

# 注册配置治理 Polaris(北极星)





#### 【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】

## 🔗 腾讯云

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

Polaris (北极星) 北极星概述 快速入门 引擎管理 引擎管理 标签管理 查看日志 服务管理(注册中心) 命名空间管理 服务管理 服务实例管理 服务实例隔离 服务实例元数据管理 服务别名 关联 K8s 集群 跨地域使用指南 服务治理(治理中心) 服务路由 就近路由 动态路由 熔断降级 服务熔断与主动探测 实例熔断 服务限流 无损上下线 无损上线 无损下线 全链路灰度 联动云原生网关实现全链路灰度 配置管理(配置中心) 配置分组 配置文件 配置轨迹 配置 Agent 安装 可观测性 引擎运行监控 注册配置监控 服务调用监控 监控排障 配置告警 事件中心 操作记录 权限控制(权限中心) Java 应用开发 Spring Cloud 应用开发 Spring Cloud Tencent 概述 Spring Cloud 版本支持 Spring Cloud 接入 注册发现



配置管理 服务限流 服务熔断 动态路由 全链路灰度 无损上下线 使用 Java Agent Java Agent 概述 使用 Java Agent 接入 Java Agent 支持无损上下线 Java Agent 支持配置管理 使用 Java SDK Polaris - Java 接入 注册发现 负载均衡 动态路由 就近访问 服务限流 分布式限流 熔断降级 配置管理 可观测性 二次寻址 Go 应用开发 Polaris-Go 接入 gRPC-Go 接入 二次寻址使用指南 就近访问使用指南 迁移指南 TSF Consul 迁移方案 Nacos 迁移方案 协议兼容接入 Eureka 迁移方案 Eureka协议使用命名空间 从 Eureka 迁移到 Polaris

## Polaris(北极星) 北极星概述

> 腾讯云

最近更新时间: 2025-04-17 16:00:52

**腾讯微服务平台(TSF) – 注册配置治理 – Polaris(北极星)**用于解决分布式或者微服务架构中的服务注册与发现、故障容错、流量控制和安全问题,提供 快速部署、高可用容灾、免运维、一键搭建北极星网格能力。

Polaris(中文名北极星,以下简称**北极星**)是腾讯开源的服务发现和治理中心。北极星在腾讯内部的服务注册数量超过百万,日接口调用量超过十万亿,通 用性和稳定性都得到了大规模的验证。

#### 核心特性

- 拥抱开源: 支持开源主流客户端接入,例如: Spring Cloud、Mesh、gRPC 等。同时,北极星已正式对外开源。
- 服务注册与发现: 高可用、大容量的服务注册中心,整合企业存量服务数据。
- 路由和负载均衡: 动态调度现网流量,实现灰度发布等各种流量调度策略。
- 故障熔断和限流: 自动进行容灾切换和服务降级,保证服务调用的可靠性。
- 可观测性: 数据面上报服务调用统计和运行记录,实现运行状态可观测性。

### 接入方式

北极星网格为解决异构系统、多语言场景下的服务治理痛点,提供多种接入方式,可根据系统现状自行选择。如下图所示:



接入方式	支持的选型
SDK	Java / Go 。
框架	提供 Spring Cloud / gRPC 官方集成,其他框架也可以自行集成。
服务网格	通过 polaris−sidecar 接入,兼容 xDS 协议和 envoy。
K8s服务治理	支持 K8s Service 自动注册到 Polarismesh,扩展其治理能力。



## 快速入门

最近更新时间: 2025-06-04 11:46:02

#### 操作场景

Polaris(北极星)用于解决分布式或者微服务架构中的服务注册与发现、故障容错、流量控制和安全问题,支持多语言客户端、集成多种主流服务框架,帮助用户实现高效、稳定、高可用、免运维的服务治理能力。

本文介绍在 TSF 控制台创建一个 Polaris(北极星)实例的操作步骤和应用接入的流程,帮助您快速了解如何使用北极星。

#### 前提条件

已获取访问授权

#### 创建引擎实例

- 1. 登录 TSF 控制台。
- 2. 左侧导航栏单击注册配置治理中心下的Polaris (北极星)进入北极星实例详情页面,单击新建。
- 3. 在购买页根据自身业务需求选择相关配置。

#### () 说明:

Polaris (北极星)默认支持同城多活高可用架构。

参数	说明
计费模式	支持包年包月和按量付费两种计费方式,如果您的北极星网格使用时间在一个月以上,建议采用预付费(包年包月)模式, 具体价格请参见 Polarismesh 价格说明 。
开源版本	支持 1.9.0.1
产品版本	<ul> <li>企业版:企业版适用于生产环境或测试环境。支持多可用区部署。具有服务治理中心全部能力:注册中心+配置中心+治理中心。</li> <li>基础版:基础版提供注册中心和配置中心的能力,适用于生产环境或测试环境,如需治理能力,推荐购买企业版。</li> <li>开发版:仅支持单节点,1核1G的规格,建议仅用于体验。开发版没有容灾能力。为了保证您业务的稳定性,正式环境强烈建议标准版。</li> <li>标准版:可用于测试、生成环境。支持多节点部署。(当前已售罄)</li> </ul>
规格	根据业务需求选择服务实例配额数量。
地域	选择与您部署业务最靠近的地域。
集群网络	所选择的私有网络必须和业务所在的私有网络一致。所选择的子网不用和业务所在的私有网络一致。
名称	最长60个字符,支持中英文大小写、-、,名称一旦创建后不支持修改。
资源标签	用于分类管理资源,选填,具体使用方法可参见 标签管理 。

4. 单击**立即购买**,在 Polaris 列表页面的状态栏,您可以查看到引擎创建的进度。

#### 后续步骤

引擎实例创建完成后,您可以根据您的业务场景选择对应的参考文档体验应用接入的操作流程。

应用场景	参考文档
	Spring Cloud 接入
Java 应用开发	Java Agent 接入
	polaris-Java 接入



Co 应用开发	polaris−Go 接入
GO MHTA	gRPC-Go 接入
迁移指南	Eureka 迁移指南



## 引擎管理

引擎管理

最近更新时间: 2025-04-17 16:00:52

### 操作场景

本文介绍通过腾讯微服务平台(TSF)控制台创建、查看、销毁引擎的操作步骤,快速了解 TSF 控制台操作流程。

### 操作步骤

### 创建引擎

- 1. 登录 TSF 控制台,在左侧导航栏选择 Polaris(北极星)。
- 2. 在引擎实例列表页,单击**新建**。
- 3. 在新建北极星网格页,根据自身业务需求选择相关配置。

参数	说明
计费模式	支持包年包月和按量付费两种计费方式,具体价格请参见
开源版本	1.17.0.0
产品版本	<ul> <li>企业版:企业版适用于生产环境或测试环境。支持多可用区部署。具有服务治理中心全部能力:注册中心 + 配置中心 + 治理中心。</li> <li>基础版:基础版提供注册中心和配置中心的能力,适用于生产环境或测试环境,如需治理能力,推荐购买企业版。</li> <li>开发版:开发版仅支持单节点,1核1G的规格,建议仅用于体验。开发版没有容灾能力。为了保证您业务的稳定性,生产环境强烈建议企业版或基础版。</li> <li>标准版(已售罄):适用于生产环境,具有注册中心、配置中心和基础服务治理能力。</li> </ul>
产品规格	根据业务需求选择接入节点配额数量。节点数为线上接入北极星的客户端数,即为访问北极星服务端的应用进程个数。接入北极 星客户端类型包括 polaris sdk、polaris agent、sidecar、polaris controller 等。
部署架构	支持高可用架构,按您业务实际需要选择同城多可用区和跨城多可用区。
地域配置	<ul> <li>同城多可用区:根据您业务实际的架构规划,选择与您部署业务最靠近的地域。</li> <li>注册配置治理-北极星将引擎实例自动部署在多可用区。</li> <li>实际部署架构可在引擎创建完成后,单击引擎 ID 进入实例信息页,最下方部署架构版块中,可以看到具体部署架构。</li> <li>跨城多可用区:根据您业务实际的架构规划,选择多个地域进行引擎实例组网。</li> <li>默认首个地域为集群主地域,主地域不可删除。</li> <li>主地域与其他地域在使用上没有区别,均支持数据读写。</li> <li>多个地域支持分别配置接入节点规格。</li> <li>多个地域支持分别配置接入集群网络。</li> <li>最多支持 5 个地域。</li> </ul>
集群网络	根据业务网络规划怎么集群网络,所选择的私有网络必须和业务所在的私有网络一致。所选择的子网没有一致性的限制。
监控与日志	<ul> <li>选择是否开启事件中心、操作记录、告警能力。</li> <li>♪ 注意:</li> <li>● PolarisMesh 事件中心、操作记录、告警数据存储在 CLS 日志服务。开启后自动创建日志集 tse_logset,及 主题: polarismesh_event、polarismesh_operation、polarismesh_alarm,您在polarismesh控制 台上可以查询到这些事件、操作记录和告警配置。</li> <li>● 请注意 CLS 日志服务为独立计费产品,例如: 100万条事件和操作记录产生费用约10.29元/月,购买页中不包括 这部分费用。具体计费信息查看 CLS 费用详情。</li> </ul>



名称	最长60个字符,支持中英文大小写、-、_,名称一旦创建后不支持修改。
资源标签	选填,具体使用方法可参见 标签管理 。

4. 单击**立即购买**,在 Polaris 列表页面的状态栏,您可以查看到引擎创建的进度。

#### 访问方式

单击实例 "ID",在引擎详情页的实例信息页面,可以查看到引擎的访问地址。

#### 控制台访问地址

默认开启公网访问,您可以单击公网地址跳转链接访问 Polarismesh 控制台。公网访问建议设置访问安全策略,您可以通过设置白名单的方式控制公网访 问。

介 注意 为了您 ● 初始 ● 初始	的业务安全,请尽快登录 Polarismesh 控制台修改初始密码。Polarismesh 控制台的初始登录账号和密码如下: 治用户名:polaris 治密码:polaris@实例ID
控制台访问地址	
内网地址	开启
公网地址 🛈	
公网访问策略 🛈	The second for the local lines were
用户名	1032a40115
初始密码	······· E @

#### 客户端访问地址

客户端访问地址是 Polarismesh 服务端暴露的地址,供用户服务连接到 Polarismesh 进行服务通信、服务配置时使用。客户端访问方式默认提供 VPC 内 网访问。公网访问方式可用于开发调试或辅助管理,生产使用建议内网访问方式,避免外网访问的潜在安全风险。若因特殊需要开启公网访问,建议配置不包 含0.0.0.0的白名单。

ana server (/)	问地址				
客户端接入端口	注册发现 8091 (service	e-grpc) 8092 (service-trpc)			
	配置管理 8093 (config	-grpc)			
	监控上报 9091 (pushga	ateway)			
DpenAPI 端口	8090 (http)				
か议兼容端口	7779 (I5) 8761 (eureka	a) 15010 (xdsv3) 15020 (xdsv2)			
负载均衡 🛈	私有网络	子网	内网地址	访问控制	操作
负载均衡 🐧	私有网络 Januaria (Januaria) (K	子同 	内网地址 КСХЭН	访问控制	操作
负载均衡 <b>③</b>	私有网络 	78 mirus (* jaanse Cab II) (*	内网地址 <b>KCK3H</b>	访问控制	操作
负载均衡 ① 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	私有网络 ・**** ********************************	子段 1000-100 (201-1-1) 公用带宽	内周地址 <b>KLIKIM</b> 访问策略	访问控制	操作 删除 操作

为了您的业务安全,请尽快登录北极星控制台修改初始密码,此处仅展示初始密码

例如 Java 应用,使用 polaris-java sdk 的方式,可在应用的 classpath 当前目录下,添加 polaris.yml 文件,配置服务端地址信息。

global:			
serverConnector:			
addresses:			



#### 10.10.0.73:8090

#### 监控数据上报地址

在客户端配置监控上报地址 上传北极星监控数据。

#### 销毁引擎

注意:
 北极星网格引擎销毁后,服务不可用且数据销毁,该操作不可逆,请谨慎操作。

#### 1. 登录 TSF 控制台。

- 2. 在Polaris (北极星)下的 Polaris 页面,找到目标引擎,单击操作列的删除。
- 3. 在二次确认的弹框中单击删除,完成北极星引擎销毁。



## 标签管理

最近更新时间: 2025-04-17 16:00:52

#### 操作场景

该任务指导您为微服务引擎实例绑定标签及修改标签。

#### 操作前提

- 拥有腾讯云主账号,且已经开通腾讯云访问管理服务。
- 主账号下至少有一个子账号,且已根据 子账号获取访问授权 完成授权。
- 至少拥有一个微服务引擎实例。
- 至少拥有一个标签,若您没有,可以前往标签控制台 > 标签列表进行新建。

#### 操作步骤

#### 资源绑定标签

1. 使用**主账号**登录到 TSF 控制台,选择已有的引擎实例并单击进入详情页。



2. 单击进入某个引擎实例的实例信息页,单击标签的编辑图标。

实例信息	K8s 集群	运行监控	运行日志	参数配置					
基本信息									
ID				产品版本	企业版	引擎版本	1.17.0.1 引擎版本升级	节点数量	北京 (50) 重庆 (50)
名称				地域	北京(主) 重庆	状态	运行中	可用区	()重庆一区,北京三区,北京七区 ₽
创建时间	2023-10-23 19:10	0:03		标签					

3. 单击添加,选择需要绑定的标签键与标签值,完成为TSE实例绑定标签。

示签①	标签键		标签值	▼ ×	
	+ 添加 💿 键值	粘贴板			
	标签用于从不同维度 创建标签	度对资源分类管理	里。如现有标签不符	合您的要求,请前往 <mark>标</mark>	签管理 🖸

#### 编辑标签

- 1. 登录 TSF 控制台,选择已有的引擎实例并单击进入详情页。
- 2. 单击进入某个引擎实例的基本信息页,单击标签的编辑图标。

实例信息	K8s 集群	运行监控	运行日志	参数配置					
基本信息									
ID				产品版本	企业版	引擎版本	1.17.0.1 引擎版本升级	节点数量	北京 (50)   重庆 (50)
名称				地域	北京(主) 重庆	状态	运行中	可用区	①重庆一区,北京三区,北京七区 ♪
创建时间	2023-10-23 19:10	03		标签					

3. 您可以对实例绑定的标签进行添加、删除、修改等操作。



标签①	-	• III	▼ ×
	标签键	▼ 标签值	- ×
	+ 添加 💿 键值料	贴板	
	标签用于从不同维度5 创建标签	讨资源分类管理。如现有标签不符	守合您的要求,请前往 <b>标签管理 </b>

## 附录

更多标签使用方法请参见标签。

若您需要微服务引擎实例的按标签鉴权能力,请参见 按标签授权。



## 查看日志

最近更新时间: 2025-06-11 15:55:02

#### 操作场景

本文为您介绍服务治理中心运行日志的使用说明。用户可以通过节点的运行日志,了解节点的运行状况,定位问题,辅助集群的应用开发和运维。

#### 操作步骤

1. 登录 TSF 控制台,左侧导航栏选择 Polaris (北极星)进入引擎实例列表页。

- 2. 在引擎实例列表页面,单击目标实例的"ID",进入基本信息页面。
- 3. 在顶部页签单击**运行日志**。
- 4. **实时日志** 
  - 4.1 选择好**节点**后,可查看对应节点的运行日志。

4.2 在日志页面的搜索框,可以通过关键字查询相关日志。输入关键词查询,例如:"info",注意日志检索区分大小写。



#### 5. CLS日志

PolarisMesh 持久化日志、事件中心、操作记录、告警数据存储在 CLS 服务中,开启 CLS 日志服务后可查看。

实时日志 CLS日志										
🕐 PREISEERE, plannen belez ha, han, Rithfer, Brizhowen, Brizhowen										
						MHCLS88.99				
日志黄型	日志集	日本主题	保存时长	状态	操作					
监控告管	c0be4785-b281-413b-9411-7d33d0f6040b tse_logset	e61e4561-fbaa-45b3-b14a-a77e10aa06be ins-3d934fa3_polarismesh_alarm	30	EX	编辑 作双					
事件	c0be4765-b261-413b-9411-7d33d0f6040b tse_logset	e78ec306-0cdb-4713-b9d2-450464f948fe ins-3d934fa3_polarismesh_event	30	ER	编辑 修理					
操作记录	c0be4765-b261-413b-9411-7d33d0f6040b tse_logset	47fbef12-f008-4f6e-a488-20f8405dbe8a ins-3d934fa3_polarismesh_operation	30	ER	编辑 修复					
共 3 张					10 ~ 条/页	H H				

## 🔗 腾讯云

## 服务管理(注册中心) 命名空间管理

最近更新时间:2025-04-17 16:00:53

### 操作场景

命名空间(Namespace)是对一组资源和对象的抽象集合,用于服务和实例的逻辑隔离。同一命名空间下的资源命名必须唯一,但是跨命名空间允许存在同 名的资源。命名空间常用于多个团队或者项目之间的资源区分隔离,如区分环境:开发、测试、预发布、生产环境等;区分业务。 Polaris 默认存在2个命名空间:

- Polaris: Polaris 命名空间存放的是北极星的系统服务, Polaris 自身的集群发现及管理需要依赖 Polaris 命名空间下的服务。
- default:默认命名空间,用户如果没有多命名空间的需求,可以直接使用 default 命名空间。

