栏的 **服务治理**，选择集群和命名空间。
3. 单击服务列表页的【新建】。

4. 设置服务的基本信息后，单击【提交】按钮。
   - **服务名称**：要创建的服务的名称，不超过 60 个字符。服务名称由小写字母、数字和 - 组成，且由小写字母开头，小写字母或数字结尾。

- **服务描述**：服务的描述信息。

删除服务

1. 单击服务列表右侧【删除】。

2. 在弹框中单击【确认】按钮。

注意：

只有当服务运行的实例数为0时，可删除服务。

服务监控

在 TSF 控制台服务治理页面可以看到线上服务的请求数、请求成功率、平均耗时等监控数据。数据统计周期都是 24 小时。

- **请求数**：对一个服务，统计其作为服务提供者，被所有消费他的服务消费者发起调用的 24 小时内总调用数。
- **请求成功率**：对于一个服务，统计 24 小时内其作为服务提供者，成功向消费他的所有服务消费者返回请求的总数比上服务请求总数。
- **平均耗时**：对于一个服务，统计 24 小时内其作为服务提供者，统计消费者从发起调用到调用返回到服务提供者的耗时平均值。

系统和业务自定义标签

最近更新时间：2019-04-10 18:30:59

标签说明

TSF 引入**标签**概念以区分不同的请求来源，TSF 标签包括系统标签和业务自定义标签。

- **系统标签**
  每一个 TSF 上运行的服务都已经被预先设置好了某些标签，如发起请求的服务消费方所在的部署组、IP、服务发起方的版本号等。

- **业务自定义标签**
  在实际的使用中，如果系统自带标签不能保证用户使用的场景，用户可以自定义标签内容。对于 Spring Cloud 应用，TSF 提供了用户配置自定义标签的 SDK，参考开发手册 参数传递；对于 Mesh 应用，用户需要在 header 中设置标签，参考 Mesh 开发使用指引-设置自定义标签。

> ⚠ **注意：**
>
> 这里的标签和腾讯云的标签产品不是同一个概念。腾讯云的标签产品是一种划分资源的方式，而 TSF 服务治理中的标签是为了区分不同的请求来源。

标签表达式

用户在控制台创建服务治理规则时，可以选择通过设置**标签表达式**区分请求来源。多个标签表达式之间是**逻辑与（AND）**的关系。例如两条标签表达式分别是：

- 系统标签主调服务名等于 consumer-demo
- 自定义标签 userid 等于123456

只有当一条请求是 consumer-demo 发出，**且**带有 userid 是123456的自定义标签时才满足上面2个标签表达式。

一条标签表达式中，逻辑关系与值的个数对应如下：

逻辑关系	值个数
包含（IN）	多个
不包含（ NOT IN）	多个
等于（==）	一个
不等于（!=）	一个
正则表达式（regex）	一个

API 列表

最近更新时间：2018-11-27 19:22:02

在服务的详情页中会显示出服务提供的 API 列表。API 列表显示服务对外提供的 API。
单击 API 进入详情页，可以查看到 API 的详细信息。

API 详情按照【应用名/版本号】划分显示了 API 的详细信息，包括：路径、方法、描述、入参、出参。其中 Models 表示参数中的复杂类型。

/v1/user/delete/user

应用名	版本号
kyson-provider	201809201738

路径 /v1/user/delete/user

方法 HEAD

描述 (无)

入参

参数名	参数位置	是否必填	类型	备注
uid	query	是	string	uid

出参

参数名	类型	备注
ResponseResult«User»	ResponseResult«User»	OK

Models

User

age	integer(int32)
height	integer(int32)
id	string
name	string
weight	integer(int32)

ResponseResult«User»

code	integer(int32)
data	User
msg	string

服务统计

最近更新时间：2019-05-08 16:11:16

操作场景

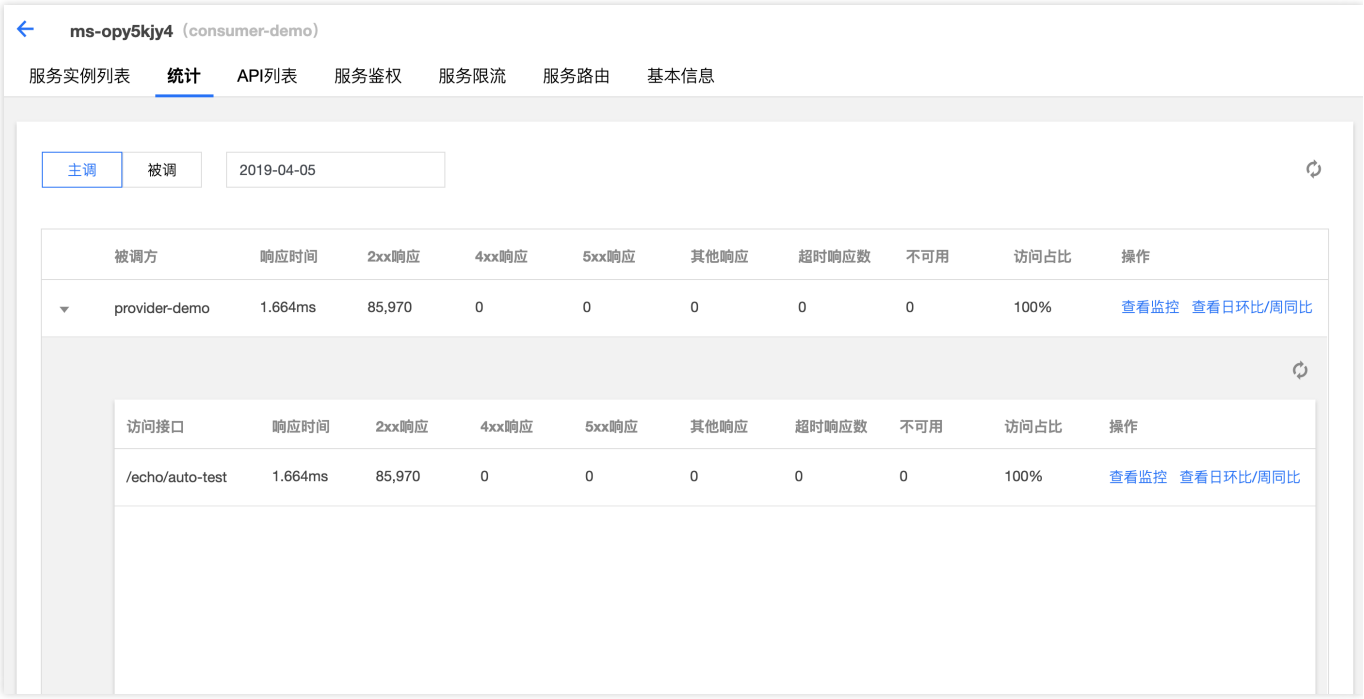
TSF 支持从主调和被调两个视角展示服务指标的统计信息。用户可以通过统计信息了解服务指标的变化情况。服务指标以天为单位进行统计，在当天只能查看**昨天之前**的统计数据。

功能说明

- 一个服务可能既是主调服务，又是被调服务。TSF 会统计服务作为主调服务调用其他服务及接口的情况，以及服务作为被调，其接口被其他服务调用的情况。
- TSF 服务统计支持查看指标的日环比和周同比。
- 统计指标：
  - 平均响应时间。
  - 按照状态码统计：2xx 响应、3xx 响应、4xx 响应、5xx 响应、其他状态码响应。
  - 按照异常请求统计：超时响应、不可用响应。其中超时响应表示服务端处理超时的请求响应，不可用响应表示服务端无可用实例时的异常请求响应。

操作步骤

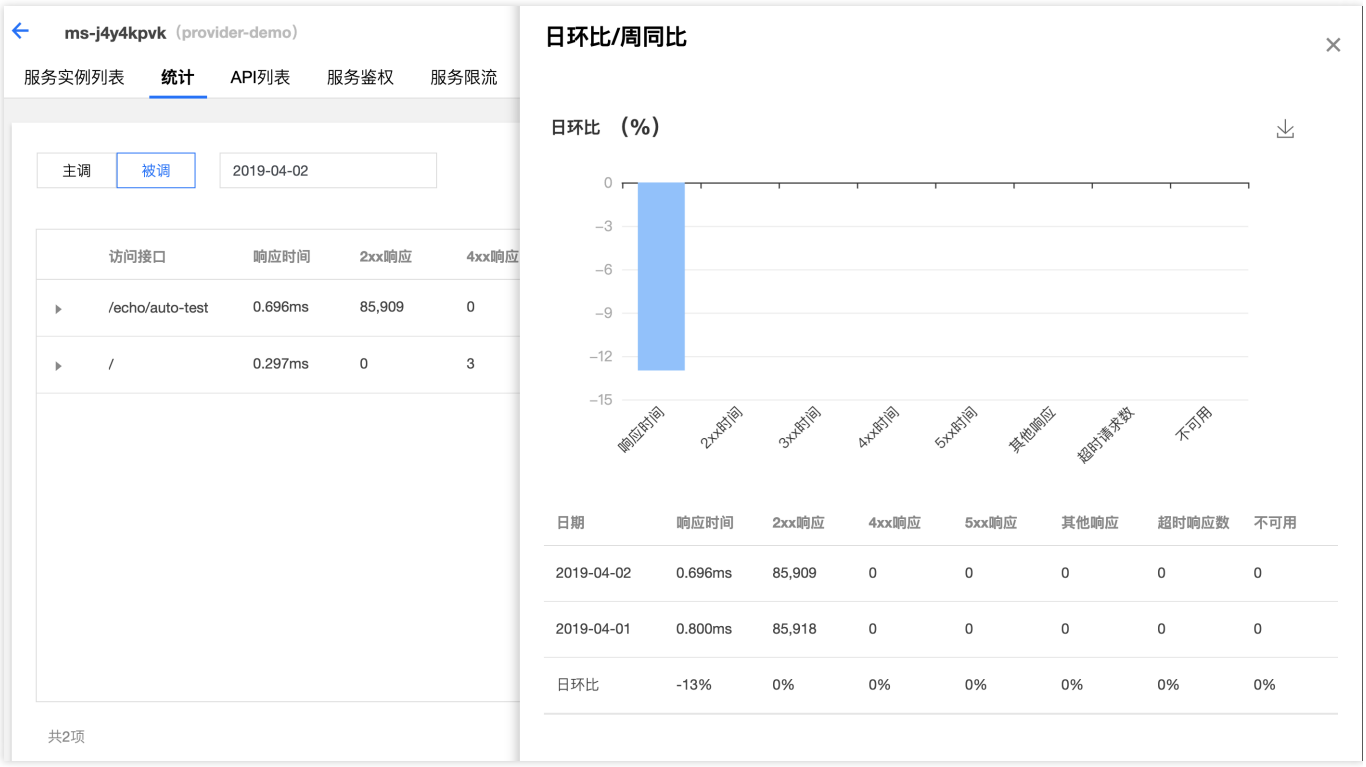
1. 登录 TSF 控制台。
2. 在左侧导航栏，单击【服务治理】，并单击某个服务的 ID 进入服务详情页。
3. 在服务详情页，单击顶部的【统计】，进入统计页面。
4. 选择主调或被调视角、统计日期。
   - **主调视角**下会展示被调方服务的指标列表。单击某个被调服务前的箭头会展示该服务不同接口被调用的统计信息。