#### 创建命名空间

- 1. 登录 TSF 控制台, 左侧导航栏选择Polaris (北极星), 进入引擎实例列表页。
- 2. 在引擎实例列表页,单击实例列表,进入实例信息页面。
- 3. 单击目标实例引擎操作栏的控制台,输入用户名和密码,进入 POLARIS MESH 控制台页面。

新建							默认开公网,如需关闭	,请前往访问控制页中关闭。	
ID/名称	地域 ▼	版本	产品版本	规格	状态	计费模式	默认账号: polaris 默认密码: polaris@ins-( 5		
ins-	广州	1.0.0	标准版	实例配额数:50	运行中	按量付费	2022-03-01 16:17:06	控制台 <b>劉除</b> 更多 マ	

4. 在左侧导航栏选择命名空间,单击新建,填写命名空间名称和描述信息。

名空间 *		字、英文字母、、_, 限制					
述	长度不适	超过1024个字符					
高级设置 权	用户	用户组					
	Û	授予权限的用户或者用户组,	具有该命名空间的读写	权限,所	有用户	具有读权限	
	请选择用	Þ				已选择 (0)	
	输入关	建字搜索		Q,			
	10	0020293116 (100020293116	)	Î			
	10	0025230909 (100025230909	)				
	10	69284099 (1069284099)		- 1			
	10	0026801751 (100026801751	)				
	10	0013113842 (100013113842	)				
	10	0008014242 (100008014242	)				
		~~		Ŧ			
	支持按住。	shift进行客选					

5. 单击提交,完成命名空间创建。

#### 编辑命名空间

- 1. 在命名空间页面,找到目标命名空间,单击操作列的编辑。
- 2. 在编辑命名空间页,修改编辑命名空间的描述信息。
- 3. 单击提交,完成命名空间编辑。



() 说明:

仅支持修改命名空间描述信息,命名空间名称创建后不可修**改**。

#### 删除命名空间

- 1. 在**命名空间**页面,找到目标命名空间,单击操作列的**删除**。
- 2. 在删除的确认弹框中,单击**删除**即可删除命名空间。

#### ▲ 注意:

- 命名空间删除后无法恢复,请确保不需要后再删除。
- 如果命名空间下存在服务,则不可删除。需要先将命名空间下的服务删除后,命名空间才可被删除。



## 服务管理

最近更新时间: 2025-04-09 11:24:22

#### 操作场景

在 Polaris(北极星 )中,每个服务具有唯一的服务名。业务通过服务名来做服务发现。对于一次 RPC 调用来说,是主调服务将请求发送给被调服务。服务 实例是服务下注册的具体的业务节点,通常业务会在服务下注册多个节点,有时也被称为节点或 RS。

Polaris(北极星) 提供可视化的服务管理界面,服务注册至注册中心后,您可以在腾讯微服务平台控制台查看服务上下线和运行情况,并对服务进行编辑修 改等。

该任务指导您通过腾讯微服务平台控制台,如何在某一北极星引擎下进行服务管理。

#### 前提条件

- 已 创建北极星网格引擎。
- 已 创建命名空间。

#### 操作步骤

#### 新建服务

#### 场景一

如果您通过如下四种方式接入 Polaris(北极星),我们将提供服务自动注册能力,您无需在控制台手动新建服务,在 TSF 控制台上可直接查看服务详情。

接入方式	支持类型
SDK	支持 Java、Go。
框架	提供 Spring Cloud 和 gRPC 官方集成,其他框架也可以自行集成。
服务网格	通过 polaris−sidecar 接入,兼容 xDS 协议和 envoy。
K8s 服务治理	支持 K8s Service 自动注册到 Polarismesh,扩展其治理能力。

<b>服务列表</b> 服务别名								
新建 删除						请选择务	条件进行过滤	Q¢¢
服务名	命名空间 🍸	部门	业务	健康实例/总实例	创建时间	修改时间	操作	
QuickstartCallerService	default			2/2	2023-11-02 19:48:44	2023-12-12 12:19:57	编辑 删除	
QuickstartCalleeService	default			2/3	2023-11-02 19:53:47	2023-12-12 12:20:09	编辑 删除	
服务标签 -								
描述 -								

如果您需要**服务标签**等能力,则通过**编辑服务**的能力即可实现。

#### 场景二

如果您使用其他方式接入 Polari(北极星),则需要在 TSF 控制台手动新建服务与服务实例,步骤如下:

- 1. 登录 腾讯微服务平台控制台,在左侧导航栏下选择 Polaris(北极星),进入北极星实例列表页面。
- 2. 单击目标实例引擎操作栏的控制台,输入用户名和密码,进入 POLARIS MESH 控制台页面。

新建							默认开公网,如需关闭,请前往访问控制页中关闭 默认账号: polaris			
ID/名称	地域 ▼	版本	产品版本	规格	状态	计费模式	默认账号:polaris 默认密码:polaris@ins-JI	5		
ins-	广州	1.0.0	标准版	实例配额数:50	运行中	按量付费	2022-03-01 16:17:06	控制台 删除 更多 ▼	٢	

- 3. 在左侧导航栏选择**服务管理**,单击新建。
- 4. 在新建服务页,根据自身业务需求选择相关信息。



新建服务					×
命名空间*	请选择		Ŧ		
服务名 *	允许数字、英文字母、、	_, 限制128个字符			
部门					
业务					
开启就近访问					
服务标签①	1二饮油			¦□.//□	
	10/22/95	标金恒		10K1F	
	10/22.98	标金值	R.	2#11-	
	101.021.022 添加标签	标业值	864	28(1)	
描述	1940年1942 添加标签 长度不超过1024个字符	标金值	2	2011	
描述 ▼ <b>高级设置</b>	10432182 添加标签 长度不超过1024个字符	标金谱	5	2011	

- 命名空间:选择创建好的命名空间。
- 服务名:填写服务名称,允许数字、英文字母、.、-、\_,限制128个字符。
- 部门:可选,根据您的实际情况填写。
- 业务:可选,根据您的实际情况填写。
- 服务标签:可选,服务标签可用于标识服务的用处、特征,格式为 key:value。
- 描述: 可选,填写服务描述信息。
- 5. 单击**提交**,完成新建服务。
- 6. 单击下拉按钮,可以查看服务相关属性。

						请选择条件进行过滤	Q¢φ
命名空间 🍸	部门	业务	健康实例/总实例	创建时间	修改时间	操作	
default	研发部	社交业务	2/2	2023-11-02 19:48:44	2023-12-12 12:1	9:57 编辑 删除	
default	研发部	社交业务	2/3	2023-11-02 19:53:47	2023-12-12 12:2	20:09 编辑 删除	
	命名空间 T default default	유名空间 T 제기	유조조제 지 제가 2 제	휴名空间 T     部门     业务     健康奕纲/检奕纲       default     研发部     社交业务     2/2       default     研发部     社交业务     2/3	유축空间 T         部门         业务         健康奕衍总英列         健雄時           default         元         社交业务         2/2         203-11-02 19:48:44           default         元         社交业务         2/3         203-11-02 19:48:44	유조亞히 자         阿기         보호         建築門/(支索)         建築印         1 <th1< th="">         1         <th1< th=""> <th1< th=""></th1<></th1<></th1<>	유조의 지         제가         제가         제가         제가         제가           유조의 지         제가         제가         제가         제가         제가         제가           네스티스스         제가         전화         1

后续操作: 服务创建完成后,您需要继续该服务下创建服务实例,参见 服务实例管理。

#### 编辑服务

- 1. 登录 腾讯微服务平台控制台,进入北极星实例列表页。
- 2. 选择目标引擎实例后,在**北极星**下的**服务列表**页面,选择目标服务,单击操作列的编辑。

_	<b>服务列表</b> 服务别名							
	新建删除			请选择条件进行过滤	Q¢¢			
	服务名	命名空间 ▼	部门	业务	健康实例/总实例	创建时间	修改时间	操作
	▶ □ p	Polaris	-	polaris	2/2	2021-09-06 15:55:07	2021-09-06 15:55:09	编辑删除

3. 在编辑服务页,根据自身业务需求编辑相关信息。



编辑服务				×
命名空间 *	Polaris		~	
服务名 *	polaris.checker			
部门				
业务	polaris			
开启就近访问				
服务标签()	标签键	标签值	操作	
		无标签		
	添加标签			
描述	polaris checker service			
▼ 高级设置				
		提交关闭	]	

4. 单击**提交**,完成编辑服务。

#### 删除服务

- 1. 登录 腾讯微服务平台控制台,进入北极星实例列表页。
- 2. 选择目标引擎实例后,在**北极星**下的**服务列表**页面,选择目标服务,单击操作列的**删除**。

服务列表	服务别名							
新建	删除					and a state	§选择条件进行过滤	Q \$\$ \$
B	服务名	命名空间 🔻	部门	业务	健康实例/总实例	创建时间	修改时间	操作
► 🗆 t	00	Polaris	-	polaris	2/2	2021-09-06 15:55:07	2021-09-06 15:55:09	编辑 删除

3. 在二次弹窗页面确认删除。





## 服务实例管理

最近更新时间: 2025-04-17 16:00:53

### 操作场景

该任务指导您如何通过 TSF 控制台在某一北极星实例下进行服务实例管理。

#### 操作前提

- 已 创建北极星引擎实例。
- 已 创建命名空间。
- •已创建服务。

#### 操作步骤

#### 新建与查看服务实例

#### 场景一

如果您通过如下四种方式接入 Polaris(北极星),我们将提供服务自动注册能力。您无需新建服务与服务实例,即查看服务实例详情。

接入方式	支持类型
SDK	支持 Java、Go。
框架	提供 Spring Cloud 和 gRPC 官方集成,其他框架也可以自行集成。
服务网格	通过 polaris−sidecar 接入,兼容 xDS 协议和 envoy。
K8s 服务治理	支持 K8s Service 自动注册到 Polarismesh,扩展其治理能力。

#### 步骤如下:

1. 登录 TSF 控制台,进入北极星实例列表页。

2. 选择目标引擎实例后,在**北极星**下的**服务管理**页面,选择目标服务,单击目标服务操作列的控制台按钮。

新建							默认开公网,如需关闭,	请前往访问控制页中	- 中关闭。
ID/名称	地域 🔻	版本	产品版本	规格	状态	计费模式	默认账号:polaris 默认密码:polaris@ins-J		
ins-	广州	1.0.0	标准版	实例配额数:50	运行中	按量付费	2022-03-01 16:17:06	控制台 删除 更多 ▼	<b>U</b>

3. 在左侧导航栏选择**服务列表**,选择某一具体服务,单击服务名进入服务实例页。您可以查看服务实例 IP、端口、协议、健康状态、隔离状态等信息。

e polaris.limiter(P	olaris)										
服务实例 路由规则	熔断规则	限流规则 服务	信息								
新建 删除	其他操作								请选择条件	进行过滤	Qφ
实例IP	端口	协议	版本	权重	健康状态 ▼	隔离状态 🔻	地域/可用区	创建时间	修改时间	操作	
•	-	grpc	-	100	健康	不隔离	/	2023-12-12 20:37:4	2 2023-12-12 20:37	:42 🖌 🔟	
实例ID											
健康检查 关闭											
实例标签 -											
共 1 条								10	条/页 🖂 🖣	1 /1页	► H

#### 场景二

如果您使用其他方式接入 Polaris(北极星),则需要在 TSF 控制台手动新建服务与服务实例,步骤如下:

1. 登录 TSF 控制台,在左侧导航栏选择Polaris(北极星)。

2. 选择目标引擎实例后,单击目标实例引擎操作栏的控制台,输入用户名和密码,进入 POLARIS MESH 控制台页面。



新建							默认开公网,如需关闭,讠	清前往访问控制页中	关闭。
ID/名称	地域 ▼	版本	产品版本	规格	状态	计费模式	默认账号:polaris 默认密码:polaris@ins-()	5	
ins-	广州	1.0.0	标准版	实例配额数:50	运行中	按量付费	2022-03-01 16:17:06	控制台 删除 更多 ▼	٢

- 3. 在左侧导航栏选择**服务列表**,选择某一具体服务,单击服务名进入服务实例页。
- 4. 在服务实例页面单击新建。
- 5. 在新建服务实例页,根据自身业务需求选择相关信息。

新建服务实例		×
实例IP *	多个IP以英文重导、英文分导、空格或法行分隔,每次最多添加100个IP	
靖口 *	多个領口以英文運号、英文分号、空格或操行分隔,每次最多添加100个領 口	
权重 *	- 100 +	
协议		
版本		
实例标签()	标签键 标签值 操作	
	无标签	
	添加标签	
健康状态		
开启健康检查		
是否隔离①		
	<b>提</b> 交 关闭	

○ 实例 IP: 填写服务实例 IP。

- 端口:填写服务实例端口。
  - ∴ 注意: 同一服务下 IP 和端口组合需唯一。
- 权重:设置实例占据的流量权重。
- 协议:可选,填写服务实例协议。
- 版本:可选,填写服务实例版本。
- 实例标签:可选,实例标签可用于标识实例的用处、特征,格式为 key:value 。
- 健康状态:关闭后,实例状态将显示异常,无法被调用。
- 开启健康检查:开启后,服务端负责检查服务实例的健康状态。
- 是否隔离:隔离状态下,主调方无法发现隔离的服务实例,无论实例 IP 是否健康。
- 6. 单击提交,完成新建服务实例。

#### 编辑服务实例

- 1. 在**服务列表**页面,选择某一具体服务,单击服务名进入服务实例页。
- 2. 在服务实例页面,选择目标服务实例,单击操作列的编辑按钮。



← polaris.checker(F	Polaris)										
服务实例 路由规则	熔断规则	限流规则 服务(	息								
新建 删除	其他操作								请选择条件进行	过滤	Q Ø
实例IP	端口	协议	版本	权重	健康状态 ▼	隔离状态 ▼	地域/可用区	创建时间	修改时间	操作	
•	8091	grpc	v1.17.0.1	100	异常	不隔离	/	2023-11-14 15:12:24	2023-11-26 09:08:18	× 10	
•	8091	grpc	v1.17.0.1	100	健康	不隔离	/	2023-11-14 15:09:13	2023-11-26 09:08:17	1	
·	8091	grpc	v1.17.0.1	100	异常	不隔离	/	2023-11-14 15:13:46	2023-11-26 09:08:17	/ 1	
• 🗆 =	8091	grpc	v1.17.0.1	100	健康	不開离	7	2023-11-14 15:10:51	2023-11-26 09:08:16	1	
共 4 条								10 -	条/页 14 4	1 /1页	► H

#### 3. 在编辑服务实例页,根据自身业务需求编辑相关信息。

4. 单击**提交**,完成编辑服务实例。

#### 删除服务实例

- 1. 在**服务列表**页面,选择某一具体服务,单击服务名进入服务实例页。
- 2. 在服务实例页面,选择目标服务实例,单击操作列的删除按钮。

polaris.che	cker(Polaris)										
服务实例 路由	规则 熔断规则	限流规则 服务	信息								
新建 删除	其他操作								请选择条件进行	于过滤	Q
实例IP	端口	协议	版本	权重	健康状态 ▼	隔高状态 ▼	地域/可用区	创建时间	修改时间	操作	
•	8091	grpc	v1.17.0.1	100	异常	不隔离	7	2023-11-14 15:12:24	2023-11-26 09:08:18	1	]
	8091	grpc	v1.17.0.1	100	健康	不隔离	/	2023-11-14 15:09:13	2023-11-26 09:08:17	1	
	8091	grpc	v1.17.0.1	100	异常	不隔离	/	2023-11-14 15:13:46	2023-11-26 09:08:17	1	
;	8091	grpc	v1.17.0.1	100	健康	不隔离	/	2023-11-14 15:10:51	2023-11-26 09:08:16	1	
专 4 条								10 -	条/页 H 4	1 /1页	i 🕨

#### 3. 在二次弹窗页面确认删除。

		×
确认删除实例		
删除后,无法恢复		
	确定取消	



## 服务实例隔离

最近更新时间: 2025-04-17 16:00:53

#### 操作场景

在微服务场景中,当服务提供者的某些实例出现异常时,一方面,需要避免服务消费者访问到异常实例,另一方面,需要保留异常现场,便于后续的问题排 查。北极星服务实例隔离的功能会将异常实例隔离,您可以登录实例排查问题,同时确保服务消费者不会访问到异常实例。在异常恢复后,您可以取消实例隔 离,恢复至正常使用。提升您业务的稳定性与质量。

例如在下图的示例场景中,存在服务 A 与服务 B 两个服务,服务 A 为服务消费者,服务 A 调用服务 B。服务B部署了3个实例,并注册至 Polaris(北极 星),服务 A 向 Polaris(北极星)查询服务 B 信息。

当服务 B 的实例2发生异常时,会导致服务 A 的部分调用失败,您可以将实例2设置为隔离状态。此时,Polaris(北极星)会将变更通知服务 A,服务 A 更 新访问服务 B 的 IP列表,从而实现服务A访问服务 B 时,不会访问到被隔离的实例,以保证服务成功调用。



本文介绍通过腾讯微服务平台控制台使用服务实例隔离的能力。

#### 操作步骤

#### 隔离服务实例

- 1. 登录 TSF 控制台,在左侧导航栏选择Polaris (北极星)进入北极星引擎实例列表页。
- 2. 单击目标实例引擎操作栏的控制台,输入用户名和密码,进入 POLARIS MESH 控制台页面。

新建							默认开公网,如需关闭,请前往访问控制页中关闭。
ID/名称	地域 👅	版本	产品版本	规格	状态	计费模式	默认账号: polaris 默认密码: polaris@ins-3
ins-	广州	1.0.0	标准版	实例配额数:50	运行中	按量付费	2022-03-01 16:17:06 更多 ▼ 🙂

- 3. 在左侧导航栏选择**服务列表**,选择某一具体服务,单击服务名进入服务实例页。
- 4. 选择异常实例,单击**其他操作**,单击修改隔离状态。

polaris.check	er(Polaris)											_
<b>服务实例</b> 路由规	则 熔断规则	限流规则 服务	信息									
STAR BURK	其他操作								请选择条件进行	nitat	Q	¢
- 实例IP	复制P	协议	版本	权重	健康状态 ▼	隔离状态 ▼	地域/可用区	创建时间	修改时间	操作		
> 🖬 🚥 🚥	修改健康状态	grpc	v1.17.0.1	100	异常	不隔离	/	2023-11-14 15:12:24	2023-11-26 09:08:18	/ 1	Ī	
• • • • • • •	修改隔离状态	grpc	v1.17.0.1	100	健康	不隔离	7	2023-11-14 15:09:13	2023-11-26 09:08:17	/ 1	Ì	
• • • •	8091	grpc	v1.17.0.1	100	异常	不隔离	7	2023-11-14 15:13:46	2023-11-26 09:08:17	/ 1	Ì	
• • • •	8091	grpc	v1.17.0.1	100	健康	不隔离	/	2023-11-14 15:10:51	2023-11-26 09:08:16	/ 1	Ì	
共 4 条								10 + 3	§/页 H ◀	1 71	页 →	H

5. 打开是否隔离的开关,单击提交。



编辑服务实例		×
是否隔离 ③		
	提交关闭	

返回服务实例页面,可以查看到服务实例的隔离状态已变更为**已隔离**。此时,服务被调用时,不会调度至该被隔离的实例上。

← polaris.checker(	Polaris)											
服务实例 路由规则	熔断规则	限流规则	服务信息									
新建 删除	其他操作								请选择条件进行	iting.	Q	ιφ
n 实例IP	端口	协议	版本	权重	健康状态 ▼	隔离状态 🍸	地域/可用区	创建时间	修改时间	操作		
> 🗹 📖 📖	8091	grpc	v1.17.0.1	100	异常	隔离	/	2023-11-14 15:12:24	2024-01-23 11:32:25	1	Ō	
	8091	grpc	v1.17.0.1	100	健康	不隔离	/	2023-11-14 15:09:13	2023-11-26 09:08:17	1	Ō	

#### 取消隔离

当实例正常后,即可取消服务实例隔离。

1. 在**服务实例**列表页面,选择需要恢复的实例,单击**其他操作**,单击修改隔离状态。

← polaris.check 服务实例 路由規	- polaris.checker(Polaris) 服务実例 路由規則 熔新規則 限決規則 服务信息										
新建 創除											
🔤 实例IP	复制IP	协议	版本	权重	健康状态 ▼	隔离状态 🍸	地域/可用区	创建时间	修改时间	操作	
> 🗹 🤃 7	修改仗重修改健康状态	grpc	v1.17.0.1	100	健康	隔离	7	2023-11-14 15:09:13	2024-01-23 11:42:07	1	
→ 3 5	修改隔离状态	grpc	v1.17.0.1	100	健康	不隔离	7	2023-11-14 15:10:51	2023-11-26 09:08:16	1	
共 2 条								10 👻 3	条/页 H 4	/1页	F. H.

#### 2. 关闭是否隔离的开关,单击提交。

编辑服务实例		×
是否隔离 ①		
	提交关闭	

3. 返回服务实例页面,可以查看到服务实例的隔离状态已变更为不隔离。此时,服务被调用时,可以调度至该实例。

← polaris.checker	polaris.checker(Polaris)											
服务实例路由规则	熔断规则	限流规则 服务	息									
新建 删除	其他操作								请选择条件进行	过速	Q	φ
- 实例IP	端口	协议	版本	权重	健康状态 ▼	隔离状态 ▼	地域/可用区	创建时间	修改时间	操作		
	8091	grpc	v1.17.0.1	100	健康	不隔离	/	2023-11-14 15:09:13	2024-01-23 11:35:42	/ 11		
• • • • •	8091	grpc	v1.17.0.1	100	健康	不隔离	/	2023-11-14 15:10:51	2023-11-26 09:08:16	/ 1		
共 2 条								10 -	条/页 H ◀ 1	/1页	► H	



## 服务实例元数据管理

最近更新时间: 2025-04-17 16:00:53

#### 使用场景

**元数据概念:**服务实例通常带有一系列的标签信息,例如实例所属的机房信息、地域信息、环境信息等,这些信息统称为服务实例的元数据。实例元数据通常 也叫实例标签。

#### 使用场景:

• 实例注册到注册中心时,会带上自身的元数据信息。当消费方从注册中心获取到实例时,既可以同时获取到每个实例的元信息。



• 通过元数据路由的能力,将指定特征的流量转发至指定特征的服务实例上,可用于灰度发布等场景。



#### 操作步骤

#### 设置元数据信息

为服务实例设置元数据的方法主要有以下四种,您可根据实际情况选择最适合的方式即可。如您同时通过几种方式设置,则生效优先级为:**自定义SPI > 环境** 变量 > 启动参数 > 应用配置。云控制台设置的标签和客户端标签不冲突的情况下同时存在,若 key 相同的情况下,取最新的数据值。

#### 方案1: 通过腾讯云控制台设置

- 1. 登录 TSF 控制台,在左侧菜单栏选择 Polaris (北极星)进入引擎实例列表页。
- 2. 单击选择目标实例后,进入实例详情页,在左侧菜单栏选择**服务管理**,进入服务管理页面。



← •test (ins-	a)								🗋 操作指南
<ul> <li>引擎管理</li> </ul>	<b>服务列表</b> 服务别名								
- 命名空间									
注册中心	新建制除						请选择条	牛进行过滤	ର୍⊜୍ଟ
。 服务管理	服务名	命名空间 了	881']	业务	健康实例/总实例	创建时间	傳改时间	操作	
治理中心 • 动态路由	pol r.cls-     ove	Polaris	-	-	1/4	2024-11-29 11:10:20	2024-11-29 11:10:20	编辑 删除	
- 就近路由	▶ _ htt	polaris-test			0/0	2024-11-27 20:18:40	2024-11-27 20:18:40	编辑 删除	
<ul> <li>培断降级</li> <li>访问图流</li> </ul>	► kul inet	default	÷	÷	2/2	2024-11-29 11:10:20	2024-11-29 11:10:20	编辑 删除	
<ul> <li>全链路友度</li> </ul>	► ecl	tke-polaris-auto			1/1	2024-11-29 11:10:20	2024-11-29 11:10:20	编辑 删除	

#### 3. 单击目标服务名,进入服务详情页。单击目标实例的编辑按钮,即可添加或修改目标实例的元数据信息。

ins-1	9/Polaris/polaris-co	ntroller.cls-										L) 排(
基本信息 服务名 pola 可见性 仅当	laris-controller.cls-< 当前命名空间可见	命名空间 Polaris 就近路由 关闭	版本号 容错保护	- 关闭		服务标签(0个) 创建时间	- 2024-11-29 11:10:20	健康/总实例数 傳改时间	1/4 2024-11-29 11:10:20	描述 -		19
服务实例	无损上下线         路由規           削除         其他操作            第0	则 焰断规则 限流规则	权限管理	投票	41.05	# <b>杰</b> 平	國憲状态 豆	<b>协雄/可用区/机械</b>	他課題寸(6)	健康状态:健康	Q	. @ (
► [] 10	80	-		100	92.0K		不隔离	-1-1-	2024-12-11 19:21:29	2024-12-11 19:21:29	编辑 别除	
<b>着服务实例</b> 用IP・ □・ ■・ 文	10. 80 - 100 +				×							
标签 ①	松登键 	₩33(t) ■	操作 × ×									
汉态 健康检查 ① 检查	<ul> <li>         ・ ・ ・</li></ul>				,							
隔离 ()												

#### 方案2: 通过项目配置文件设置

在 Spring Cloud 项目里的 application.yml 中配置元数据信息,以下示例中设置 idc=shanghai, env=dev1的两个元数据。应用在启动注册时,会自 动读取配置文件并带上 idc=shanghai 和 env=dev1 两个元数据信息。



#### 方案3:通过应用启动参数设置

Spring Boot/Spring Cloud 应用配置文件定义的配置项都可以通过 -D 启动参数设置,例如通过以下方式覆盖 env 值为 dev2:

Java -Dspring.cloud.tencent.metadata.content.env=dev2 -jar demo.jar

#### 方案4: 通过环境变量设置



环境变量的方式完全跟运行的应用解耦。Spring Cloud Tencent 约定了前缀 SCT\_METADATA\_CONTENT\_的环境变量为应用的元数据信息。例 如:

SCT\_METADATA\_CONTENT\_IDC=shanghai SCT\_METADATA\_CONTENT\_ENV=dev1

#### 方案5: 自定义实现SPI

前面三种方式为 SCT(Spring Cloud Tencent 的缩写) 内置的方式,但是并不一定符合每个公司自己的规范或者实现。例如:

- 把元数据放到机器上的某一个配置文件里,例如 /etc/metadata。
- 启动时,调用 CMDB 的接口获取元信息。

所以 SCT 定义了 InstanceMetadataProvider SPI,支持通过实现内部提供的接口来进行读取,开发者可以实现相关的方法,提供对应的元数据,方便 用户自己实现元数据来源。SCT 在注册前,回调 SPI 获取元数据信息,并注册到注册中心。示例代码:



#### 查看元数据信息

- 1. 登录 TSF 控制台,在左侧导航栏选择Polaris (北极星)进入北极星引擎实例列表页。
- 2. 选择目标引擎实例,在引擎实例详情页,进入服务管理页面。

← test-cls (ins-d	)							L) 操	作指南
<ul> <li>引擎管理</li> </ul>	<b>服务列表</b> 服务别名								
<ul> <li>命名空间</li> </ul>									
注册中心	新建 删除						请选择条件进行	行过滤 Q 参 C	;
- 服务管理	服务名	命名空间 V	部门	业务	健康实例/总实例	创建时间	修改时间	操作	
<ul> <li>· 动态路由</li> </ul>	E IgTest	echo	-	-	1/1	2024-08-22 14:07:45	2024-08-22 14:07:45	编辑 删除	
- 就近路由	► □ p r.cls-	Polaris	-	-	1/2	2024-08-02 10:28:02	2024-08-02 10:28:02	编辑剧除	
<ul> <li>/ 熔断降级</li> <li>· 访问限流</li> </ul>	► □ F	Polaris	-	polaris	2/2	2021-09-06 15:55:07	2021-09-06 15:55:09	编辑 删除	
- 全链路灰度	共 3 条						20 ~ 条/页 4 4	1 /1页 ▶ №	

3. 单击目标服务名,进入服务详情页。展开目标实例左侧三角图标,即可查看实例标签(元数据)信息。



÷	ins-(	/echo/Ech	oServerGolar	ngTest										
	基本信	息												
	服务名 可见性	EchoServerGolangTest 仅当前命名空间可见	命毛	G空间 echo II路由 关闭		版本号 容错保护	- 关闭		服务标签(0个) 创建时间	- 2024-08-22 14:07:45	健康/总实例数 修改时间	1/1 2024-08-22 14:07:45	描述 -	
	服务实	例 无损上下线	路由规则	熔断规则	限流规则 权限管	理								
	新建	删除 其他 实例IP	操作	10.10	版本		权雷	an market	* 7	四直计本 豆	地域/可用区/机座	合成要取101	健康状态:健康	Q 損作
	•	127	8080	-	-		100	健康		不隔离	-1-1-	2024-08-22 14:07:45	2024-08-22 14:30:	编辑删除
	实例ID 健康检	71fc1 章 关闭		>8324										
	实例标:	env:pre ; 12312:1231	3											
	共1条											20 ~	条/页 № ◀	1 /1页



## 服务别名

最近更新时间: 2025-04-09 11:24:22

### 操作场景

服务别名可以看作是服务的映射,访问服务别名等同于访问服务,允许多个服务别名指向一个服务。服务别名主要适用于以下场景:

- 您不想用 service 名作为 polarismesh 服务的名字。例如您希望 polarismesh 的服务名是大写,但是 K8s 的 service 名限制只能小写。这时可以 使用别名注解指定一个大写的 polarismesh 服务名。
- 您希望将某个 namespace 下的某个 service 暴露到另外命名空间中。这时可以使用服务别名指定另一个命名空间。
- 对不同的外部客户需要用不同的应用名提供服务,但是实际上后端真实的服务是同一个。

#### 操作步骤

#### 新建别名

- 1. 登录 腾讯微服务平台控制台,在左侧菜单栏选择Polaris (北极星)进入引擎实例列表页。
- 2. 单击目标实例引擎操作栏的控制台,输入用户名和密码,进入 POLARIS MESH 控制台页面。

	新建							默认开公网,如需关闭,	请前往访问控制页中:	关闭。
	D/名称	地域	版本	产品版本	规格	状态	计费模式	默认账号: polaris 默认密码: polaris@ins-3 5		
1	ns-	广州	1.0.0	标准版	实例配额数: 50	运行中	按量付费	2022-03-01 16:17:06	控制台 删除 更多 ▼	٢

3. 在左侧导航栏选择**服务别名**,单击新建按钮,根据自身业务需求填写以下配置信息:

参数	说明
服务别名	服务别名为数字、英文字母等字符组成的字符串。
命名空间	选择服务别名所属的命名空间,和映射的服务可以不在同一个命名空间。可根据您的业务需要自行选择。
指向服务	服务别名指向的服务。
描述	您对服务别名的描述。

#### 4. 单击提交, 创建服务别名。

新建服务			
服务别名 *	test_service		
命名空间・	Test		Ŧ
指向服务 ★	polaris.checker (Polaris)		Ŧ
描述	长度不超过1024个字符		
		提交关闭	

#### 查看服务别名

1. 服务别名创建完成后,可以查看服务别名。



<b>服务别名</b> 服务别名可以看作是服	3务的映射,访问服务别名等	<sup>年同于访问服务,允许多个服9</sup>	务别名指向同一个服务	<del>Ş</del>		
新建别名    删图	涂				请选择条件进行过	滤 <b>Q </b>
服务别名	命名空间 ▼	指向服务	备注	创建时间	修改时间	操作
test_service	Test	polaris.checker Polaris	-	2021-12-07 17:48:48	2021-12-07 17:48:48	编辑 删除
共 1 条				20 -	条/页 🛛 🖣	1 /1页 )

- 2. 单击 🍸 图标,可根据命名空间进行过滤。
- 3. 支持基于命名空间与指向服务的检索。

请选择条件进行过滤			Q ¢ ¢
选择资源属性进行过滤	147 3 k n + 3 - 3	19.16	
命名空间	修改时间	操作	
指向服务			



## 关联 K8s 集群

最近更新时间: 2025-05-16 14:47:32

#### 操作场景

北极星提供将 K8s 集群关联到 Polaris 的能力,Polaris Controller 可以同步您 Kubernetes 集群上的 Namespace,Service,Endpoints 等资源 到 Polaris 中,从而实现 K8s Service 自动注册到 Polaris ,使用 Polaris API 和多语言 SDK 可以访问,使用 gRPC 和 Spring Cloud 等开源框架也 可以访问。主要适用于以下场景:

- 异构系统与多技术栈场景下, SpringCloud 等框架服务调用 K8s 集群服务。
- 跨集群场景下的服务调用。

本文介绍通过 TSF 控制台使用 K8s 集群的能力。

#### 操作步骤

#### 创建引擎

- 1. 登录 TSF 控制台,在左侧菜单栏中选择 Polaris (北极星)。
- 2. 在引擎实例列表页面,单击目标引擎的"ID",进入基本信息页面。
- 3. 在顶部页签单击 K8s 集群,单击关联集群。

#### () 说明:

- 1. 添加容器集群会安装 Polaris-Controller 组件,请确保您的集群至少有 1C2G 的空闲资源,否则会添加失败。
- 2. 容器集群中安装 Polaris-Controller 组件后默认支持 K8s Service 服务同步(按需同步)和 Java Agent 自动注入能力。您也可以在控制台 选择**全量同步**使用 k8s 全量同步方式。
- 集群选择:可以根据自身业务需求选择目标关联的 K8s 集群,支持 TKE (容器集群) / EKS (弹性容器集群)。
- K8s Service 同步:也可以根据业务的特点选择服务的同步方式,支持全量同步(集群内全部 service 自动同步至北极星)/按需同步(仅指定的 namespace 或者 service 自动同步至北极星)。
- Java Agent 接入:容器集群中安装 Polaris-Controller 组件后默认支持 Java Agent 自动注入能力。

添加集群		
<ol> <li>添加容器</li> <li>容器集群</li> <li>以选择下;</li> </ol>	集群会安装Polaris-Controller组件,请确保您的集群至少有1C2G的空闲资源,否则会添加失败 中安装Polaris-Controller组件后默认支持K8s Service服务同步(按需同步)和Java Agent自动注入能力。 您也可 方"全量同步"按钮使用k8s 全量同步方式。	
集群 🛈 *	tke       ②     ④       如果现有的网络不合适,你可以 <b>新建集群 ビ 刷新</b> ●	
K8s Service 同步	按需同步     全量同步       K8s 集群中,仅指定的 namespace 或者 service 将自动同步至北极星。       您需要在指定的 namespace 或者 service 上添加 annotation,使用指引 [2]	
Java Agent接入	用户可以通过添加注解的方式,在 Spring Cloud 应用里注入 Java Agent。 Java Agent 自动引入 Spring Cloud Tencent 依赖,接入北极星注册配置治理中心。 <b>使用指引 <sup>[2]</sup></b>	
	添加取消	