- **被调视角**下会展示该服务的接口指标列表。单击某个接口前的箭头会展示该接口被不同主调服务调用的统计信息。

5. 单击操作列的【查看日环比/周同比】，可以查看各指标日环比/周同比情况。

- 日环比表示查询日 **T** 和之前一天 **T - 1** 的指标数据对比。



- 周同比表示查询日 **T** 和查询日前7天 **T - 7** 的指标数据对比。

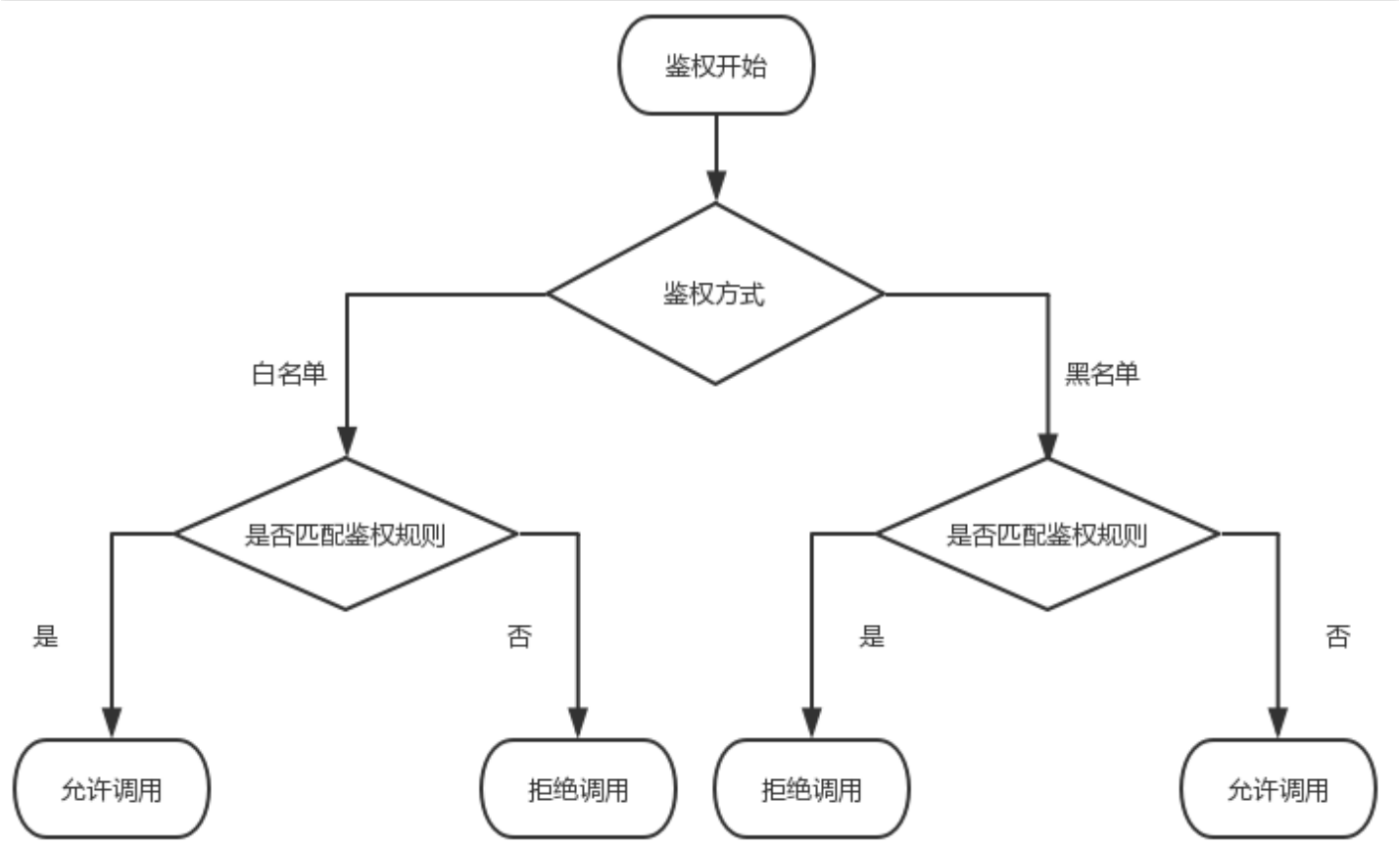
服务鉴权原理

最近更新时间：2018-11-22 11:41:28

服务鉴权是处理微服务之间相互访问权限问题的解决方案。配置中心下发鉴权规则到服务，当请求到来时，服务根据鉴权规则判断鉴权结果，如果鉴权通过，则继续处理请求，否则返回鉴权失败的 HTTP 状态码 403（Forbidden）。

鉴权原理

鉴权流程如下：



服务鉴权功能支持白名单和黑名单两种鉴权方式。

- **白名单**：当请求匹配**任意一条**鉴权规则时，允许调用；否则拒绝调用。
- **黑名单**：当请求匹配**任意一条**鉴权规则时，拒绝调用；否则允许调用。