#### 原理说明

- 在 TSF 控制台关联 tke/eks 集群后,会自动在您对应的 tke/eks 集群中部署 polaris-controller 。具体请参见 资源清单。
- polaris-controller 默认会同步 K8s 集群所有的 namespace, service 和 endpoints 。具体请参见 polaris-controller 同步行为 。
- 您可以在 K8s 的 service 指定注解,操作 polaris-controller 同步的行为。具体指引和示例请参见 polaris-controller 支持的注解。

#### 资源清单



#### 在您的 tke/eks 集群创建的 K8s 资源清单如下:

资源类型	资源名	资源用途
namespace	polaris-system	polaris-controller 和其相关配置都在这个命名空间下
deployment	polaris-controller	提供同步服务的工作负载
configmap	injector-mesh, polaris-sidecar- injector	polaris-controller 使用的配置文件
serviceAccount	polaris-controller	提供访问 K8s 资源需要的权限
clusterRole	polaris-controller	提供访问 K8s 资源需要的权限
clusterRoleBinding	polaris-controller	提供访问 K8s 资源需要的权限
mutatingWebhookConfigurati on	polaris-sidecar-injector	提供自动注入能力

#### 可以使用下面图片中的命令查看到上述的资源。

[root@VM-1-7-centos ~]# kube	ectl get ns	grep polaris-s	ystem
polaris-system Active	16d		
[root@VM-1-7-centos ~] # kube	ectl get deplo	oy -n polaris-s	ystem
NAME READY	UP-TO-DATE	AVAILABLE A	GE
polaris-controller 1/1	1	1 1	.6d
[root@VM-1-7-centos ~]# kube	ectl get confi	.gmap -n polari	.s-system
NAME	DATA AGE		
injector-mesh 2	1 16d		
polaris-sidecar-injector 2	2 16d		
[root@VM-1-7-centos ~] # kube	ectl get servi	.ceaccount -n p	oolaris-system   grep polaris-controller
polaris-controller 1	16d		
[root@VM-1-7-centos ~]# kube	ectl get clust	errole   grep ;	polaris-controller
polaris-controller			2021-11-02T07:57:51Z
[root@VM-1-7-centos ~]# kube	ectl get clust	errolebinding	grep polaris-controller
polaris-controller			ClusterRole/polaris-controller
[root@VM-1-7-centos ~]# kube	ectl get mutat	ingWebhookConf	iguration   grep polaris-sidecar-injector
polaris-sidecar-injector	1	16d	

#### polaris-controller 同步行为

polaris-controller 默认会同步 K8s 集群所有的 namespace, service 和 endpoints 的行为。

资源类型	资源行为
namespace	● 创建 namespace 时,会在 polarismesh 创建一个同名的命名空间。 ● 删除 namespace 时,不会移除 polarismesh 的同名命名空间。
service	<ul> <li>创建 service 时,不论创建的 service 的类型,都会在 polarismesh 对应的命名空间下创建一个 同名的服务。</li> <li>删除 service 时,不会移除 polarismesh 对应命名空间下的服务,会移除 polarismesh 中对应 服务的实例(只移除本 K8s 集群同步到 polarismesh 对应服务下的实例)。</li> </ul>
endpoints	<ul> <li>某个 service 的 endpoints 创建时,会把 endpoints 列表中 IP 和 port 每个组合,作为 polarismesh 对应服务下的一个服务实例,同步到 polarismesh 。如果 IP 在 endpoints 的就绪 地址列表中,则 polarismesh 服务实例状态为健康,如在未就绪地址列表中,则 polarismesh 状 态为不健康。</li> </ul>
	<ul> <li>某个 service 的 endpoints 地址列表受化的, 会动态的同步到 polarismesh 。地址列表增加, 则 在 polarismesh 服务下注册新实例; 地址列表减少,则反注册 polarismesh 服务下对应的实例。</li> <li>某个 service 的 endpoints 被移除时,本 K8s 集群同步到 polarismesh 对应服务下的实例都会 从 polarismesh 的服务中移除。</li> </ul>

### polaris-controller 支持的注解



注解名称	注解说明
polarismesh.cn/sync	是否同步这个服务到 polarismesh。true 同步,false 不同步,默认不同步。
polarismesh.cn/aliasService	把 k8s service 同步到 polarismesh 时,同时创建的服务别名的名字。
polarismesh.cn/aliasNamespace	创建的别名所在的命名空间,配合 polarismesh.cn/aliasService 使用。

#### 如何按需同步

腾讯云

polaris-controller 默认会同步 K8s 集群所有的 service。某些场景下,您可能只想同步部分 service 到 polarismesh,这时可以将同步方式选择为 (按需同步),同时使用 polarismesh.cn/sync 注解来指定需要同步的服务。

• 按需同步单个服务: 下面的 service 创建时, polaris-controller 会将该服务同步到 polarismesh。

```
apiVersion: v1
kind: Service
metadata:
name: details
annotations:
polarismesh.cn/sync: true
... ...
```

• 按需同步命名空间下所有的服务:为命名空间打上标签,polaris-controller 会将该命名空间下所创建的所有服务同步到 polarismesh。

```
apiVersion: v1
kind: Namespace
metadata:
name: default
annotations:
polarismesh.cn/sync: "true"... .
```

#### 创建别名示例

polaris-controller 默认会以 service 名字,创建一个对应的 polarismesh 服务。可能有以下情况,需要创建服务别名:

- 您不想用 service 名作为 polarismesh 服务的名字。例如您希望 polarismesh 的服务名是大写,但是 K8s 的 service 名限制只能小写。这时可以 使用别名注解指定一个大写的 polarismesh 服务名。
- 您希望将某个 namespace 下的某个 service 暴露到另外命名空间中。这时可以使用别名注解指定另一个命名空间。

下面示例的 service 创建时,polaris-controller 会在 polarismesh 的 development 命名空间下创建一个名为 productpage 的服务。同时也会在 Development 命名空间下创建一个名为 Productpage 的服务别名。

```
apiVersion: v1
kind: Service
metadata:
   namespace: development
   name: productpage
   annotations:
      polarismesh.cn/aliasService: Productpage
      polarismesh.cn/aliasNamespace: Development
... ...
```

#### 如何确认polaris-controller版本

点击集群ID,跳转到容器控制台页面,查看polaris-system命名空间下polaris-controller这个statefulset的镜像版本。



Deployment	Statefulset	Daemonset	Job	Cronjob					
新建监控	Workload	Мар			polaris-system	~	名称只能搜索	一个关键字,	Label格式
名称	Lab	els	Selector	4	镜像()	可观察/排	朝望Pod数量	Request/Li	imits
polaris-controller		ud-app:polaris	k8s-app:po	laris-con	confocs.tencentyun.com/	1/1		CPU : 1/ 1 机 内存 : 2000 Mi	亥 // 2000
45 A -									



## 跨地域使用指南

最近更新时间: 2025-04-17 16:00:53

### 功能描述

北极星支持跨地域部署,提供全局注册配置治理中心。支持将多个地域的北极星管控节点组成一套北极星集群,即同大区内跨地域组网。例如:北极星同一个 实例中,支持广州节点和北京节点。选择广州作为北极星所在的主地域,多个地域的数据确保一致性。

主地域和从地域对于客户端接入和使用上来说,没有差异,均提供服务注册发现能力。数据最终存储在主地域。

#### 使用场景

#### 场景1: 跨地域服务注册发现

北极星提供服务跨地域注册发现,可满足业务服务跨地域调用的场景。在电商、游戏等场景下,业务一般会多个地域部署,以提升系统的稳定性,降低客户访 问时间。但是由于某些业务场景的特殊性或者数据同步的复杂性,部分服务无法在多个地域部署,需要跨地域服务调用。例如:某电商应用,北京地域服务访 问上海地域的服务。



#### 场景2:跨地域容灾场景

北极星支持同城就近路由,跨城容灾。在跨地域容灾场景下,一个服务在两个地域部署节点,正常情况下,就近访问。在某个地域发生故障后,则路由至另一 个地域。例如:当广州的 provider 不可用时,consumer 会跨城调用上海的 provider。



### 操作步骤

#### 步骤1: 跨地域节点组网

- 1. 登录 TSF 控制台,在左侧菜单栏选择Polaris (北极星)后进入引擎实例列表页。
- 2. 单击新建,创建实例,详情参见 引擎管理。



#### 部署架构选择**跨城多可用区**,选择跨地域的目标地域,并配置每个地域集群的客户端接入节点数、集群网络信息。

部署架构	同城多可用区	跨城多可用区						
	提供高可用版注册中心,默	认支持同城多活						
地域配置	中国大陆	中国自测	中国预发	中国提测	亚太地区	欧洲地区	美洲地区	
	地域	广州 主地域的北极星节点不可被	✓					删除
	规格	接入节点数量:	~					
	集群网络 🚯	vp	1) 🗸 si	ut	3) ~ ⑦ 共	个IP, 剩 个可用		
		如果现有的网络不合适,修	R可以新建集群网络 🗹 或者	新建子网 🖸 刷新 🗘				
				添加地域配置				

## 步骤2: 服务注册发现

应用场景	参考文档			
	polaris-Java 接入			
Java 应用开发	gRPC-Java 接入			
	Spring Cloud 接入			
	polaris-Go 接入			
Go 应用开发	gRPC-Go 接入			
	Dubbo-Go 接入			
网关接入	Nginx 接入			
迁移指南	Eureka 迁移指南			

## 服务治理(治理中心)

## 服务路由

## 就近路由

最近更新时间:2025-04-0911:43:32

### 操作场景

本文将帮助您快速了解如何使用Polaris(北极星)的就近路由能力。

### 前提条件

已创建 PolarisMesh 北极星网格,请参见 创建 Polaris(北极星)治理中心。

## 操作步骤

使用北极星就近路由的能力,主要包括两部分:

- 在控制台开启就近路由能力。
- 在客户端通过腾讯云 API 或者环境变量获取地域信息。

### 步骤1: 在控制台开启就近路由能力

1. 登录 腾讯微服务平台控制台,在左侧导航栏选择Polaris(北极星),进入引擎实例列表页。

2. 单机目标服务实例在左侧导航栏,**治理中心**下,单击**服务管理**,上方下拉框选择目标北极星实例。

3. 单击**目标服务**,单击**路由规则** Tab,选择**就近路由**后开启**就近路由**按钮。

### 步骤2:在客户端通过腾讯云API或者环境变量获取地域信息

详细参见开发文档: Java SDK 开发 > 就近访问。


# 动态路由

最近更新时间: 2025-04-09 11:43:32

## 概述

动态路由功能是指用户根据路由规则控制流量分发的机制,将满足规则的流量转发至目标实例分组。该功能通常用于灰度发布、容灾降级等场景。为了满足客 户的定制化需求,北极星支持用户为服务实例打上自定义标签,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。

## 应用场景

灰度发布场景: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,设置灰度特性的流量被分配到某个版本号中。通过路由功能,用户也可以为指定灰度用户的请求转发至灰度版本上。路由能力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。



**容灾场景**: 业务为了应用系统整体的健壮性以及高可用,通常会跨可用区、跨地域进行应用部署。服务路由提供配置目标服务优先级的能力,支持在某可用
 区或者某地域目标服务发生故障后,将请求路由至次优先级的实例分组上。助力用户更加便捷的实现多活方案。



## 实现原理

当 A 服务调用 B 服务时,先从注册中心获取全量 B 服务地址信息。当没有服务路由,直接进行负载均衡,根据负载均衡算法从全量 B 服务地址中挑选一个服 务实例发起服务调用。 当加入服务路由阶段后,挑选服务实例分为两个阶段:

- 阶段一: 从全量服务地址中根据路由规则选取一批目标服务实例地址。
- 阶段二:从阶段一选取的一批目标服务实例地址中,再根据负载均衡算法挑选一个实例。



#### 使用说明

要实现服务路由需要完成两部分操作:

- 在控制台上,设置路由规则。
- 客户端获取路由规则, 根据规则来分发请求。

### 步骤1:设置路由规则



- 1. 登录 腾讯微服务平台控制台。
- 2. 在左侧导航栏,选择 Polaris (北极星),在实例详情页选择目标引擎实例。
- 3. 单击左侧导航栏动态路由,进入动态路由规则展示页面,单击新建路由规则。

规则名称 •					
	最长64个字符				
描述					
优先级 🛈 🕯	- 0 +				
匹配条件	<b>主调服务</b> 主调请求按照匹配规则匹配成功后,将按照当前规则进行目标服务路由 」	点击切换主被	<b>皮调服务</b>	▶ 被调服务 请求会按照规则路由到目标服务	5分组
	命名空间· 全部命名空间 *		命名空间 *	请选择命名空间	v
	服务名称• <b>全部服务</b>		服务名称•		

- 规则名称:填写路由规则名称,不超过64个字符。
- 优先级:优先级数字设置越小,规则匹配顺序越靠前。
- 匹配条件: 配置路由规则的主调服务(服务消费者)和被调方(服务提供者),其中主调服务支持选择全部命名空间和全部服务。您可按需选择。

来源服务的请求满足以下四	配条件			
请求头(HEADER)	▼ version		等于 v2_beta	
+ 添加				
将转发至目标服务的以下实	例分组,按优先级转发,可	「拖动调整优先级:		
第 第【O】优先级 优先级	常用于容灾场景,高优先级实例	故障后,路由至次优先	级实例。相同优先级下分组权重加和为100。	
实例分组名称	权重 (1)	是否隔离	实例标签	
灰度分组	100		key:version 等于 value:v2_beta(值) × +	×
+添加实例分组				

- 路由策略支持配置多条,按顺序进行匹配。可拖动调整顺序。若规则全未匹配上,则请求拒绝。
- 流量匹配策略,非必选。
  - 流量参数位置支持:请求头(Header)、请求Cookie、请求参数(Query)、方法(Method)、主调IP、路径(Path)、自定义。
  - 逻辑关系支持:等于、不等于、包含、不包含、正则表达式、范围表达式。
  - 流量无差别,按百分比进行灰度的场景下,删除流量匹配策略即可。
- 实例分组权重:
  - 相同优先级下,实例分组权重加和为100。
  - 流量百分比场景下,可为不同实例分组配置不同流量百分比。如下图所示,为灰度版本分配10%的流量,90%的流量路由至基线版本。



			无匹配条件	
添加				
发至目标服务的以下实	<b>实例分组,按优先级转发,</b> 可	拖动调整优先级:		
第 [0] 1元元级 优先级	段常用于容灾场景,高优先级实例;	故障后,路由至次优先	级实例。相同优先级下分组权重加和为100。	
<b>第【U】 优先级</b> 优先级 实例分组名称	投常用于容灾场景,高优先级实例的 权重 <b>①</b>	故障后,路由至次优先 <b>是否隔离</b>	级实例。相同优先级下分组权重加和为100。 实例标签	
第【U】 (九先级) 优先级 实例分组名称 灰度分组	2常用于容灾场景,高优先级实例: 权重 ③ 10	效障后,路由至次优先 是否隔离	級实例。相同优先级下分组权重加和为100。 实例新签 key:version 等于 value:v2_beta(值) × +	×

- 是否隔离: 隔离通常用于现网故障或者上线验证场景,先手动摘除,待故障恢复或者验证通过后解除隔离。
- 优先级:优先级常用于容灾场景,高优先级实例故障后,路由至次优先级实例。
- 4. 单击**提交**,完成新建规则。

#### 步骤2: 客户端开发添加相关逻辑

根据您对应的接入方式查看开发文档下**动态路由**相关的篇章,以使用 Spring Cloud 接入为例,可参见文档 Spring Cloud 应用开发 > 动态路由。

# 熔断降级

# 服务熔断与主动探测

最近更新时间: 2025-04-17 16:00:53

# 场景说明

Polaris(北极星)支持可视化熔断规则管理,支持设置服务、实例、API 三种隔离级别的熔断规则。**在常见的熔断器的基础之上,北极星还额外支持故障主 动探测**。

### 熔断原理

### 定义

**服务熔断定义**:当下游的服务因为某种原因导致服务不可用或响应过慢时,上游服务为了保证自己整体服务的可用性,不再继续调用目标服务,直接返回。当 下游服务恢复后,上游服务会恢复调用。

### 熔断器状态

服务熔断中涉及到关键概念熔断器。

#### 1. 主动探测能力关闭的情况下,熔断器的状态转化如下:



**1.1 最开始处于** closed 状态,一旦检测到错误或慢响应等达到一定阈值,便转为 open 状态,此时不再调用下游目标服务。

1.2 等待一段时间后,会转化为 half open 状态,尝试放行一部分请求到下游服务。

1.3 一旦检测到响应成功,回归到 closed 状态,也即恢复服务;否则回到 open 状态。

其中熔断器从 close 变为 open 状态要同时满足以下2个条件:

- 前提条件:在滑动时间窗口内至少有一定数量的请求(即最少请求数)
- 指标达到阈值: 在滑动时间窗口内统计的错误请求率或慢请求率达到一定阈值
- 2. 打开主动探测能力,断路器的状态流转如下:



在业务请求的同时,加上主动探测的请求。在没有业务请求时,也能及时了解业务运行状态。

## 隔离级别及场景



#### 北极星支持服务、实例、接口 三种隔离级别的熔断规则:

隔离级 别	请求统计范围	熔断对象	适用场景
服务	下游目标服务的所有实例的 所有 API	服务	当下游服务属于不主要的业务,可以熔断所有实例
实例	下游目标服务的单个实例的 所有 API	达到熔断触发条件的实 例	当下游服务属于比较重要的业务,只对异常的实例进行熔断,避免所有实例被 熔断后导致服务雪崩
接口	下游目标服务的所有实例的 指定 API	达到熔断触发条件的单 个接口	下游服务不同 API 的重要程度不同,需要根据不同 API 设置不同的熔断策略

# 使用说明

要实现服务熔断需要完成两部分操作:

- 在控制台上,设置熔断规则、主动探测规则。
- 客户端接入熔断能力。

#### 步骤1:设置熔断规则

- 1. 登录 TSF 控制台。
- 2. 在左侧导航栏,选择 Polaris (北极星),在实例详情页选择目标引擎实例。
- 3. 单击左侧导航栏**熔断降级**,进入熔断规则展示页面,单击**新建熔断规则。**

最长64个字符					
(先级 ①	•				
·26.8 ()	T				
	主调服务			▶ 目标服务	
主调	服务		被调服务		
命名空	<b>间</b> · 请选择命名空间	~	命名空间•	请选择命名空间 🗸 🗸	
服务名	郗•		服务名称 •		
3浙和/8 服务	案例 接口				
浙策略					
					×
错误判問	f条件 满足以下任意应答条件的请求会被标识为错误请求	ł			
	返回码 > 全匹配 >		+		
熔断触发	<b>2条件</b> 满足以下任意条件可触发熔断				
熔断触发	<b>线条件</b> 满足以下任意条件可触发缩断 错误事义 2= 50 % 统计周期	30 秒 是小语史数	5 1 +		
熔断触发	<b>1条件</b> 满足以下任意条件可触发熔断 错误事 ✓ →= 50 % 统计周期	30 秒 最小请求数	5 个 +		
<b>培新触发</b>	<b>2条件</b> 满足以下任意条件可触发缩断 错误率 ✓ →= 50 % 统计周期	30 秒 最小请求数	5 个 +		
<b>培新数3</b> + 添加增新資源 1.新炊業業務	<b>2条件</b> 满足以下任意条件可触发缩断 错误率	30 秒 最小请求数	5 个 +		
<b>和新敏3</b> + 透加增新推 時快度策略	<ul> <li> <b>該条件</b> 满足以下任意条件可触发缩断         <ul> <li> </li> <li></li></ul></li></ul>	30 <sup>4</sup> 9 最小请求数 1, 满足条件后即可结束培断状;	5 个 + 5 个 +	明明明新状态	
<b>培新能</b> 3 + <u>添加</u> 熔新設 時代支算部 <b>培新快</b> 支算部	<ul> <li>2条件 满足以下任意条件可触发缩断</li> <li>1 24 (1) (1) (1) (1) (1) (1) (1) (1) (1) (1)</li></ul>	30 秒 最小请求数 ), 满足条件后即可结束缩断状;	5 个 + 念検量业务请求, 否则重新的	到柳新状态	
增新数3 + 添加增新数0 描板道策新 描述 描述	2条件 演足以下任意条件可触发缩断 错误率 → >= 50 % 統计周期 6 1 送入場紙状态后,通过设置超时探测或者主动探测规则 断时长 60 秒	30 <b>秒</b> 最小请求数 ), 满足条件后即可范束角断状	5 个 <b>+</b>	2到段新状态	
<b>编新数3</b> + <b>派加段新推</b> - 海動機断推 - 海新快動 - 道 - 道	集体 満足以下狂意条件可触发描紙   編集本 > >= 50 % 統計周期   は 込入領紙状态后, 通过设置超封採測或者主动採測規則   新規 60 82	30 <b>秒</b> 最小请求数 ), 满足条件后即可结束缩断状;	5 个 + 5 你算业务请求, 否则重新	到细新状态	
<ul> <li>2月時後3</li> <li>第二日の日の日の日の日の日の日の日の日の日の日の日の日の日の日の日の日の日の日の</li></ul>	<ul> <li>         ・ ・・・・・・・・・・・・・・・・・・・・・・・・・・・</li></ul>	<ul> <li>30 秒 最小请求致</li> <li>30 点, 满足条件后即可范束编断状:</li> </ul>	5 个 + 态快复业务请求, 否则重新0	到细胞状态	
<ul> <li>第時位3</li> <li>・添加場断第2</li> <li>・添加場断第2</li> <li>・添加場断第2</li> <li>・</li> <li></li></ul>	<ul> <li>         は課年 、 &gt;= 50 % 統計周期         <ul> <li></li></ul></li></ul>	30 <b>秒</b> 最小请求数 ], 课起条件后即可结束相断获;	5 个 +	的推断状态	
場断他3 ・ 添加増新焼き - 添加増新焼き - 通勤焼新焼業 - 二 -	<ul> <li>         ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・</li></ul>	30 <b>秒</b> 最小请求致 ), 满足条件后即可结束缩断状;	5 个 + 容核重业务请求, 否则重新f	到细带状态	

• 规则名称:规则名,需全局唯一,最长64个字符。



- 描述: 规则的描述信息,用于补充规则的说明。
- 匹配条件主要用于决定熔断规则的适用范围,客户端根据匹配条件来过滤本地调用所适用的熔断规则。
  - 主调服务: 配置作为主调方的服务名和命名空间,可选。不配置则默认对所有的主调生效。
  - 被调服务: 配置被调的服务名和命名空间,可选。不配置则默认对所有的被调生效。
  - 被调接口: 配置被调的接口名,表明该熔断规则只对调用某个接口的请求生效,可选。接口名支持多种匹配条件。
- 错误判断条件:可配置多个判断条件,满足任意一个条件的请求会被标识为错误请求。支持返回码和时延2种判断方式。
- 熔断触发条件:可配置多个触发条件,满足任意一个条件即会触发熔断,资源会进入熔断状态。支持连续错误数和错误率2种触发条件。
  - 连续错误数:统计调用该服务/接口的请求连续错误数,达到阈值即触发熔断。
  - 错误率:统计在周期内调用该服务/接口的请求的错误率,达到阈值即触发熔断。同时为避免少流量下的放大效应,可配置错误率统计的起始请求阈 值,请求数超过阈值才进行熔断判断。
- 熔断恢复:资源触发熔断后,会熔断请求调度到该资源一段时间。随后系统会对资源进行恢复尝试,当满足一定次数的连续成功请求数后,资源会恢复正常 状态。用户可配置资源熔断时长(单位秒),以及连续成功请求数。
- 可选择开启主动探测。开启后,主调方会根据被调服务/接口所关联的探测规则,向被调方发起探测,探测结果会与业务调用合并,作为熔断触发或恢复的 依据之一。开启主动探测后需配置主动探测规则。

#### 步骤2:设置主动探测规则(可选)

- 1. 登录 TSF 控制台。
- 2. 在左侧导航栏,单击**服务治理中心 > 流量管理**,进入流量管理页面。
- 3. 在页面上方选择熔断降级页签,选择主动探测 Tab。
- 4. 单击新建主动探测规则。

规则名称 *	Test
	最长64个字符
描述	
命名空间*	default
服务名称 🔹	QuickstartCalleeService
接口名称	/ 全匹配 ▼
周期	30 秒
超时时间	60
端口	0
协议 🛈	HTTP TCP UDP
方法	GET -
Url	请输入url 以/开头
Headers	+
提交	取消

- 规则名称:必选。规则名,需全局唯一。
- 描述: 可选。规则的描述信息,用于补充规则的说明。
- 命名空间:必选。被调服务的命名空间。
- 服务名称:必选。被调服务的名称。
- 接口名称:可选。被调接口的名称,支持多种匹配条件。
- 周期:可选。探测周期,多久执行一次探测,默认30秒。
- 超时时间:可选,探测最大超时时间,超时未响应则以最大超时时间作为熔断依据。默认60秒。
- 端口:可选,探测端口,默认为服务实例端口。
- 协议:使用什么协议进行探测,会影响接下来的配置,支持 HTTP、TCP、UDP 3种协议。
  - HTTP 协议配置:



- 方法:HTTP 方法,默认 Get。
- Url:探测 Url,默认/。
- Headers:发送探测包所需的消息头,默认空。
- TCP 协议配置: TCP 默认会采用 tcp connect 的方式进行探测,用户也可以配置基于报文的探测手段。
  - Send: 配置所需发送的TCP二进制报文,格式为0x开头的十六进制字符串,例如: 0x12ab。
  - Receive: 配置所接收的TCP二进制报文,可配置多个,不配置则不校验应答。
- UDP 协议配置:UDP 没有连接检测的方式,只能通过基于报文的探测手段。
  - Send: 配置所需发送的 UDP 二进制报文,格式为0x开头的十六进制字符串,例如: 0x12ab。
  - Receive: 配置所接收的 UDP 二进制报文,可配置多个。

## 步骤3: 客户端接入熔断能力

根据您对应的接入方式查看开发文档下**熔断降级**相关的篇章,以使用 Spring Cloud 接入为例,可参见文档 Spring Cloud 应用开发 > 服务熔断。



# 实例熔断

最近更新时间: 2025-04-17 16:00:54

# 功能说明

实例级熔断规则用于针对特定分组下多个服务实例或者单个服务实例进行熔断。

## 使用说明

要实现服务熔断需要完成两部分操作:

- 步骤1: 在控制台设置熔断规则、主动探测规则。
- •步骤2:客户端接入熔断能力。

详细操作指引可参见下文。

## 步骤1:设置熔断规则

- 1. 登录 TSF 控制台。
- 2. 在左侧导航栏,选择 Polaris (北极星),在实例详情页选择目标引擎实例。

3. 单击左侧导航栏**熔断降级**,进入熔断规则展示页面,单击**新建熔断规则。** 

则名称•	Test
	最长64个字符
苗述	
优先级 🛈	- +
匹配条件	主調羅务 ▶目标服务
	主调服务 被调服务
	命名空间・         echo         v         命名空间・         语选择命名空间         v
	服务名称 · EchoServerGolangTest 服务名称 ·
断粒度	服务 实例 接口
	<
	<b>编新触发条件</b> 满足以下任意条件可触发缩断
	错误率 → >= 50 % 统计周期 30 秒 最小请求数 5 个 +
	+ 添加焊新領總
浙恢复策略	<b>编新教复</b> 进入场断状态后,通过设置超时探测或者主动探测规则,满足条件后即可结束塌断状态教复业务请求。否则重新回到编断状态
	增新时长 60 秒
	主动探测 【 开在主动探索线,客户端将仓根能愈乱置的探测规则时目标被调度务进行探测:主动探测请求与业务调用合并列断培断依疑 (如未匹配到探测规则,则 不会生物): 半开启主动理想她 全切地接收生调用到解迷踪影响

界面各字段含义如下:

基础信息:

- •规则名称:规则名,需全局唯一,必填,不超过64字符。
- 描述:规则的描述信息,用于补充规则的说明,非必填。
- 优先级: 数字越小,优先级越大。
- 匹配条件:匹配条件主要用于决定熔断规则的适用范围,客户端根据匹配条件来过滤本地调用所适用的熔断规则。
  - 主调服务: 配置作为主调方的服务名和命名空间,可选。不配置则默认对所有的主调生效。
  - 被调服务: 配置被调的服务名和命名空间,可选。不配置则默认对所有的被调生效。
  - 另外,熔断规则中所填的服务名,可以是任意服务名,不一定需要存在于北极星注册中心。
- 熔断粒度:选择**实例**。



#### 熔断配置:

#### 熔断策略:

- 错误判断条件:可配置多个判断条件,满足任意一个条件的请求会被标识为错误请求。支持返回码和时延2种判断方式。
- 熔断触发条件:可配置多个触发条件,满足任意一个条件即会触发熔断,资源会进入熔断状态。支持连续错误数和错误率2种触发条件。
- 连续错误数:统计调用该实例的请求连续错误数,达到阈值即触发熔断。
- 错误率:统计在周期内调用该v的请求的错误率,达到阈值即触发熔断。同时为避免少流量下的放大效应,可配置错误率统计的起始请求阈值,请求数 超过阈值才进行熔断判断。
- 熔断恢复策略:
  - 熔断恢复:资源触发熔断后,会熔断请求调度到该资源一段时间。随后系统会对资源进行恢复尝试,当满足一定次数的连续成功请求数后,资源会恢复 正常状态。用户可配置资源熔断时长(单位秒),以及连续成功请求数。
  - **主动探测**:可选择开启主动探测。开启后,主调方会根据被调服务/接口所关联的探测规则,向被调方发起探测,探测结果会与业务调用合并,作为熔 断触发或恢复的依据之一。
  - **是否开启**:选择是否开启该规则。

### 步骤2:客户端接入熔断能力

根据您对应的接入方式查看开发文档下**熔断降级**相关的篇章,以使用 Spring Cloud 接入为例,可参见文档 Spring Cloud 应用开发 > 服务熔断 。



# 服务限流

最近更新时间: 2025-06-11 15:55:02

## 场景说明

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,服务需要对请求做限流,启动流 控保护机制。

限流的原理是监控服务流量的 QPS 指标,当达到指定的阈值时进行流量控制,避免被瞬时高峰流量冲垮,从而确保服务的高可用。限流模式有两种:

- 单机限流:通过统计单机 QPS 指标,当达到规则指定阈值时对流量进行限制,保障服务实例不被瞬时流量给冲垮。
- 分布式限流:通过统计全局 QPS 指标,当达到规则指定阈值时对流量进行限制,保障服务实例不被瞬时流量给冲垮。

# 操作步骤

- 1. 登录 TSF 控制台。
- 2. 在左侧导航栏,选择 Polaris (北极星),在实例详情页选择目标引擎实例。
- 3. 单击左侧导航栏 访问限流,进入熔断规则展示页面,单击新建限流规则,在弹窗中配置好限流规则后,单击提交。

#### 单机限流

参数	是否必填	说明
接口路径	否	指定限流规则的接口过滤参数,接口名可对应方法名、HTTP、URL 等信息,不填写代表不过滤。 <ul> <li>接口名称:规则生效所对应的接口名,用于匹配客户端传入的 method 参数,默认为空(全部)。</li> <li>匹配方式:接口字段的匹配方式,支持全匹配、不等于、包含、不包含、正则表达式五种匹配模式。</li> </ul>
请求匹配规则	否	<ul> <li>指定限流规则的请求参数匹配条件,不填代表不过滤,支持以下四种参数类型:</li> <li>自定义参数:自定义 KEY 和 VALUE,具体的请求参数值可通过 SDK 进行传入。</li> <li>请求终认 (HEADER):针对协议请求参数(http header/grpc header)进行过滤。</li> <li>请求参数(QUERY):针对协议请求参数(http query)进行过滤。</li> <li>方法(METHOD):针对协议的 METHOD (http method/grpc method)进行过滤。</li> <li>主调服务:针对微服务调用场景下,主调方的服务名进行过滤。</li> <li>主调 IP:针对主调方机器的 IP 地址进行过滤。</li> <li>主调字例元数据:通过主调实例标签进行过滤。</li> <li>每种类型参数值支持以下几种值匹配模式:</li> <li>全匹配:全值匹配,传入的值与配置的值相同才匹配通过。</li> <li>正则表达式:用户配置正则表达式,通过正则表达式对传入的值进行匹配,正则表达式支持 Google RE2 标准。</li> <li>不等于:取反匹配,传入的值与所配置的值不相等才算匹配成功。</li> <li>包含:多字符串取 OR 匹配,传入的值只要匹配到其中一个字符串,就算匹配成功。字符串之间使用逗号进行分割。值格式为 value1,value2,value3,匹配到其中一个就算成功。</li> <li>不包含:多字符串取反匹配,传入的值必须都没有出现在所配置的字符串列表中,才算匹配通过。值格式为 value1,value2,value3,全部不等于才算成功。</li> </ul>
限流指标	是	<ol> <li>限流指标:请求数 指定统计周期内的统计阈值,达到阈值则进行限流。可以配置多个限流阈值,多个限流阈值可同时生效,任 意触发了一个即可限流。         <ul> <li>统计窗口时长:限流阈值的统计时长,单位秒,默认为1秒。</li> <li>请求数阈值:达到限流条件的请求数阈值。默认为1。</li> </ul> </li> <li>限流指标:并发数(仅支持单机模式) 指定并发数阈值,达到阈值则进行限流。</li> </ol>
合并计算阈值	否	当限流指标为请求数时,如果目标请求匹配到多个接口及参数,则将匹配到的所有请求汇合,合并计算阈值。



限流效果	是	<ul><li> 快速失败:直接拒绝。</li><li> 匀速排队:根据设置的最大排队时长依次通过。</li></ul>
失败处理策略	否	仅分布式限流需要选择。 • 退化至单击限流;当出现通信失败或者 Token Server 不可用时,限流方案退化到单机限流的模式。 • 直接通过:当出现通信失败或者Token Server不可用时,不做额外处理,直接通过。
失败后返回 Http 脚本	否	填写普通文本或 JSON 格式的 http 返回值。