鉴权规则

鉴权规则的关系

一个服务可能有多个鉴权规则，多个鉴权规则之间是逻辑或（OR）的关系，只要请求满足任意一条鉴权规则，就相当于匹配成功。

逻辑关系与值个数的关系

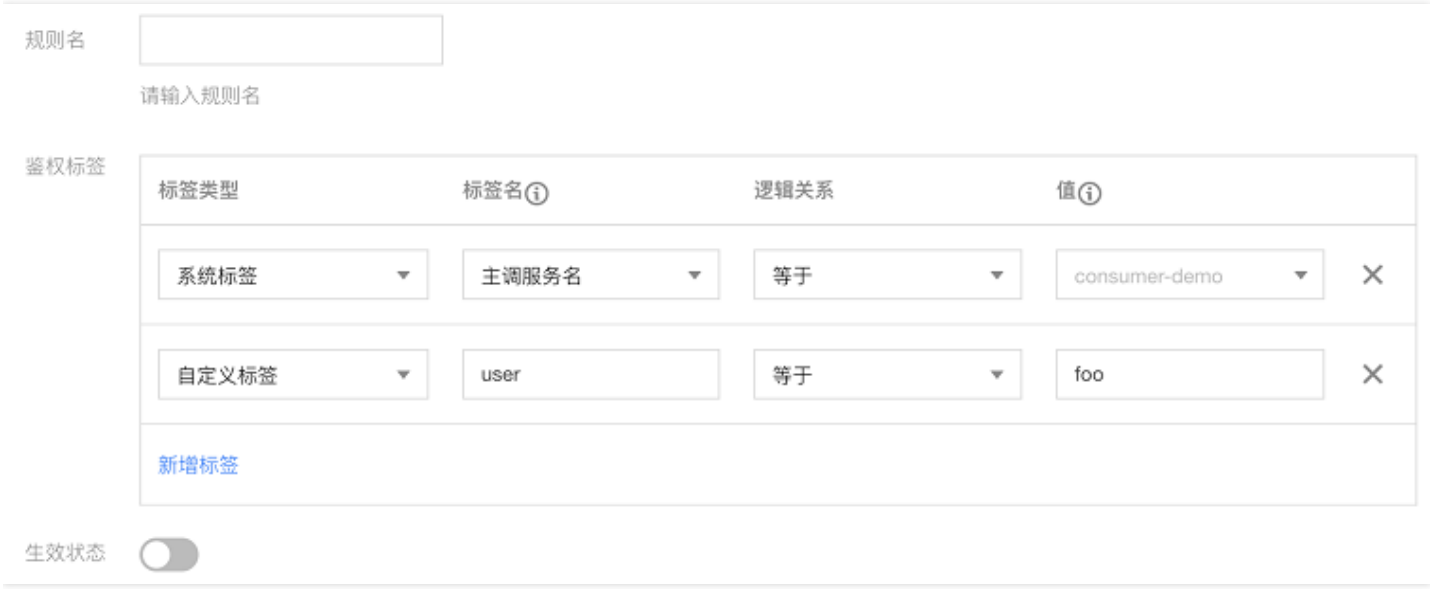
一条标签中，值的个数与逻辑关系的关系如下：

逻辑关系	值个数
包含（IN）	多个
不包含（ NOT IN）	多个
等于（==）	一个
不等于（!=）	一个
正则表达式（regex）	一个

示例说明

示例1

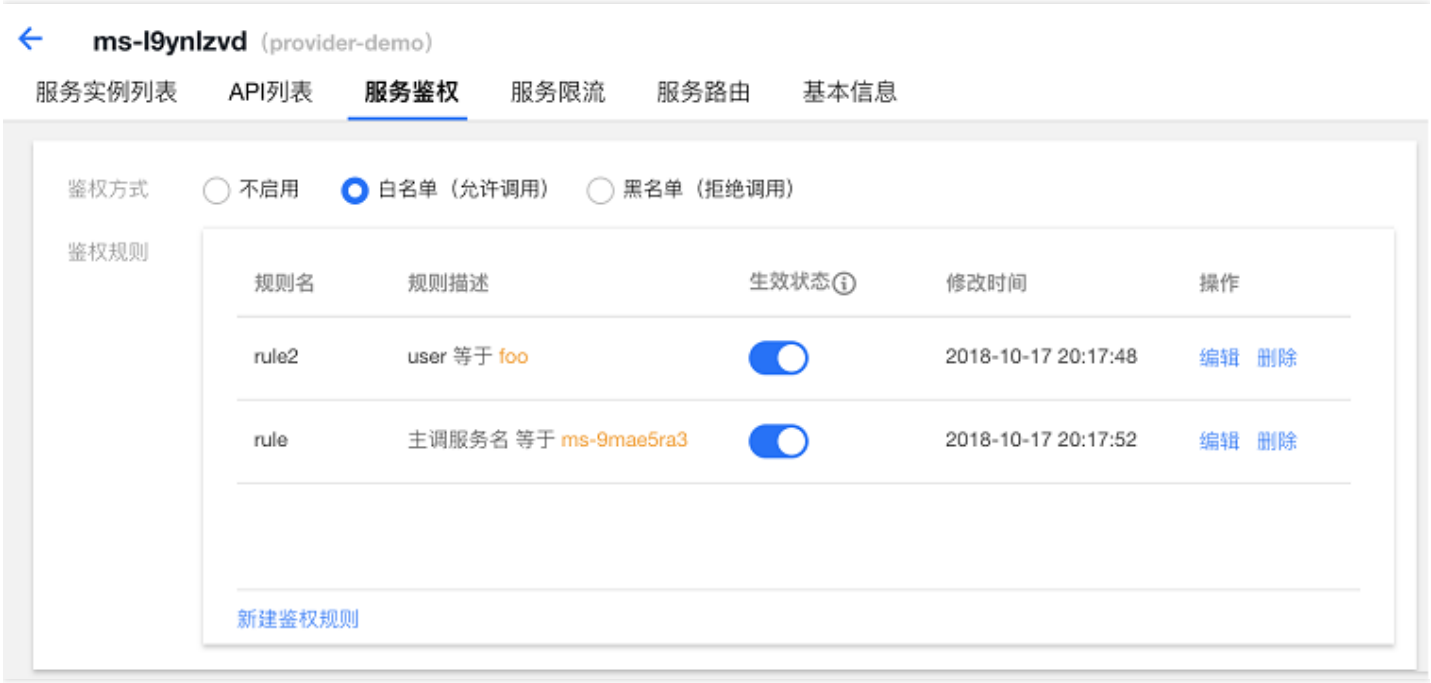
需求：服务 provider-demo 只允许来自 consumer-demo 服务**且**带有 user=foo 的自定义标签的请求调用。
解决方案：要满足上面的鉴权需求，用户可以在 provider-demo 的鉴权页面，设置鉴权方式为白名单，鉴权规则如下图（注意最后要将生效状态改为生效）：



结论：要满足 `逻辑与 AND` （既满足条件 A ，又满足条件 B）时，需要使用标签表达式。

示例2

需求：服务 provider-demo 只允许来自 consumer-demo 服务**或**带有 user=foo 的自定义标签的请求调用。
解决方案：要满足上面的鉴权需求，用户可以在 provider-demo 的鉴权页面，设置鉴权方式白名单，创建2条鉴权规则，如下图：



结论：要满足 逻辑与 OR （满足条件 A 或 条件 B）时，需要使用多条鉴权规则。

说明

白名单鉴权方式示例：
鉴权规则内容是 username 等于 foo，当请求中带有 username=foo 的 tag 时，因为匹配规则，服务允许调用；当请求中带有 username=bar 的 tag 时，因为不匹配规则，服务拒绝调用。

黑名单鉴权方式示例：鉴权规则内容是 username 等于 foo，当请求中带有 username=foo 的 tag 时，因为匹配规则，服务**拒绝**调用；当请求中带有 username=bar 的 tag 时，因为不匹配规则，服务允许调用。

服务鉴权使用说明

最近更新时间：2019-04-01 11:56:09

使用鉴权功能时，用户需要先在客户端配置依赖项，然后在 TSF 控制台设置鉴权规则。

1. 配置依赖项

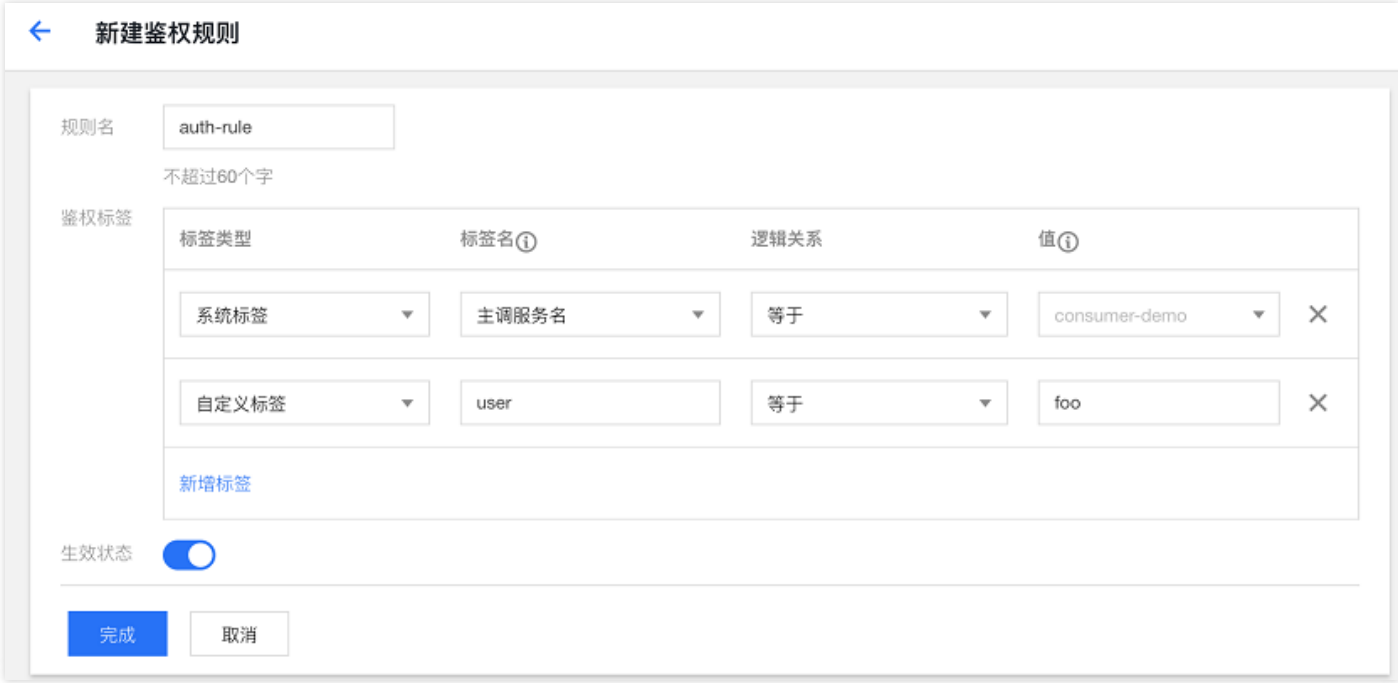
- 对于 Spring Cloud 应用，请参考开发者手册中的 服务鉴权。
- 对于 Mesh 应用，无须额外配置。

2. 设置鉴权规则

1. 登录 TSF 控制台。
2. 在左侧导航栏，单击【服务治理】。
3. 在服务列表，单击服务名，进入服务详情页。
4. 在服务详情的【服务鉴权】标签页，选择鉴权方式：
   - 不启用：关闭鉴权功能。
   - 白名单：匹配任意一条规则的请求，允许调用。
   - 黑名单：匹配任意一条规则的请求，拒绝调用。

5. 选择白名单或黑名单，单击【新建鉴权规则】按钮。

6. 在新建页面中填写规则信息，并选择规则的【生效状态】。



3. 切换鉴权方式（可选）

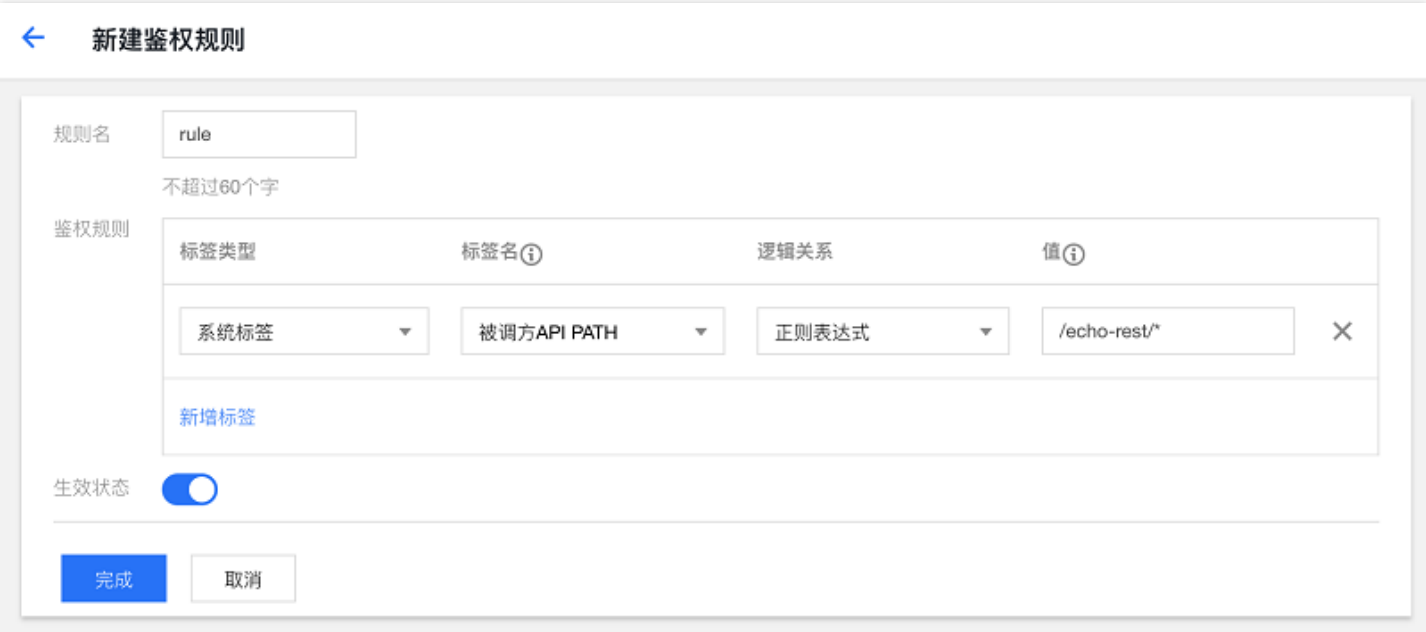
用户可以通过控制台，从一种鉴权模式切换到另外一种鉴权模式。

- 白名单切换到黑名单（或黑名单切换到白名单）：保留鉴权规则，但鉴权逻辑逆转。如果用户希望切换后，使用新的规则，则需要删除原有的，再创建新的规则。
- 白名单（或黑名单）切换到不启用：关闭鉴权功能。

4. 检查鉴权效果

以官网 Demo 为例说明如何验证鉴权功能。consumer-demo 中已包含鉴权依赖 jar 包，因此这里只需要说明在控制台上创建鉴权规则用来限制特定 API 的调用。

consumer-demo 中提供了三个 API `/echo-rest/{str}` 、`/echo-async-rest/{str}` 、`/echo-feign/{str}` 。在控制台上设置鉴权方式为**白名单**，鉴权规则的标签表达式：



创建好规则后，登录机器，使用 curl 命令来验证鉴权是否生效。

`curl IP:PORT/echo-rest/hello?user=test` 预期：正常返回

`curl IP:PORT/echo-async-rest/hello?user=test` 预期：返回鉴权失败

`curl IP:PORT/echo-feign/hello?user=test` 预期：返回鉴权失败

限制说明

等于、不等于、包含、不包含属于严格匹配，正则表达式属于模糊匹配。因此当系统标签是被调方 API PATH 时，目前仅支持使用**正则表达式**的逻辑关系来匹配带参数的 API 请求。
例如标签的逻辑关系是正则表达式，值填写 `/echo/.*` ，可以匹配带参数的请求 /echo/test123（其中test123是参数）；当标签的逻辑关系是等于、不等于、包含、不包含关系，值是 `/echo/{param}` 时，不能匹配带参数的请求 /echo/test123（其中 test123 是参数）。

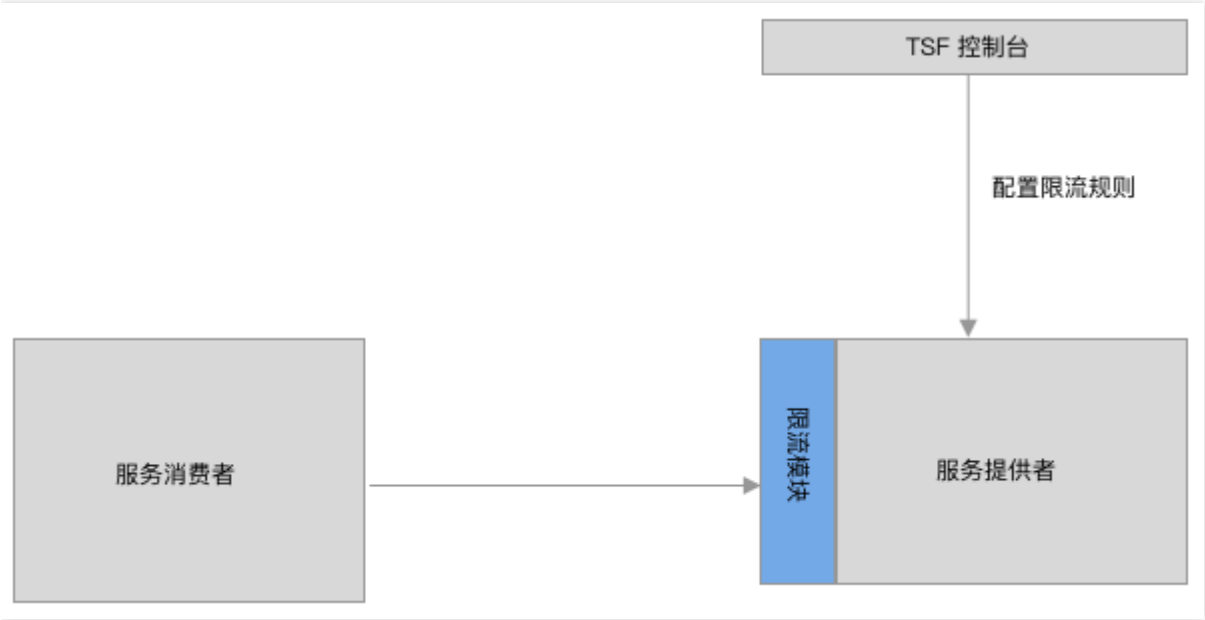
服务限流原理及使用

最近更新时间：2018-11-27 19:22:31

服务限流主要是保护服务节点或者数据节点，防止瞬时流量过大造成服务和数据崩溃，导致服务不可用。当资源成为瓶颈时，服务框架需要对请求做限流，启动流控保护机制。

限流原理

在服务提供者端配置限流依赖项，在 TSF 控制台配置限流规则。此时服务消费者去调用服务提供者时，所有的访问请求都会通过限流模块进行计算，若服务消费者调用量在一定时间内超过了预设阈值，则会触发限流策略，进行限流处理。
TSF 限流方案采用了动态配额分配制，限流中控根据实例的历史流量记录，动态计算预测下一时刻该实例的流量，若所有实例的流量预测值都小于额定平均值（总配额/在线实例数），则以该平均值作为所有实例分配的配额；否则按预测流量的比例分配，且保证一个最小值。



限流使用场景

- 全局限流：服务设置一个最大的 QPS（每秒请求数）阈值 T，当服务实际接收到的每秒请求数超过 T 时，触发限流。
- 基于标签限流：服务设置【基于标签】的限流规则。

- 示例1：设置自定义标签 `username=foo` 的 QPS 阈值1000次/秒，当服务实际接收到的请求中包含了标签 `username=foo` 的 QPS 超过1000次/秒 时，触发限流。
- 示例2：设置自定义标签 " username 包含 foo, bar " 的 QPS 阈值1000次/秒，当服务实际接收到的请求中包含了标签 `username=foo` 或 `username=bar` 的 QPS 超过1000次/秒 时，触发限流。

使用限流功能

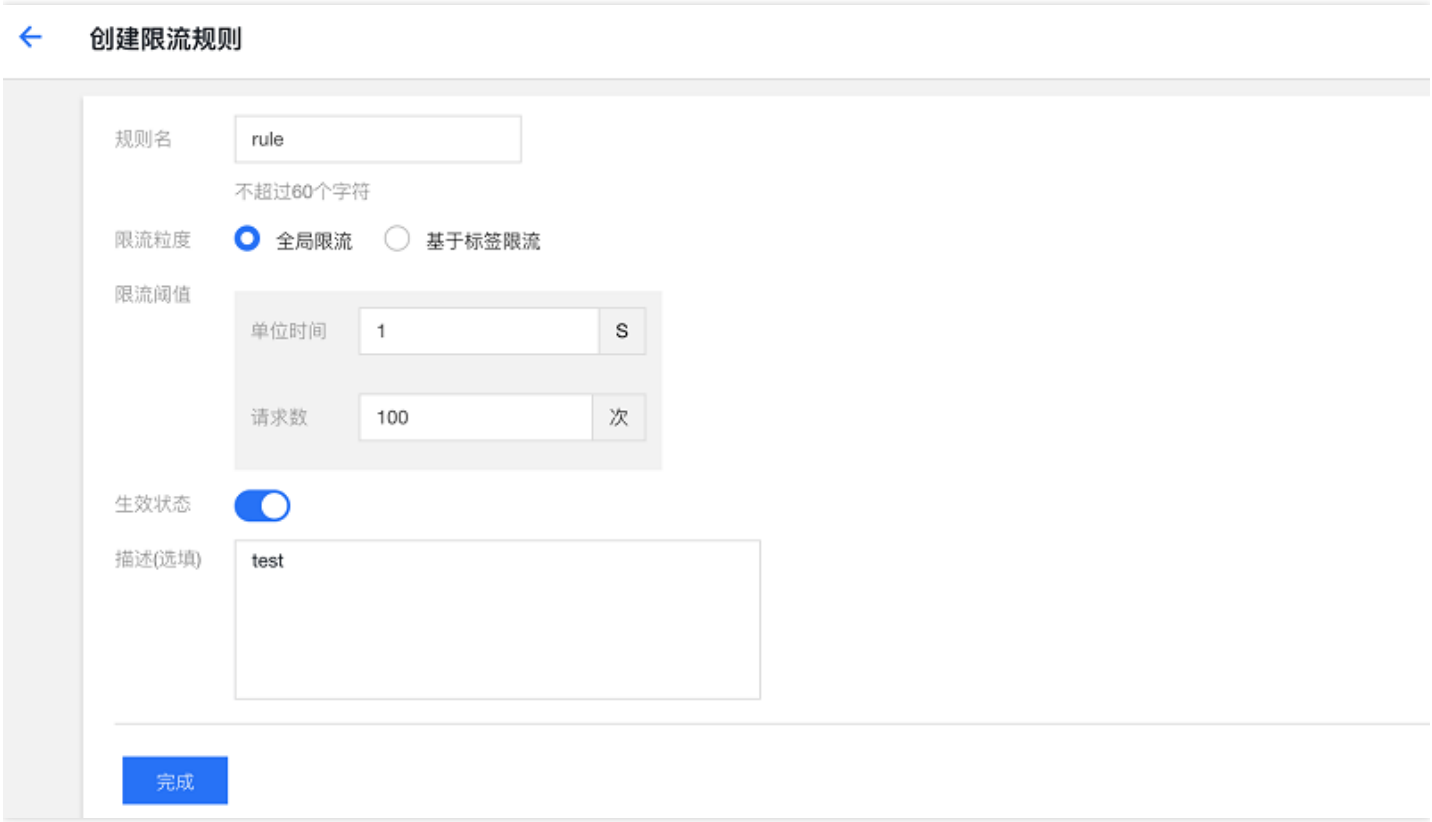
要使用限流功能，用户需要在客户端配置依赖项，然后在 TSF 控制台设置限流规则。