#### 分布式限流

参数	是否必填	说明
接口名称	否	指定限流规则的接口过滤参数,接口名可对应方法名、HTTP U等信息,不填写代表不过滤。 <ul> <li>接口:规则生效所对应的接口名,用于匹配客户端传入的 method 参数,默认为空(全部)。</li> <li>匹配方式:接口字段的匹配方式,支持全匹配、不等于、包含、不包含、正则表达式五种匹配模式。</li> </ul>
请求匹配规 则	否	指定限流规则的请求参数匹配条件,不填代表不过滤,支持以下四种参数类型: • 自定义参数:自定义KEY和VALUE,具体的请求参数值可通过SDK进行传入。 • 请求头(HEADER):针对协议消息头(http header/grpc header)进行过滤。 • 请求参数(QUERY):针对协议请求参数(http query)进行过滤。 • 方法(METHOD):针对协议的METHOD(http method/grpc method)进行过滤。 • 主调服务:针对微服务调用场景下,主调方的服务名进行过滤。 • 主调 IP:针对主调方机器的IP地址进行过滤。 每种类型参数值支持以下几种值匹配模式: • 全匹配:全值匹配,传入的值与配置的值相同才匹配通过。 • 正则表达式:用户配置正则表达式,通过正则表达式对传入的值进行匹配,正则表达式支持 Google RE2标 准。 • 不等于:取反匹配,传入的值与所配置的值不相等才算匹配成功。 • 包含:多字符串取OR匹配,传入的值只要匹配到其中一个字符串,就算匹配成功。字符串之间使用逗号进行 分割。值格式为'value1,value2,value3',匹配到其中一个就算成功。 • 不包含:多字符串取反匹配,传入的值必须都没有出现在所配置的字符串列表中,才算匹配通过。值格式 为'value1,value2,value3',全部不等于才算成功。
限流阈值	是	指定统计周期内的统计阈值,达到阈值则进行限流。可以配置多个限流阈值,多个限流阈值可同时生效,任意触 发了一个即可限流。 • 统计窗口时长:限流阈值的统计时长,单位秒,默认为1秒。 • 请求数阈值:达到限流条件的请求数阈值。默认为1。
合并计算阈 值	否	如果目标请求匹配到多个接口及参数,则将匹配到的所有请求汇合,合并计算阈值,具体规则查看
失败处理策 略	是	分布式限流需要依赖token server,如果出现token server不可访问,则客户端可以根据配置的规则进行降级,保证用户请求最大限度不受影响。 • 退化成单机限流:默认策略,直接退化成单机计算配额的方式进行限流,单机配额=(全局配额/节点数)。 • 直接通过:不执行限流,所有请求都直接放通。
是否启用	否	启用后,限流规则将立即生效。

# 使用示例

针对服务进行限流



在 RateLimitServiceJava 服务下新建限流规则,指定 QPS 为10,限流效果选择直接拒绝。

限流规则名称•	test-limit-rule
	最长54个字符
限流类型★	单机限流 分布式限流
限流规则详情	目标服务 您可以对目标服务的斯德接口设置限此规则,当该接口做调用时,符合匹配规则的请求,则会触发现此规则
	화名空间• default v
	服务名称• RateLimitCalleeService
	援口名称 请输入接口名称,默认全话 <b>全匹配 ▼</b>
	请求匹配规则 + 透加
	限法规则
	<b>限造条件</b> 满足以下任意条件即可触发度流
	限流阈值 统计窗口时长 请求数词值
	1 秒 🕶 10 次
	+ 淡如 翻除所有
	会并计算调查 👥 如果目标源水匹配到多个银口及争款,则将匹配到的所有请求汇合,合并计算调查,具体规则查看
	网络方案 满足网络触发条件后的处理方案
	展流效果 快速失效 均速转入
是否启用	
提交	<b>取消</b>

# 针对接口 + 标签进行细粒度限流

在 RateLimitServiceJava 服务下新建限流规则,指定 QPS 为10,方法名为/echo,针对 HTTP 请求中的 HEADER user=foo 进行限流,限流效果 选择直接拒绝。

限流类型•	单机限流 分布式限流
限流规则详情	
	目标服务。您可以对目标服务的指定接口设置限流规则。当该接口被调用时,符合匹配规则的请求,则会触发限流规则
	승名空间・ default v
	能务名称• RateLimitServiceJava
	接口名称 /echo 全匹配 v
	请求近配规则① 请求失(HEADER) v user 全匹配 v foo X
	+ 添加 删除所有
	限流规则
	<b>限流条件</b> 满足以下任意条件即可触发跟流
	限流阈值 統计窗口时长 请求数词值
	1 秒 v 10 X
	十添加 删除所有
	合并计算简值 💦 如果目标请求匹配到多个接口及参数,则将匹配到的所有请求汇合,合并计算简值,具体规则重看
	<b>限流方案</b> 满足限流触发条件后的处理方案
	限流效果 快速失救 匀道排队

# 针对接口 + 标签进行热点参数限流

热点参数限流规则配置需要满足2个条件,否则无法生效:



- 存在非全匹配规则: "接口名称"、"请求匹配规则"配置中至少存在一条规则是非全匹配规则,例如"不等于";如果全是全匹配规则,那该限流只是普通限流。
- 不启用"合并计算阈值"。

热点参数限流是一种针对高并发请求中某些热点参数的限流策略;如上述规则的限流效果图如下:



在 RateLimitServiceJava 服务下新建限流规则,指定 QPS 为10,方法名为/echo,针对 HTTP 请求中的 header user 的每一个参数值进行单独限 流,限流效果选择直接拒绝。

限流现则名称。	热点现间接流
1	墨长6个字符
限流类型・	<b>単机限点</b> 分布式限済
限流规则详情	目标服务 它可以对目标服务的指定使口或重用地规则,当该使口被调用时,符合否認规则的事实,则会被规则地取
	会会の時・ default ・
	服务名称• RateLimiServiceJava
	#口名称 echo 全正祝 *
	请求正配规则 ① 请求关(HEADER) ▼ User 不等于 ▼ test X
	十语加 舰除所有
	限法规则
	<b>限油条件</b> 派起以下在意条件即可触发用流
	限治周痛 杭计會口時长 请求教词籍
	1 B + 10 X
	十次20 期時所有
	合并计算网编 💽 如果自标清实还配到多个独口及参数,则将匹配到的所有需求汇合,合并计算网编,具体规则查看
	<b>限点方法</b> 满亚报师被送外件后的处理方案
	形地如果 (快速完效 )均逾接入

# 使用匀速排队

在 RateLimitServiceJava 服务下新建限流规则,指定 QPS 为10,限流效果选择匀速排队。



PRUIDHUUI 446* * <b>test-imit-fuie</b> 最长64个字符	
12 COA 1 2 12	
限流类型 · 单机限流 分布式限流	
限流规则详循 目标服务 您可以对目标服务的指定接口设置限流规则。当该接口被调用时,符合匹配规则的请求,则会就发现流规则	
命名空间• default v	
服务名称• RateLimitCalleeService	
接口名称 请输入接口名称,默认全选 <b>全匹配 v</b>	
请求匹配规则 +添加	
<b>限流规则</b> <b>限流条件</b> 满足以下任备条件如可触发限流	
段流過值 统计窗口时长 请求数间值	
1 秒 10 次	
+添加 删除所有	
合并计算调查 👥 如果目标请求匹配到多个按口及参数,则将匹配到的所有请求汇合,合并计算调查,具体规则查看	
<b>開造方案</b> 满足限流触发条件后的处理方案	
現流效果 <b>快速失敗 勾速排队</b>	
最大编队时长 <b>1 秒</b>	
是百足用	
提文取消	



# 无损上下线 无损上线

最近更新时间: 2025-06-11 15:55:02

# 功能概述

对于任何一个线上应用来说,发布、扩容、缩容、重启等操作不可避免。Java 应用经常会延迟加载,启动后异步加载一些资源,例如:服务需要从文件存储 COS 获取数据或者文件,待数据或者文件拉取完成后才能对外提供服务。如果在应用启动后直接注册服务,会导致服务实际没有就绪而调用失败。为了确保 服务无损上线,需要实例成功注册并且就绪后,先小流量预热,再放流量进来。具体功能包含:

- 服务延迟注册
- 服务注册就绪检查(提供服务注册是否成功的接口,以便打通 K8s 就绪检查)
- 服务小流量预热

	应用初始化	 服务延迟注册	<b> </b> →	服务注册就绪检查	 服务小流量预热	 正式流量进入	
_							

#### 操作步骤

要实现服务路由需要完成两部分操作:

- 在腾讯云控制台上,设置无损上线的规则。
- 客户端接入,并引入无损上线的插件。北极星提供两种接入方式:
  - SpringCloud Tencent
  - Java Agent

下文将详细讲解使用步骤。

## 腾讯云控制台操作步骤

- 1. 登录 TSF 控制台。
- 2. 在左侧导航栏,选择 Polaris (北极星),在实例详情页选择目标引擎实例。
- 3. 单击左侧导航栏访问限流,进入熔断规则展示页面,单击新建限流规则,在弹窗中配置好限流规则后,单击提交。
- 4. 在左侧导航栏,**服务治理中心**下,单击**服务管理**,选择目标北极星引擎和目标服务,进入**服务详情页**。
- 5. 单击无损上下线的 TAB。
- 6. 在无损上线的配置框中,点击设置按钮。

版本号 - 容错保护 关闭 则 限流规则 权限管理	服务标签(0个) - 健康/总卖明数 0/0 健康时间 2024-09- 10 11:31:20 修改时间 2024-09- 10 11:31:20	描述 -
容错保护 关闭	创建时间 2024-09- 修改时间 2024-09- 10 11:31:20 10 11:31:20	
则 限流规则 权限管理		
已关闭	<b>无损下线</b> 已关闭	设置
启动要晚,为了确保服务实例成功注册 服务延迟注册、服务预热、服务就绪枪	在用在滾动发布或者下线过程中,被调方服务实例向注册中心发起反注 中,存在时间gap,此时依然有可能调用到已经下线的实例,从而调用9 口/offine,支持与K8s生命周期prestop接口接合使用。	<sup>册</sup> ,主调方从北极星更新ip过程 败。北极星提供无损下线接
启服	动要晚,为了确保服务实例成功注册 务延迟注册、服务预热、服务就绪检	动要教。为了确保服务实例成功注册 房延迟注册、服务预热、服务就绪给 中,存在时间gap。此时依然有可能调用到已经下线的变例,从而调用失 口[offine、支持与K8s生命周期prestop接口接合使用。

7. 按业务诉求,开启配置开关。





功能	描述	字段
	北极星提供延迟注册的能力,待应用完全启动后, 再进行服务注册。北极星支持两种延迟注册方式:	时长延迟: • 延迟注册时间:延迟注册的时间,支持0-3600秒,默认延迟注册时 间:30秒
延迟注册	<ul> <li>的长远远: 通过设置远达注册的间,可任应用住充分初始化后再注册到注册中心对外提供服务。</li> <li>探测延迟: 探测应用对外暴露的检查接口,接口探测成功,返回200,确认应用可对外服务后再注册到注册中心。</li> </ul>	探测延迟: • 探测协议:支持HTTP • 探测方法:支持PUT、GET、POST、DELETE、全部。 • 探测接口:填写业务暴露健康检查接口,接口探测成功后,服务注 册。例如:/health
服务预热	北极星提供服务预热的能力,小流量验证后无误 后,再放入正常流量。	<ul> <li>预热时长:默认120秒,支持 0~86400 秒(24小时)。</li> <li>预热终止保护百分比:当超过一定百分比的节点处于预热阶段,则预 热终止,避免流量聚集在少量节点导致过载。默认50%</li> <li>预热曲线值:默认为1,支持 1~5 之间的整数值,流量权重会根据配 置的预热曲线值呈指数型增长。数值越大,预热曲线梯度越大。</li> <li>服务实例权重计算规则:</li> <li>【(当前时间 - 服务实例注册时间)/预热时长]<sup>预热曲线值</sup></li> </ul>
就绪检查	就绪检查开关开启后,提供接口判断注册是否完成。和 K8s 就绪检查功能配合使用,当应用注册完 成返回200状态码,帮助 K8s 判定应用已就绪;未 完成注册返回500,帮助 K8s 判定应用未就绪。 接口:/readiness 端口: 28080	

#### 8. 可观测性。

可以实时查看服务无损上线的状态、服务请求等,确保每一步操作符合预期。



P均请求延时 (ms	<b>;)</b> 平均值 ~				[] …	总请求数 (Coun	t) 总值 ~				[] ·
100						100					
80						80					
60		暂无数	据			60		暂无数排	居		
40		E / CAA	270-4			40			71-0		
20						20					
0 15:04:43	15:05:43	15:06:43	15:07:43	15:08:43		0 15:04:43	15:05:43	15:06:43	15:07:43	15:08:43	

# 应用客户端操作步骤

北极星提供两种接入方式:

- 1. SpringCloud Tencent
- 2. Java Agent

SpringCloud Tencent	
<ol> <li>服务通过 Spring Cloud Tencent 接入北极星服务注册与发现能力。具体步骤请参见: Spring Cloud Tencent 接入北极星。</li> <li>引入 spring cloud tencent 无损上下线插件依赖。</li> <li>在 pom.xml 中添加依赖:</li> </ol>	
<dependencies> <dependency> <groupid>com.tencent.cloud</groupid> <artifactid>spring-cloud-tencent-lossless-plugin</artifactid> </dependency> </dependencies>	

Java Agent

服务通过 Java Agent 接入北极星。具体步骤请参见: Spring Cloud 使用 Java Agent。



# 无损下线

最近更新时间: 2025-06-11 15:55:02

# 功能概述

应用在滚动发布或者下线过程中,被调方服务实例向注册中心发起反注册,主调方从注册中心更新 IP 的过程中,存在时间 gap,导致依然有可能调用到已经 下线的实例,从而调用失败。

北极星提供服务无损下线的接口:/offline,端口:28080,结合 K8s 的生命周期,实现服务无损下线。整体流程如下:



# 操作步骤

要实现服务路由需要完成两部分操作:

- 在腾讯云控制台上,打开无损下线的开关。
- 客户端接入,并引入无损下线的插件。北极星提供两种接入方式:
  - SpringCloud Tencent
  - Java Agent

下文将详细讲解使用步骤。

### 控制台操作步骤

- 1. 登录 TSF 控制台。
- 2. 在左侧导航栏,**服务治理中心**下,单击**服务管理**,选择目标北极星引擎和目标服务,进入**服务详情页**。
- 3. 单击无损上下线的 TAB。
- 4. 在无损下线的配置框中,点击**设置**按钮。

』本信息	L										编
i务名 to	test	命名空间	default	版本号	-	服务标签(0个)	-	健康/总实例数	0/0	描述 -	
(见性 <b>(</b> ) り	仅当前命名空间可 见	就近路由	关闭	容错保护	关闭	创建时间	2024-09- 10 11:31:20	修改时间	2024-09- 10 11:31:20		
服务实例	例  无损上下线	路由规则	」 熔断规则	限流规则	权限管理						
服务实例	例 <b>无损上下线</b> 上线 <sub>延迟注册:已关闭</sub>	路由规则 服务预热:已多	」 熔断规则 长闭 就绪检查:已关闭	限流规则	权限管理	无损下线	已关闭				设置
服务实例 无损上 Java应/ <sup>并且就维</sup> 查。查看	人類上下线           延迟注册:已关闭           如月经常会延迟加载,应用           (端后,再放流量进来,需)           者支持版本	路由规则 服务预热:已封 正常对外提供E 要无损上线的自	」 熔断规则 统闭 就绪检查:已关闭 服务的时间比应用启动理 能力。北极星支持服务硕	限流规则 ] {晚,为了确保 {迟注册、服务	权限管理 设置 服务实例成功注册 预热、服务就绪检	无损下线 应用在滚动劝 中,存在时间 口/offine,3	已关闭 c在或者下线过程中 Jgap,此时依然有 c持与K8s生命周期	,被调方服务实例 可能调用到已经下约 prestop接口接合使	向注册中心发起反泛 线的实例,从而调用 E用。	主册,主调方从北极星 3失败。北极星提供无f	设置 更新ip过程 横下线接
服务实例 无损上 Java应 并且就能 查,查得 译	例 无损上下线 上线 延迟注册:已关闭 四段常会延迟加载,应用 线后,再放流量进来,需 着支持版本	路由规则 服务预热:已引 正常对外提供E 要无损上线的前	」 熔新规则 統備 就绪检查:已关闭 服务的时间比应用启动到 能力。北极星支持服务颈	<b>限流规则</b> 1 長晩,为了确保 5迟注册、服务	<b>权限管理</b> 设置 服务实例成功注册 预热、服务就绪检	<b>无损下线</b> 应用在滚动 中,存在时间 口/offine,3	已关闭 <sup>读</sup> 布或者下线过程中 Jīgap,此时依然有 持与K8s生命周期	,被调方服务实例 可能调用到已经下约 prestop接口接合使	向注册中心发起反泛 线的实例,从而调用 。	主册,主调方从北极星 月失败。北极星提供无扩	<mark>设置</mark> 更新ip过程 惯下线接

5. 按业务诉求,开启配置开关。



;	无损下线		×
Ŧ	无损下线	【】 提供无损下线接□/offline。可和K8sprestop接口结合使用。 查看 K8s prestop 配置示例	
		提交关闭	

6. 在容器服务 TKE 等 K8s 应用部署平台中配置 preStop 生命周期检查。

preStop 配置检查指令: curl -X PUT http://localhost:28080/offline && sleep 20

# 应用客户端操作步骤

北极星提供两种接入方式:

- SpringCloud Tencent
- Java Agent

SpringCloud Tencent

- 1. 服务通过 Spring Cloud Tencent 接入北极星服务注册与发现能力。具体步骤请参见: Spring Cloud Tencent 接入北极星。
- 2. 引入 spring cloud tencent 无损上下线插件依赖。

在 pom.xml 中添加依赖:

<dependencies> <dependency> <groupId>com.tencent.cloud</groupId> <artifactId>spring-cloud-tencent-lossless-plugin</artifactId> </dependency> </dependencies>

Java Agent

服务通过 Java Agent 接入北极星。具体步骤请参见: Spring Cloud 使用 Java Agent。

# 全链路灰度 联动云原生网关实现全链路灰度

最近更新时间: 2025-04-17 16:00:53

# 使用场景

腾讯云

应用发布在软件开发过程中非常常见与频繁,但是变更过程容易引发故障。为了降低每次发布带来的变更风险,可以通过灰度发布策略来解决这一问题。灰度 发布是一种软件发布策略,它允许您在生产环境中渐进式部署应用,新版本只对部分用户可见,在问题出现时尽量减少影响。在微服务体系架构中,服务间的 依赖关系错综复杂,有时单个服务发版依赖多个服务同时运行联动,对这个新版本服务的上下游进行分组验证,这是微服务架构中特有的全链路灰度发布场 景。

TSF-Polaris(北极星)提供的全链路灰度能力,在不修改业务代码的情况下,可视化配置灰度规则,就可以构建出从网关到后端服务的端到端的全链路流 量控制。泳道可以将应用灰度发布的版本隔离成一个独立的运行环境。通过设置泳道规则,可以将满足规则的请求流量路由到目标版本的应用。如下图所示, 我们划分出一个或多个灰度环境,针对线上灰度版本1的所有服务,我们希望都能配置一定规则。当满足一定规则后,所有请求都会流入灰度环境1中。



# 名词解释

- 泳道组:一组泳道、服务、入口网关的集合,通常一个业务系统下的应用放在同一个泳道组中。
- 泳道:一组具有相同特性标签的服务实例节点的集合,是灰度发布规则的目的地。
- 基线: 没有添加至任何泳道的服务节点为基线版本节点。
- 实例标签: TSF 北极星提供特定的泳道标签,标签 key 为 "lane", value 值为您自定义的「泳道标签值」。打上该标签的实例会自动添加至目标泳道中。

# 使用限制

大支	的组件	版本限制
北极星		<ul> <li>● 产品版本:企业版</li> <li>● 引擎版本 &gt;= v 1.18.0.0</li> </ul>
	北极星 SDK	版本 >= v 1.15.3
微服务应用	SpringCloud Tencent	<ul> <li>SpringCloud Tencent 2023版本: &gt;= 1.13.1-2023.0.0</li> <li>SpringCloud Tencent Hoxton 版本: &gt;= 1.14.0-Hoxton.SR12-RC2</li> </ul>
网关	云原生网关	版本无限制



	SpringCloud Gateway	● SpringCloud Tencent 2023版本:>= 1.13.1-2023.0.0 ● SpringCloud Tencent Hoxton 版本:>= 1.14.0-Hoxton.SR12-RC2
	Ckafka	建设中
消息队列	TDMQ-RocketMQ	建设中
	TDMQ-Pulsar	建设中

# 前提条件

- 已 创建北极星引擎实例。
- 我们为您提供了全链路灰度 Demo ,请参见 Demo 。
- 全链路灰度能力需要您在客户端接入北极星服务注册发现、服务路由能力。
  - SpringCloud Tencent 应用接入,请参见 接入全链路灰度 。
  - Java Agent 接入请参见 通过 Java Agent 接入北极星 。

#### 配置流程总览

#### 步骤1: 搭建业务系统基线版本

- (1) 部署业务应用基线版本。
- (2) 创建 云原生网关。
- (3)关联北极星注册中心,配置云原生网关服务路由。

#### 步骤2:搭建业务系统灰度环境

- (1) 部署业务应用灰度版本,并打上灰度标签。
- (2)创建泳道组,配置业务系统的入口网关、业务应用。
- (3) 创建泳道并配置灰度规则,符合泳道规则的请求将被转发至泳道内。

## 步骤1: 搭建业务系统基线版

#### (1) 部署业务应用基线版本。

根据您自身的业务,准备好业务部署的资源,通过 容器化部署 、 虚拟机部署 选择其中一种方式部署业务系统后端应用的基线版本。

○ 容器化部署:

- 创建 TKE 容器集群,请参见 创建 TKE 集群 。
- 在所选集群中,创建工作负载,完成业务应用部署,请参见 创建工作负载 。
- 虚拟机部署:
  - 创建 CVM 虚拟机,请参见 创建 CVM 虚拟机 。
  - 部署业务应用。

### (2) 创建云原生网关。

创建云原生网关,参见 创建云原生网关。

#### (3)关联北极星注册中心,并配置服务和路由信息。

服务来源关联北极星注册中心(PolarisMesh),并配置网关服务和路由信息,参见使用云原生 API 网关访问北极星服务。

#### 步骤2: 搭建业务系统灰度环境

如果业务应用要发布新版本,可以通过全链路灰度来同时验证不同应用的灰度版本,以确保发布上线的稳定性。

#### (1) 部署业务应用灰度版本,并打上灰度标签。

根据您自身的业务,准备好业务部署的资源,通过 容器化部署 、 虚拟机部署 选择其中一种方式部署业务系统后端应用的**灰度版本**。

#### 容器化部署

1. 创建 TKE 容器集群,请参见 创建 TKE 集群。



- 2. 在所选集群中,**创建工作负载**,请参见 创建工作负载。
- 3. **灰度打标**:北极星提供灰度泳道标签,标签key为"lane",value 值为您自定义的**版本标签值**。您可以在配置工作负载的过程中,设置启动参数 或者 设置环境变量 (二选一)来配置灰度标签:

## 方案一:设置应用启动参数

Java -Dspring.cloud.tencent.metadata.content.lane=\$(lane\_value) -jar demo.jar

### 方案二: 设置环境变量

设置环境变量:

## () 说明:

```
$(lane_value) 替换为您自定义的版本(泳道)标签,例如: grey、v2等。
```

4. 完成灰度版本部署。

#### 虚拟机部署

- 1. 创建 CVM 虚拟机,请参见 创建 CVM 虚拟机。
- 次度打标:北极星提供灰度泳道标签,标签 key为 "lane",value 值为您自定义的「版本标签值」。您可以在配置工作负载的过程中, 设置启动参数 或者 设置环境变量(二选一)来配置灰度标签:

```
方案一:设置应用启动参数
```

Java -jar -Dspring.cloud.tencent.metadata.content.lane=\$(lane\_value) demo.jar

#### 方案二: 设置环境变量

**设置环境变量:** SCT\_METADATA\_CONTENT\_lane=\$(lane\_value

#### () 说明:

```
$(lane_value)替换为您自定义的版本(泳道)标签,例如:v2。
```

3. 完成灰度版本部署。

# (2) 创建泳道组,配置业务系统的入口网关、业务应用。

- 1. 侧边栏单击全链路灰度 > 新建泳道组,进入新建泳道组的页面。
- 2. 填写泳道组相关配置,配置入口网关、业务应用。



	As the Wards - the second P	MR Hullow Andrew	~			
	允许数字、英文字均	:、.、-、_、, 限制 64 个字符	3			
2 配置泳道组	λп					
入口网关类型	✓ TSE 云原生网	×				
	法择國关定例		(广州)		× Ó	
		如无合适网关实例,前往新	建。		-	
		请确保云原生网关已关联服	务治理中心(北极县	l),操作详见 <mark>文档链接</mark>	ŧ 🖸	
	选择入口服务	caller			-	
	添加入口					
	微服务网关Sp	ringCloud Gateway				
	微服旁应用					
<ol> <li>3 泳道组服务</li> <li>泳道组服务</li> </ol>	<b>配置</b> 未注册的服务,直接	添加至泳道组				
<ol> <li>泳道组服务</li> <li>泳道组服务</li> </ol>	<b>配置</b> 未注册的服务,直接 选择泳道组服务 default	添加至泳道組	8 Q	已选择 (6)个	会名亞個	
3 決道組服务 決道組服务	<b>配置</b> 未注册的服务,直接 <b>选择泳道组服务</b> default マロ/服务名	添加至 <b>泳道组</b> 命名空间	0 Q	<b>已选择 (6)个</b> ID/服务名	命名空间	
3 泳道组服务 涂道组服务	<b>配置</b> 未注册的服务,直接 选择泳道组服务 default ☑ □/服务名	<b>添加至冰道组</b> 命名空间	Q	<b>已选择 (6)个</b> ID/服务名	命名空间 default	G
<ol> <li>決道組服务</li> <li>決道組服务</li> </ol>	<b>記置</b> 未注册的服务, 直接 透持沫道組服务 default ♥ 10/服务名	源加至外道组 命名空间 default	© Q	<b>已选择 (6)个</b> ID/服务名	命名空间 default default	C
3 決道組服务 決道組服务	R重 未注册的服务,重排 法持沣連組服务 default マロノ服务名	湯加至沖道館 命名空间 default	00	已遗择 (6)个 10/服务名	命名空间 default default	8
3 決道組服务 決道組服务	R置 未注册的服务,重排 法择泳道組服务 default マロ/服务名	湯加至決道線 命名空间 default	© Q	<b>已透择 (6)个</b> □/服务名	命名空间 default default default	2
<ol> <li>決道組服务</li> <li>決道組服务</li> </ol>	<b>記置</b> 未注册的服务、直接 遠接沐道組服务 default ☑ □0/服务名	湯加至外道線 命名空间 default default	© Q	已遗择 (6)个 ID/服务名	命名空间 default default default	000000000000000000000000000000000000000
<ol> <li>決道組服务</li> <li>決道組服务</li> </ol>	<b>記置</b> 未注册的服务。直接 遠接沐道組服务	湯加至外道線 命名空间 default default		已递择 (6)个 ID/服务名	命名空间 default default default default	0 0 0 0
<ol> <li>決進組織务 決進組織务</li> </ol>	R置 未注册的服务,直接 选择沐道组服务 default ご 10/照务名	時加至外道館 命名空间 default default		已递择 (6)个 ID/服务名	命名空间 default default default default	0 0 0 0
<ol> <li>決進組織务 決進組織务</li> </ol>	R置 未注册的服务。直接 选择沐道组服务	時加至外道報 の名空阿 default default		已送择 (6)↑ ID/服务名	命名空间 default default default default default	0 0 0 0
3 決道相限务 決運相限务	R置 未注册的服务。直接 选择沐道组服务 default ご 10/照务名	時加賀外道館 小名空阿 default default default		B递择 (6)个 ID/服务名	命名空间 default default default default default	0 0 0 0 0

- 泳道组名称:允许数字、英文字母、点、短横线、下划线限制 64 个字符。
- 配置泳道入口: 支持对接云原生网关、微服务网关 SpringCloud Gateway、普通微服务应用。自动同步您在北极星所配置的泳道规则。
  - 云原生网关:选择与北极星在相同地域和 vpc 下的网关实例,并选择业务系统入口服务。入口服务支持多选。
  - 微服务网关 SpringCloud Gateway:选择作为系统入口的 SpringCloud Gateway 网关服务。支持多选。
  - 微服务应用:选择作为系统入口的 SpringCloud Tencent 服务。支持多选。
- 配置泳道组服务:
  - 一般一个泳道组对应一个业务系统,可将业务系统中全部服务全部导入进来。
  - 如果服务暂未注册,也支持手动添加。
- 3. 单击提交按钮,泳道组即被创建,可在泳道组列表中查看。

#### (3) 创建泳道并配置灰度规则,符合泳道规则的请求将被转发至泳道内。

- 1. 侧边栏单击全链路灰度,在目标泳道组右上角单击创建泳道,进新建泳道的页面。
- 2. 填写泳道组相关配置,配置入口网关、业务应用。



100 III T 1010	grav					
	#*** 允许数字、英文字母、-、 . 限制	64 个字符				
描述	全链路灰度					
所属泳道组	janice-test					
泳道标签	gray					
	标签key为"lane",匹配该value值的	的服务实例将被自动添加至泳道。允许	轩数字、英文字母、-、_, 限制 64 个字	Ť		
	ID/服务名		命名空间		泳道标签	
	service-consumer-2023		default		lane:gray	
	service-provider-2023		default		lane:gray	
▶ 高级选项	共 2 奈				请选择 ▼ 条 / 页	₩ ≪ 1 /1页
▶ 高级选项 泳道路由规	共 2 条 <b>刻</b>				请选择 ▼ 条 / 页	iii < 1 /1页
▶ 高級选项 <b>泳道路由規</b> 灰度規則	共 2 张 例	參数名	关系	參数值	请选择 ▼ 条 / 页	× < 1 /1页
▶ 高级选项 泳道路由规 灰度规则	共 2 张 <b>刻</b> · · · · · · · · · · · · · · · · · · ·	参数名 マ canary	关系等于	争数值 マ gray	请选择 ▼ 条 / 页	<u>к с 1</u> /1) Х
▶ 高级选项 泳道路由规 次度规则	共 2 奈 <b>刻</b> 参数位置 	参数名 ▼ canary	关系等于	争致值 v gray	请选择 ▼ 条 / 页	× < 1 /1页
▶ 高级选项 泳道路由规 灰度规则	共 2 奈 別 ●数位置 - - - - - - - - - - - - -	●数名 ▼ canary & (講足以上任意条件)	关系等于	争致值 gray	请选择 ▼ 条 / 页	× « 1 /1)) Х
▶ 高级选项 泳道路由规 灰度规则 规则生效关系 规则匹配策略	共 2 奈 19)  参数位置  · · · · · · · · · · · · · · · · · · ·	参数名           マ         canary           成 (編足以上任意条件)            空配不到対应标签的节点,则跨级造由            空配不到対应标签的节点,则跨级造由	关系 等于 / <b>2基线</b> 节点 1, 2 <u>集国</u> 院在指定 <b>以</b> 道内	參設值 マ gray	请选择 ▼ 条 / 页	× < 1 /1π ×
▶ 高級选项 泳道路由規 灰度规则 規則生效关系 规則匹配領導 是否高用	共 2 奈 第 第 第 第 第 次 、 二 第 次 、 二 二 第 次 、 二 二 第 次 、 二 二 二 二 二 二 二 二 二 二 二 二 二	参数名                 (編足以上任意条件)           2配不到对应标签的节点,则逐回错误	关系 等于 至基线节点 , 流量固定在指定决道内	参数值 マ gray	请选择 ▼ 条 / 页	× < 1 /1π ×

#### • 创建泳道

- 泳道名称: 允许数字、英文字母、短横线、下划线, 限制 64 个字符。
- 描述: 填写泳道的描述和用途。
- 泳道标签:填写灰度发布服务的泳道标签,北极星自动将符合该标签的服务自动添加至泳道。
  - 使用示例:如上图,业务系统整体有10个服务,全部添加至泳道组。此次发布其中2个服务上线(service-consumer-2023, serviceprovider-2023),并配置 lane = grey 的标签。则这两个灰度版本实例自动添加至泳道。

#### • 泳道路由规则

- 灰度规则:
  - 通过配置灰度规则,将符合灰度规则的流量染色并转发至目标泳道内。灰度规则将自动同步至泳道入口(云原生网关、SpringCloud Gateway、入口微服务)。
  - 灰度规则参数支持请求头、请求方法、请求参数、请求 Cookie、请求路径、主调 IP。
- 规则生效关系: 支持 与 (满足全部条件)、或 (满足以上任意条件)。
- 规则匹配策略: 支持宽松匹配与严格匹配策略,请根据您业务实际情况选择。
  - 宽松匹配策略:若在泳道内匹配不到对应标签的节点,则降级路由至基线节点。
  - 严格匹配策略: 若在泳道内匹配不到对应标签的节点,则返回错误,流量固定在指定泳道内。
- 3. 单击提交,即可完成泳道的创建,在泳道组列表中可以查看泳道列表信息。
- 4. 单击 **云原生网关 > 路由管理 > 服务** > 选择入口服务 > **灰度策略**,查看泳道规则已同步至云原生网关。至此已完成全部的规则配置。

÷ 📃	/ 路由管理 / 服务 / caller					
路由管理	<b>灰度策略</b> 限流策略 服务信息					
规则匹置	<b>記顺序</b> <b>全链路灰度规则</b> 暂不支持创建全链路灰度规则,如需创建规则,请前往服务治理中心 I2					
	条件	流量目的地	匹配模式	所属泳道组/泳道	状态	
	满足所有条件: Header <b>参数</b> canary == gray <b>门</b>	服务: caller 标签: lane:gray	宽松匹配策略	-test / gray		
	共 1 条				10 ▼ 条 / 页 🛛 🖣	1 /1页 🕨 🕅



# 配置管理(配置中心) 配置分组



最近更新时间: 2025-04-17 16:00:53

### 操作场景

北极星提供可视化的配置管理界面,您可以在 TSF 控制台查看配置文件的发布情况,并对配置进行编辑修改等。 该任务指导您如何通过 TSF 控制台在某一北极星引擎下进行配置分组管理。