1. 配置依赖项

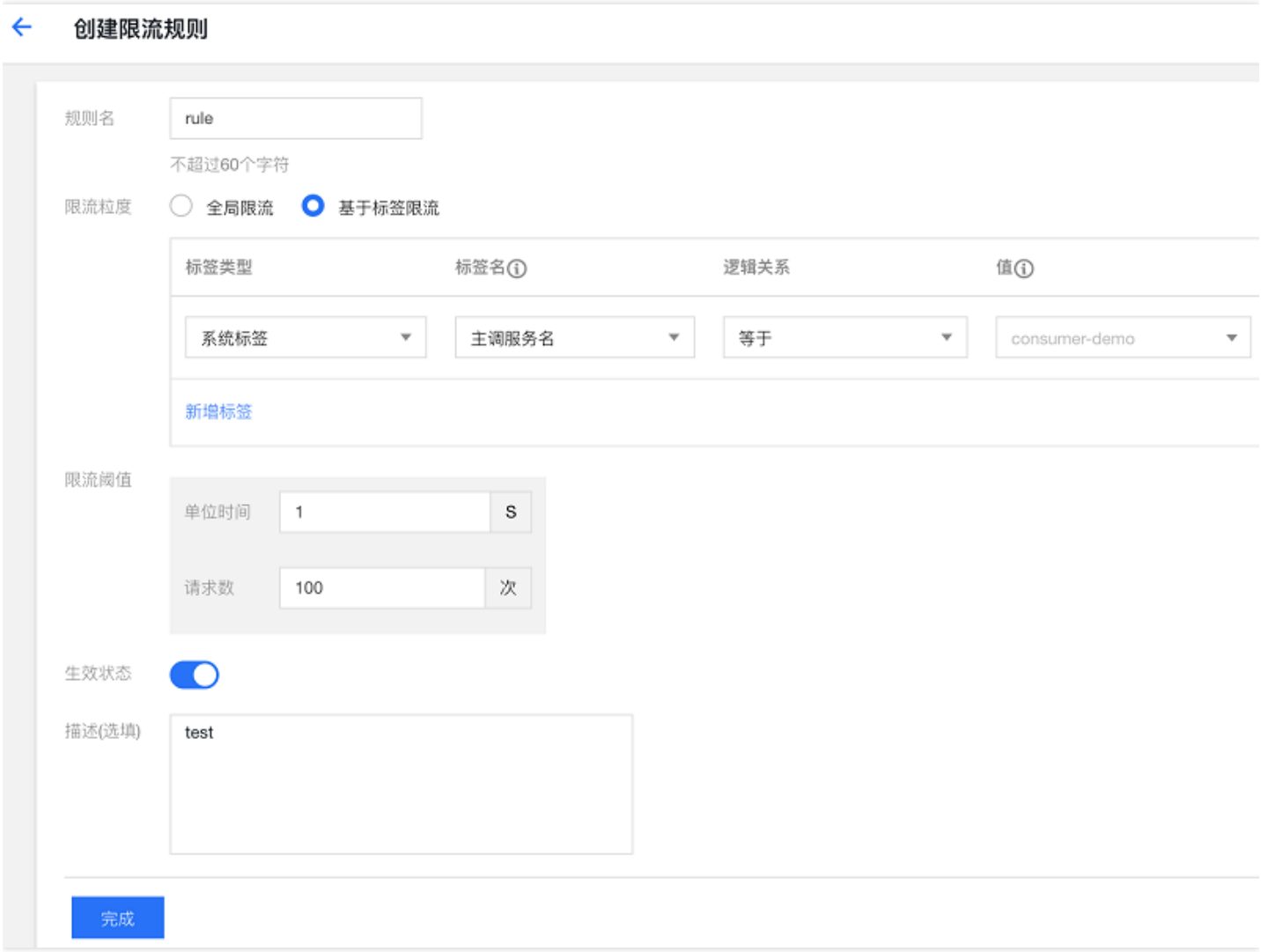
对于 Spring Cloud 应用，参考开发手册中的 服务限流。对于 Mesh 应用，如果希望使用基于标签的限流，需要在代码中设置标签。

2. 新建限流规则

前提条件：服务列表上有 "在线" 状态的微服务。
2.1 登录 TSF 控制台。
2.2 在左侧导航栏，单击【服务治理】。
2.3 在服务列表页，单击服务名，进入服务详情页。
2.4 选择**服务限流**标签页，单击【新建限流规则】按钮。
2.5 填写限流规则信息。
全局限流：


创建限流规则
规则名
rule
不超过60个字符
限流粒度
全局限流
基于标签限流
限流阈值
单位时间
1
S
请求数
100
次
生效状态
描述(选填)
test
完成

基于标签限流：

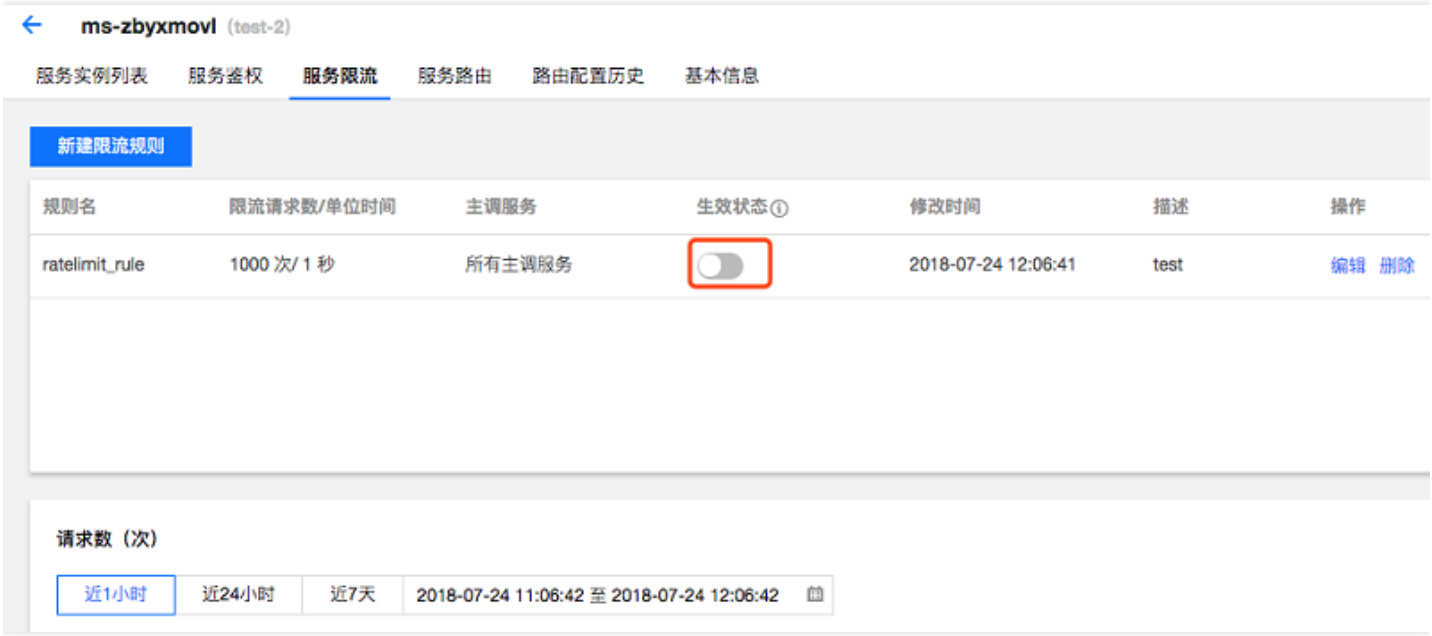


- **规则名**：填写规则名
- **限流粒度**：
  - 全局限流：不区分限流来源，统计所有请求
  - 基于标签限流：根据标签规则设置限流
- **单位时间**：正整数，单位：秒
- **请求数**：正整数，单位：次
- **生效状态**：是否立即启用限流规则
- **描述**：填写描述信息

2.6 单击【提交】按钮。

3. 启动限流规则

在限流规则列表中，可以修改规则的【生效状态】。
多条限流规则都是生效状态时，只要服务接收到的请求满足**任意一条**限流规则，就会触发限流逻辑。



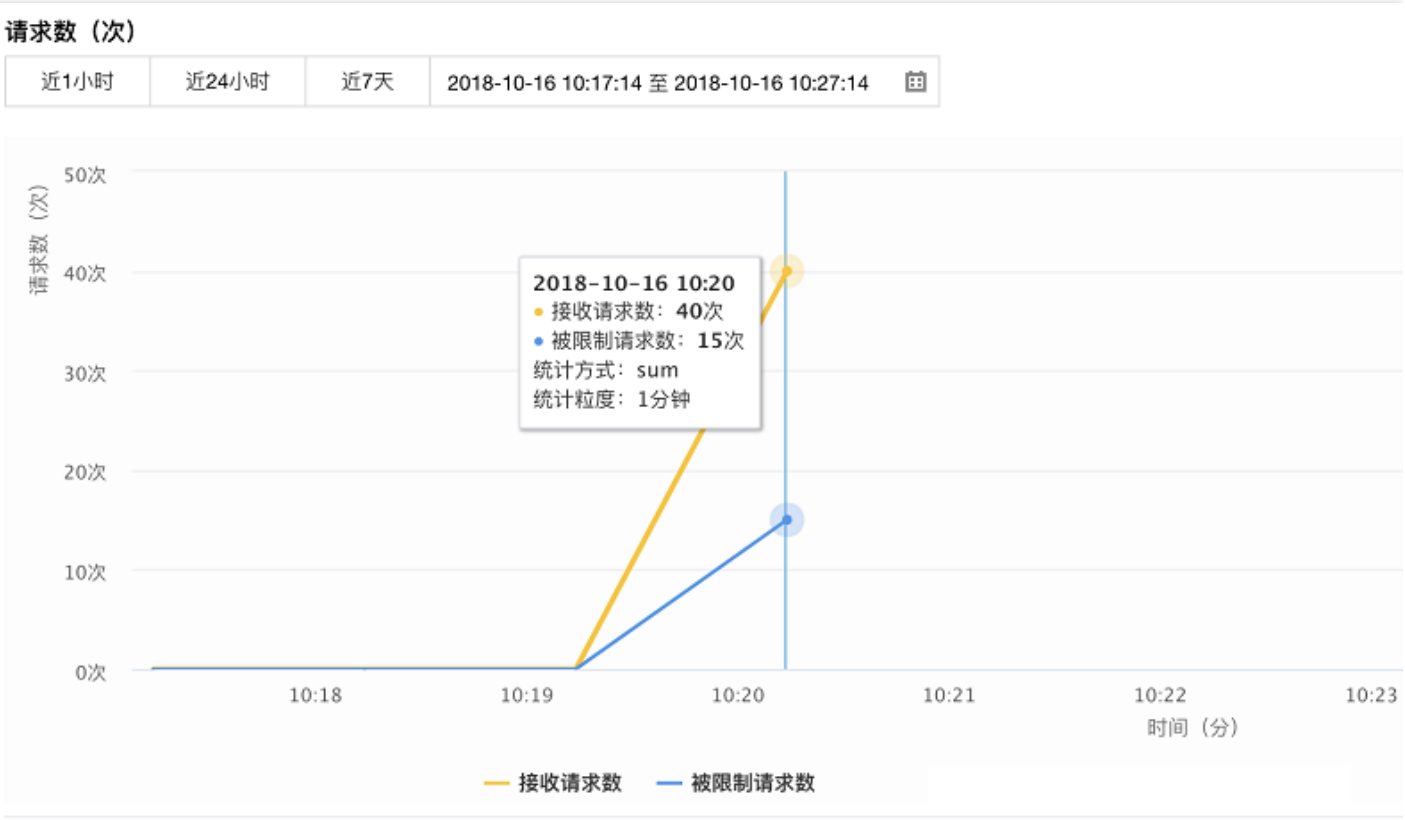
4. 触发限流

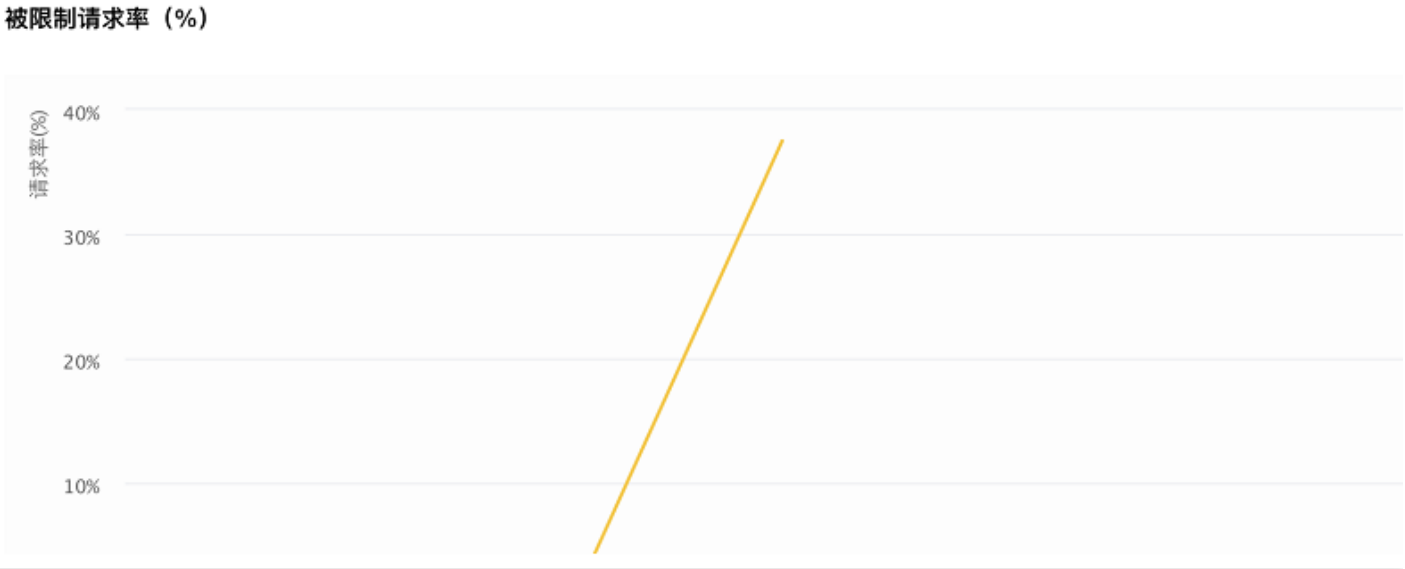
假设服务提供了 `/echo` API，可以通过不断执行 `curl /echo` 来模拟限流场景。
示例：在 Demo 中 consumer-demo 服务提供了 `/echo-feign/{str}` API ，那么针对 consumer-demo 服务新建限流规则，限流粒度为全局限流，单位时间 2 秒，请求数 5 次。启用限流规则。
下载脚本 tsf_ratelimit.sh ，登录可以访问到 consumer-demo 的机器（consumer-demo 所在机器也可以），执行 `./tsf_ratelimit.sh <IP>:<Port>` ，其中 IP 是 consumer-demo 所在机器 IP， Port 为服务监听端口 18083。脚本的作用是**每 2 秒触发 10 次**调用。由于调用的频率大于限流规则，正常情况下，会收到 HTTP 429 (Too Many Requests) 的状态码。

5. 查看限流效果

如果请求数达到了限流阈值，任何到达的请求都会限流模块处理。如果该服务上的配额已经消耗完，会对请求返回 HTTP 429 （Too Many Requests）；否则会正常放行。用户可以在限流规则列表下方的**请求数-时间**图中查看到被限制的请求数或者**被限制请求率-时间**图中查看到被限制请求率（计算公式 被限制请求率 = 被限制的请求数 / 请求

数 ）随时间的变化。


请求数（次）
近1小时
近24小时
近7天
2018-10-16 10:17:14 至 2018-10-16 10:27:14
50次
40次
30次
20次
10次
0次
请求数（次）
2018-10-16 10:20
接收请求数：40次
被限制请求数：15次
统计方式：sum
统计粒度：1分钟
10:18
10:19
10:20
10:21
10:22
10:23
时间（分）
接收请求数
被限制请求数


被限制请求率（%）
40%
30%
20%
10%
请求率(%)

服务路由基本原理

最近更新时间：2019-04-01 11:56:01

服务路由概述

用户在使用 TSF 运行自己的业务时，由于业务的复杂程度，常常需要部署数目庞大的服务运行在现网环境中。这些服务运行在属性不同的实例上、部署在不同的地域中，用户经常需要根据符合自己特定要求的属性选择服务的提供者，对服务间流量的分配起到掌控的作用。
同时，在微服务的场景下，用户研发新版本上线的迭代周期越来越快，稳定敏捷的上线新版本需要微服务框架能够支持灰度发布、金丝雀发布、滚动发布等发布方式。通过服务路由功能，用户可以配置流量分配权重，设置某些权重的流量被分配到某个版本号中，为灰度发布等上线模式提供了无需终止服务的底层能力支持。为了保证满足客户的定制化需求，TSF 支持用户定制自己的路由标签，并支持选择不同的逻辑形式配置标签值，定向分配流量。 总而言之，服务路由功能的主要作用是将调用流量按照自己的需求进行分配。

服务路由原理

要实现服务路由需要完成两部分操作：

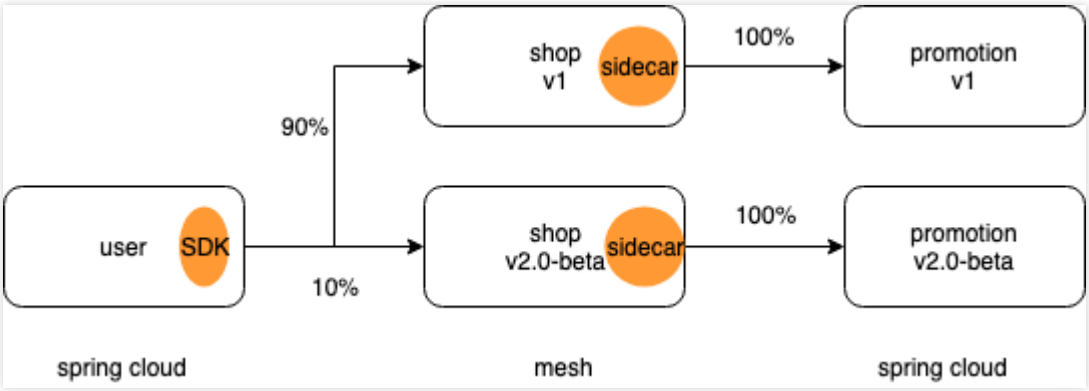
- 在控制台上，给**服务端（服务提供者）**设置路由规则。
- **客户端（服务消费者）**获取路由规则， 根据规则来分发请求。

以 `user -> shop -> promotion` 为例说明服务路由的原理，三个服务特点如下：

- user：Spring Cloud 应用，使用路由 SDK。
- shop：Mesh 应用，有两个版本 v1 和 v2.0-beta。
- promotion：Spring Cloud 应用，有两个版本 v1 和 v2.0-beta。

服务调用和路由情况如下图所示。用户需要在控制台创建如下路由规则：

- shop 服务详情页中配置路由规则：90%的流量分配到 v1 版本，10%的流量分配到 v2 版本。
- promotion 服务详情页中配置路由规则：服务名等于 shop 且版本号为 v1 的流量 100% 分配到 v1 版本，服务名等于 shop 且版本号为 v2 的流量100%分配到 v2 版本。


shop
v1
sidecar
100%
promotion
v1
90%
user
SDK
shop
v2.0-beta
sidecar
100%
promotion
v2.0-beta
10%
spring cloud
mesh
spring cloud

服务路由使用方法

最近更新时间：2019-04-01 11:55:52

前提条件

要使用路由功能，用户需要在客户端配置依赖项，然后在 TSF 控制台设置路由规则。

使用服务路由

配置依赖项

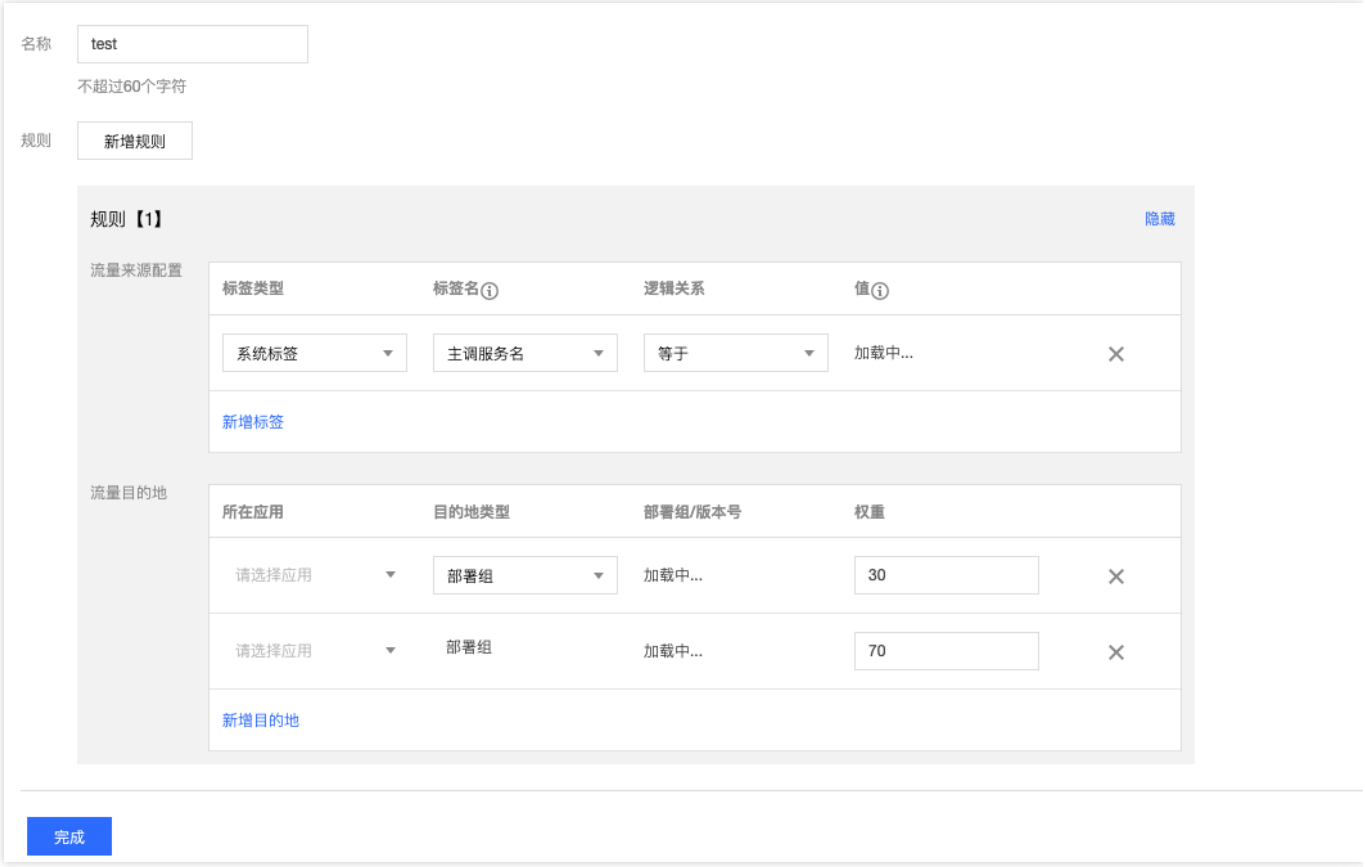
Spring Cloud 应用请参考开发手册中的 服务路由。对于 Mesh 应用，如果希望使用基于自定义标签的路由，需要在代码中设置标签，关于如何设置标签参考 Mesh 开发使用指引 。

新建路由规则

1. 登录 TSF 控制台。
2. 在左侧菜单中，选择【服务治理】。
3. 在服务列表中，选择需要配置服务路由规则的服务，单击服务名称，进入服务详情页。
4. 选择服务路由选项，单击【新建路由规则】。

5. 新建路由规则
   i. 路由规则名称
   ii. 填写规则
   - 流量来源配置：设置系统标签和自定义标签表达式。
   - 流量目的地：支持部署组和版本号两种目的地类型，确保权重加总为100。

iii. 单击【提交】。



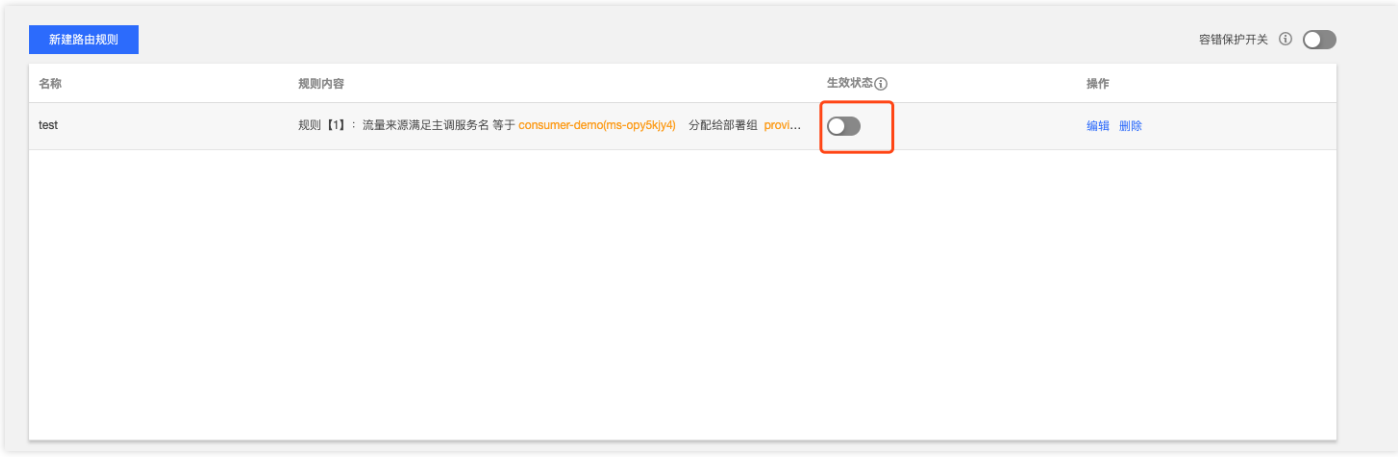
说明：

最多新建10条路由规则。

启用路由规则

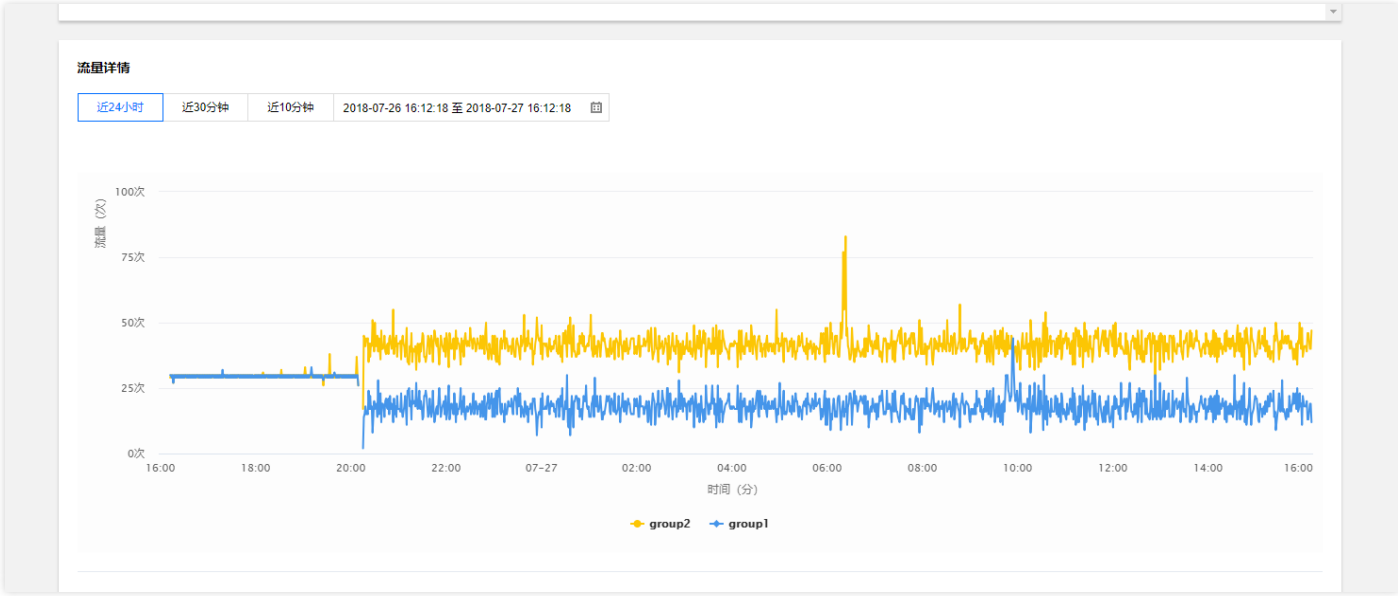
1. 登录 TSF 控制台，单击左侧导航栏的【服务治理】，进入服务详情页面。
2. 选择服务路由标签。

3. 单击【生效状态】的切换按钮，当按钮为蓝色表明已经生效。

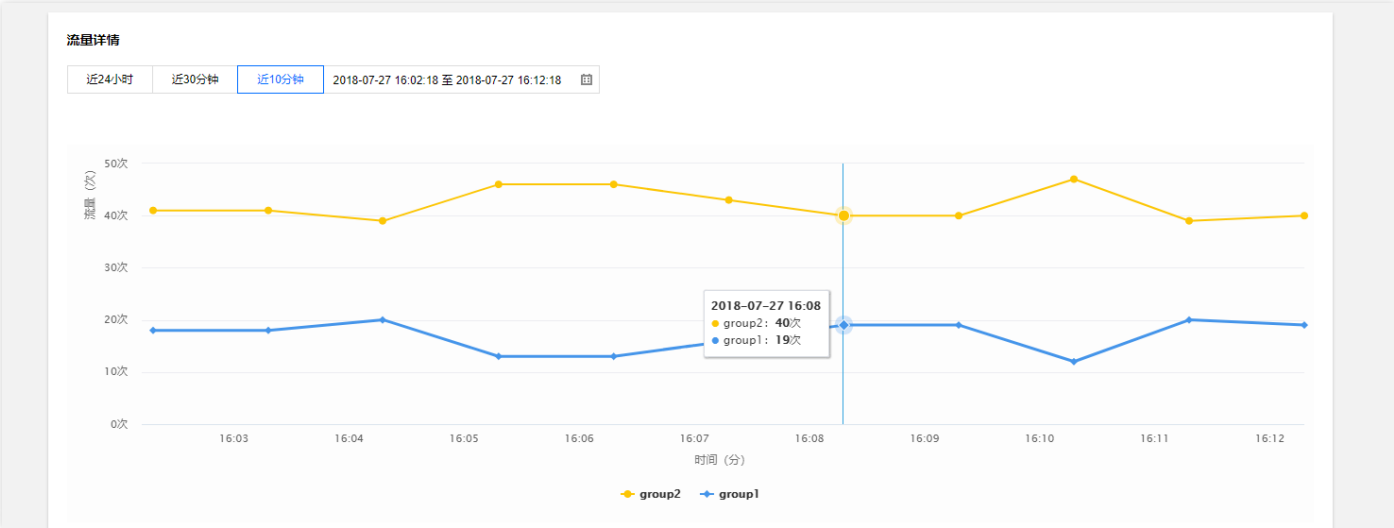


4. 配置生效后，可以在列表项的下面流量分配图中查看流量分配情况，用户可以选择时间段，查看部署组上流量分配情况。

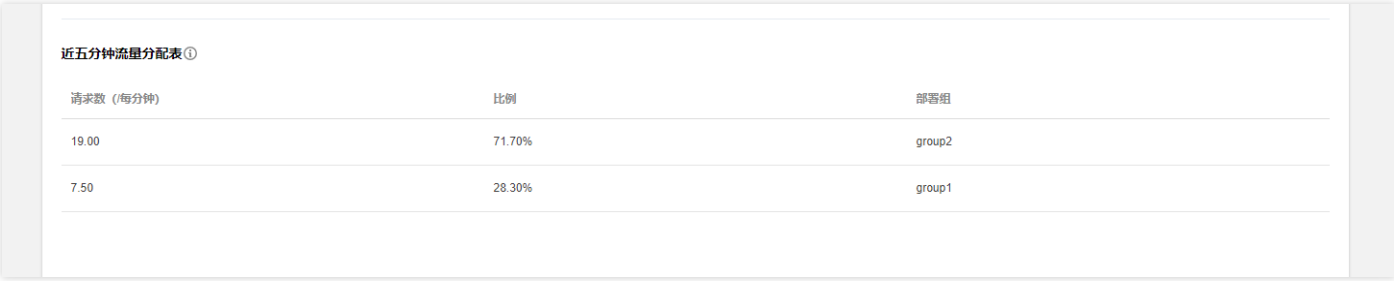
24小时内的流量分配情况如下：



应用路由规则后10分钟之内的流量分配曲线如下：



5. 在流量分配图下方的流量分配表中，可以查看近五分钟内的平均每分钟请求数比例。



近五分钟流量分配表

请求数（/每分钟）	比例	部署组
19.00	71.70%	group2
7.50	28.30%	group1

容错保护

开启容错保护后，会实现兜底策略。例如服务设置了如下路由规则：

- 10%的流量分配到 v1.0 版本。
- 40%的流量分配到 v1.1 版本。
- 50%的流量请求分配到 v1.2 版本。

假设场景： v1.0 版本的实例全部不可用

- 如果不开启容错保护，仍然会有10%的请求分发到 v1.0 版本的实例上，此时请求会失败。
- 如果开启容错保护，SDK 发现 v1.0 版本的实例不可用时，会这 10% 的请求随机分配给 v1.1 和 v1.2 版本实例，最终结果是 v1.1 和 v1.2 收到的请求比例**约等于**45%和55% 。

使用说明

- 填写路由规则需要在服务提供方进行配置，例如 A 服务调用 B 服务，需要在 B 服务上配置服务路由规则。
- 对于 Spring Cloud 服务，配置路由规则后，若配置的目标部署组无法运行，流量将按照原有默认的轮询方式分配到其他部署组上。
- 对于 Spring Cloud 服务，当服务提示未绑定应用时，需要在服务详情页单击编辑，绑定服务，才能开始配置路由规则。**服务绑定应用操作，一经绑定，不能修改。**
- Spring Cloud 服务调用其他服务的场景时，要使服务路由生效，需要确保 Spring Cloud 服务使用了 SDK 并添加开启路由注解，详情请参考开发手册中 服务路由。
- 对于 Mesh 应用，配置路由规则后，若配置的目标部署组无法运行，则路由规则配置失败，请求无法发送。

其他操作

编辑路由规则

1. 登录 TSF 控制台，单击左侧导航栏的【服务治理】，进入服务详情页面。
2. 选择服务路由标签。
3. 在服务路由页面已经提交的规则页面单击【编辑】。

在编辑页面，仅仅支持编辑规则的详情，不支持编辑规则类型。

4. 单击【提交】，完成路由编辑。

说明：

在生效状态的服务路由规则不能被删除，只能先停用，再删除。

服务路由最佳实践

最近更新时间：2018-08-20 15:05:11

灰度发布

- 使用目的：当用户需要上线新的功能时，希望使用灰度发布的手段在小范围内进行新版本发布测试。
- 使用方法：用户可以将新的程序包上传到原有的应用中。用户选择按照权重的方式配置路由规则，填写权重大小，并选择目标版本版本号，便可以实现使用部分流量进行灰度发布的能力。生效中的权重可以被编辑，实时生效，间接实现了滚动发布的功能。

同地机房优先

- 使用目的：当企业规模较大时，单个机房的容量已经不能满足业务需求，业务常常出现跨机房部署的情况。然而由于异地跨机房调用出现的网络延迟问题，需要能够保证服务消费方能优先调用本地的服务消费方，这就需要采用服务路由的方式。
- 使用方法：用户选择系统自带标签路由选项，配置系统自带标签为发起方ip，在正则表达式中填写服务消费方的ip字段规则。对于服务提供方，用户可以将ip地址相近的实例归属在同一个部署组上，作为目标部署组，实现优先调用同地机房。

部分帐号内测

- 使用目的：希望配置某些使用者使用的版本为新的内测版本。
- 使用方法：用户可以配置自定义标签为用户id，设置id值的正则表达式计算方式，保证服务消费方发起的请求带有以上条件的流量分配到服务提供方的某个版本号上，实现帐号内测功能。

其他实践

- 在实际的使用中，用户也可以通过服务路由功能，实现优先保护重要服务的运行质量、前后端分离、读写分离等等功能。