## 前提条件

- 已 创建北极星引擎。
- 已 创建命名空间。

## 操作步骤

#### 新建配置分组

- 1. 登录 TSF 控制台,在左侧菜单栏选择 Polaris (北极星),进入引擎实例列表页。
- 2. 在实例列表页选择目标引擎,点击 配置管理,进入配置分组页面。
- 3. 在左上角单击新建,根据自身业务需求选择相关信息。
  - 命名空间:选择创建好的命名空间。
  - 分组名:填写服务名称,允许数字、英文字母、.、-、\_,限制128个字符。
  - 部门: 可选, 填写业务部门信息。
  - 业务:填写业务信息。
  - 标签: 可选,用于分类管理资源,具体使用方法可参见 标签管理。
  - 备注:可选,填写配置分组描述信息。

冷名空间 *	请选择		*	
分组名★				
βî"]				
络				
示签	标签键	标签值	操作	
		无标签		
	添加标签			
	标签键的长度不能超过12	8字符,标签值的长度不能超过4096个字符		
F注	长度不超过200个字符			
- 高级配置				

4. 单击**提交**,完成新建配置分组。

后续操作:配置分组创建完成后,您需要继续在该配置分组下创建配置文件,参见配置文件。

## 删除配置分组

- 1. 在左侧导航栏选择配置管理 > 配置分组,选择目标配置分组,单击操作列的删除。
- 2. 在二次弹窗页面确认删除。



# 配置文件

最近更新时间: 2025-05-16 15:28:02

# 操作场景

该任务指导您通过 TSF 控制台,如何在某一北极星引擎下进行配置文件管理。

## 前提条件

- 已 创建北极星引擎。
- 已 创建命名空间。

# 操作步骤

### 创建配置文件

- 1. 登录 TSF 控制台,在左侧导航栏点击 Polaris (北极星)进入实例列表详情页。
- 2. 单击目标实例 ID,进入引擎实例信息页,左侧选择配置管理进入配置分组页面。
- 3. 选择目标配置分组并单击进入以下页面。

← asd(default)					📑 操作指南
<b>配置文件</b> 配置版本 发布历史					
新增利新配置					
请输入文件名搜索	Q ddd				查看发布历史 已
🖸 ddd <mark>待发布</mark>	状态	编辑待发布	最后修改人	-	
	最后修改时间	2025-03-12 12:29:04	最后发布人	-	
	最后发布时间	0001-01-01 00:00:00	备注	-	
	标签	-	格式	text	
	发布	編編句現			
	1 1	112231313sdsd			

#### 4. 单击新增创建配置文件。

- 命名空间:目标配置分组所在命名空间。
- 配置分组: 配置分组名。
- 配置文件名: 配置文件的名称; 允许数字、英文字母、-、\_, 限制128个字符。可通过/分隔符创建文件夹,强烈建议文件名带上后缀,如: datasource/master.json。
- 备注:可选,填写配置文件的描述信息。
- 配置标签:可选,配置标签可用于标识配置的用处、特征,格式为 key:value。
- 配置加密:可选,支持 AES 配置加密算法。
- 高级设置:
  - 推送方式: 仅 SDK 读取/仅 Agent 读取 / SDK 和 Agent 同时读取。
  - 当选择 "仅 Agent 读取" 时候:
    - 文件编码方式: UTF-8/gbk。
    - 配置下发路径: 配置下发到服务器的路径。
    - 后置脚本命令: 配置下发到服务器后执行的命令(不需要包含 #! /bin/bash)



建配置文件			
名空间 • defai	ult ~		
置分组• alex-	test v		
置文件名• 允许	数字、英文字母、.、-、_,同	限制128个字符	文件格式: text
可通过。	'分隔符创建文件夹,强烈建议	义文件名带上后缀,如:datasourd	ce/master.json
主 长度	不超过200个字符		
<sup>11</sup> 标签 标签:	名标签	<u>ā</u>	
新增			
iæ 🚺			
i法 AES	~		
级设置			
方式 仅5	DK读取 仅Agent读取	SDK和Agent同时读取	
+编码方式 UT	F-8 gbk		
置下发路径 e.g. /	/etc/nginx/conf.d/		
置脚本命令 1			

5. 单击提交,完成新建配置文件。

## 编辑配置文件

- 1. 在配置文件页面,选择目标配置文件。
- 2. 选择期望编辑的配置文件,并单击**编辑**按钮。
- 3. 对配置文件内容进行编辑之后,单击**保存**。

test(default)					
<b>配置文件</b> 配置版本 发布历史					
新培用新設置					
请输入文件名搜索	Q,	polaris.yaml			查看发布历史 🖸
■ polaris.yaml 待发布		状态 编辑待发布 最后	后修改人	10 2	
		最后修改时间 2024-01-23 14:23:36 最后	后发布人		
		最后发布时间 - 备注	Ξ	-	
		标签 格1 发布 保存 取消	Ť	yaml	
		1 server: 2 port:8081			13

## 发布配置文件

- 1. 在配置文件页面,选择目标配置文件。
- 2. 选择期望发布的配置文件,并点击**发布**。

配置文件 配置版本 发布	历史					
新增期新配置						
请输入文件名搜索	Q	polaris.yaml				查看发布历史 🖸
■ polaris.yaml 待发布		状态	编辑待发布	最后修改人	1 2	
		最后修改时间	2024-01-23 14:25:53	最后发布人	-	
		最后发布时间	-	备注	-	
		标签		格式	yaml	
	[	发布	編辑			
						53
		1 ser 2 ····	ver:  port:8081			

3. 在弹窗中输入配置版本好,并进行二次确认发布动作即可。



发布配置文件	×
💙 填写发布信息 > 2 版本对比	
上一版本(v1.0)	当前版本(v2.0)
1 server: 2 port:8081	1 server: 2+ port:8062
上一步	提交关闭

# 查看发布历史

- 1. 在左侧导航栏单击**配置管理**,选择好地域和实例后,单击**发布历史**页签。
- 2. 在发布历史页面,可以查看配置发布记录,您也可以在右上角通过命名空间和分组名/配置文件名来查询您想要查看的配置发布记录。

配置管理	⑤ 中国大陆 ▼	A	v		(	产品体验,您说	<mark>说了算</mark> 加入用户3	交流群 ♡ 操作指南
配置分组	发布历史							
						全部命名空间 🔻	请选择条件进行过滤	Q Ø
名称	版本	操作类型	状态	命名空间	配置分组 🍸	操作人	发布时	前
polaris.ya	aml v2.0	) 发布	成功	default	test	1	2 2024	-01-23 14:29:06
polaris.ya	ami v1.0	发布	成功	default	test	1	2 2024	-01-23 14:27:23
共 2 条						20 - 条/页	∺ ∢ 1	/1页 ▶ ▶

# 配置回滚

- 1. 在左侧导航栏单击配置管理,选择好地域和实例后,单击目标配置分组进入配置文件页面。
- 2. 在页面上方选择**配置版本**,可以查看历史配置版本。

test(default	)						
記置文件 配置	版本 发布历史						
						请选择条件进行过滤	
名称	版本	发布时间	操作	polaris.yaml v2.0			
polaris.yaml 使用中	v2.0	2024-01-23 14:29:06	版本对比 回滚 删除	格式 text	发布人	1	
polaris.yaml	v1.0	2024-01-23 14:29:06	版本对比 回滾 删除	备注 -	发布时间	2024-01-23 14:29:06	
				1 server: 2port:8082			

3. 在操作栏单击版本对比,可以查看当前版本与历史版本差异对比。



版本对比	×
所遗版本 v2.0	对比版本 <b>v1.0 v</b>
v2.0	v1.0
1 server: 2- port:8062	1 server: 2+ · · · · · port:8081

4. 在操作栏单击回滚,在弹窗中二次确认后可回滚配置。

确定回滚配置polaris.yaml到以下版本?					
版本号	v1.0				
版本备注					
	<b>确定</b> 取消				



# 配置轨迹

最近更新时间: 2025-06-11 15:55:02

# 操作场景

该任务指导您通过 TSF 控制台,如何在某一北极星引擎下的配置下发轨迹,查看配置数据从服务端下发至运营端的完整过程与状态流转路径。

# 前提条件

- 已 创建北极星引擎实例。
- 已 创建命名空间。

# 操作步骤

## 创建配置文件

- 1. 登录 TSF 控制台,在左侧导航栏单击 Polaris (北极星)进入实例列表详情页。
- 2. 单击目标实例 ID,进入引擎实例信息页,左侧选择配置管理后,进入配置管理详情页。
  - 2.1 选择**发布历史**

选择目标发布记录后,单击操作栏的**配置轨迹**后跳转至**配置轨迹**页面。

记置管理									
配置分组	发布历史 配置制	九迹							
						全部命名空间	<ul> <li>请选择条件;</li> </ul>	进行过滤	QB
名称	版本	配置分组 丁	命名空间	发布类型	发布人	发布时间	状态	操作	
	2025060302	a 3t	default	发布	10 3	2025-06-03 11:08:19	成功	配置轨迹	
-	20250603	a st	default	发布	10 3	2025-06-03 10:50:25	成功	配置轨迹	
共 2 条						20 ~ 条 /	页 🛛 🖣	1 /1页	► ►

配置管理

配置分组 发布历史 <b>配置轨迹</b>						
近30分钟 近1小时 近1天 近7	₹ 2025-04-08 17:40:39 ~ 2025-04-15 17:40:39	Ë				金額命名空间 → 请选择条件进行过滤 Q 3
配置文件名称	版本	配置分组 7	命名空间	客户端IP	客户端类型 V	客户端更新时间
file1		g1	default	9 5	agent	2025-04-15 16:51:15
instance-id-xxx	host-xxx	service-a	namespace-a	is .	agent	2025-04-14 18:43:00
Instance-Id-xxx	host-xxx	service-a	namespace-a	l¢	client	2025-04-14 18:43:00
Instance-Id-acc	host-xxx	service-a	namespace-a	ŧ	sdk	2025-04-14 18:43:00
Instance-Id-acc	host-xxx	service-a	namespace-a	l¢	sdk	2025-04-14 18:43:00
Instance-Id-xxx	host-xxx	service-a	namespace-a	it.	sdk	2025-04-14 18:43:00
						4 >

2.2 选择**配置轨迹**标签页,可以通过配置分组、客户端IP、客户端类型筛选并查看目标配置轨迹。



# 配置 Agent 安装

最近更新时间: 2025-06-11 16:08:51

# 功能概述

北极星创建的配置文件支持推送到本地的对应路径下,创建对应的文件内容。 订阅对应服务端的文件配置后,polaris-config-agent 会自动感知服务端是否有变更,有变更后会同步更新本地文件。

## 操作步骤

# CVM 场景

#### 步骤1:下载 agent

- x86版本: polaris-config-agent-x86\_64.tar.gz
- arm版本: polaris-config-agent-arm64.tar.gz

### 步骤2:解压

解压下载的 agent.tar.gz 物料包: tar -zxvf polaris-config-agent.tar.gz 解压后生成的目录如下:

--|-bin |-config

#### 步骤3:启动

到 bin 目录下执行启动脚本 sh start.sh 具体参数有2种方式注入:

1. 通过 start.sh 后添加启动参数。

-discover-addresses=\${polaris\_discover\_addresses} -config-addresses=\${polaris\_config\_addresses} config-groups=\${polaris\_config\_groups} -config-files=\${polaris\_config\_files} -configcoken=\${polaris\_config\_token}

2. 通过配置文件, agent 启动会自动读取 config 目录下的 polaris-config-agent.yaml 配置文件,可在 config 目录下 cp polaris-config-agent.example.yaml 并在 polaris-config-agent.yaml 后在配置文件中填写具体的配置内容。

#### △ 注意:

启动参数和配置文件都有内容的情况下,优先级为:启动参数 > 配置文件。

#### 容器场景

#### 步骤1: 获取镜像地址

agent 镜像地址: ccr.ccs.tencentyun.com/tsf\_100011913960/polaris-agent

## 步骤2:配置 sidecar 容器

将 polaris-config-agent 以 sidecar 的方式启动,agent 参数可通过配置环境变量、configmap 的方式注入,具体参数介绍如下: 参数介绍

参数	描述	支持环境变量注 入	参数示例	环境变量注入示例	是否必须
discove r-	北极星服 务发现地 址	\${polaris_dis cover_addre sses}	-discover- addresses=127.0.0.1:8 091,127.0.0.2:8093	-discover- addresses=\${polaris_ discover_addresses}	可选,如果不填,则 discover- address=config_ip:8091



address es					
config- address es	配置中心 访问地址	\${polaris_co nfig_address es}	-config- addresses=127.0.0.1:8 093,127.0.0.2:8093	−config− addresses=\${polaris_ config_addresses}	必须
config- token	配置中心 访问 token	\${polaris_co nfig_token}	-config-token=aaaa	−config− token=\${polaris_confi g_token}	可选
config- groups	待获取的 配置分组	\${polaris_co nfig_groups}	−config− groups=ns1#group1,n s2#group2	−config− groups=\${polaris_con fig_groups}	config-groups 和 config- files 任意有一个或两个都有
config- files	待获取的 配置文件	\${polaris_co nfig_files}	−config− files=ns1#group1#file1, ns2#group2#file2	−config− files=\${polaris_config _files}	config-groups 和 config- files 任意有一个或两个都有

# 可观测性 引擎运行监控

最近更新时间: 2025-06-04 11:46:02

# 操作场景

北极星对运行的实例、接口提供了多项监控指标,用以监测北极星网格节点及服务的运行情况,例如:请求失败总数、请求延迟等**请求指标**,CPU、内存、网 络等**系统指标**。您可以根据这些指标实时了解北极星网格的运行状况,针对可能存在的风险及时处理,保障系统的稳定运行。本文为您介绍通过 TSF 控制台 查看北极星监控数据的操作方法。

# 监控指标及含义

## 注册配置指标

指标分类	指标名	指标含义
注册中心	总服务数	注册中心内所有已完成注册的服务总量,包含在线、离线、异常等全状态 服务
	在线服务数	当前至少存在1个健康实例的服务数量,可正常接收请求
	离线服务数	在线服务实例为0的服务数量
	异常服务数	存在服务实例但无健康节点的服务数量
	总服务实例数	所有服务实例的全局计数,包含健康、异常、隔离、未隔离状态
	在线服务实例数	通过健康检查且可响应请求的实例数量
	异常服务实例数	连续心跳超时或主动标记为不可用的实例数量
	隔离服务实例数	因熔断策略或人工操作暂停流量转发的实例数量
配置中心	配置文件总数	配置文件数量
	配置分组总数	配置分组数量
	已发布配置文件总数	已推送至客户端的配置文件数量

### 请求指标

指标分类	指标名	指标含义
请求数	请求失败数(次)	注册中心集群内各节点/接口的请求失败数。
	请求数(次)	注册中心集群内各节点/接口的请求次数。
	请求成功数(次)	注册中心集群内各节点/接口的请求成功数。
	网络失败数(次)	注册中心集群内各节点/接口的网络失败数。
请求失败细分	其他失败数(次)	注册中心集群内各节点/接口的其他失败数。
	系统失败数(次)	注册中心集群内各节点/接口的系统失败数。
	请求平均延时(ms)	注册中心集群内各节点/接口的请求平均延时。
请求延时	最大请求延迟(ms)	注册中心集群内各节点/接口的最大请求延迟。
	请求最小延时(ms)	注册中心集群内各节点/接口的请求最小延时。

## 系统指标



指标名	指标含义
CPU使用率 (%)	注册中心集群内各节点的 CPU 使用率。
内存使用率(%)	注册中心集群内各节点的内存使用率。
网络入包量(个/s)	注册中心集群内各节点的网络入包量。
网络入流量(MBytes)	注册中心集群内各节点的网络入流量。
网络出包量(个/s)	注册中心集群内各节点的网络出包量。
网络出流量(MBytes)	注册中心集群内各节点的网络出流量。
磁盘读取次数(次)	注册中心集群内各节点的磁盘读取次数。
磁盘读取带宽(MBps)	注册中心集群内各节点的磁盘读取带宽。
磁盘写入次数(次)	注册中心集群内各节点的磁盘写入次数。
磁盘写入带宽(MBps)	注册中心集群内各节点的磁盘写入带宽。

# 查看监控入口

1. 登录 TSF 控制台,在左侧导航栏选择 Polaris (北极星)进入引擎实例列表页。

- 2. 在引擎实例列表页,单击目标引擎的"ID",进入基本信息页面。
- 3. 在左边页签单击**引擎运行监控**,可查看业务指标和系统指标。

### 注册配置指标

选择好**指标、节点**和命名空间,设置好时间范围(支持近1小时、近12小时、近1天、近30天和自定义),可查看对应的监控数据。

有指标 > 所有节点	◇ 所有命名空间 ◇						
小时	<ul> <li>〇 时间程度: 1分钟 ~ ジ 关闭 ~</li> </ul>	🗸 宮元田利					
注册中心							
服务数(个) ①	A [] ···	在线服务数(个) ①	♣ [] ···	南线服务数(个)①	♣ ⊡ …	异常服务数(个) ①	▲ □ …
5	105411.000	1	100411000	0.5		0.6	
0 4.42 14.47 14:52 14:57 15:02 15:07	15:12 15:17 15:22 15:27 15:32 15:37	0	5.27 15:32 15:37	0	15:32 15:37	0	15:32 15:37
polaris-0   ins-fcfebb37   Polaris 最大值: 1.00 最, polaris-0   ins-fcfebb37   default 最大值: - 最小伊 polaris-1   ins-fcfebb37   Polaris 最大值: 1.00 最,	Ⅰ小蛋: 1.00 平均值: 1.00 值: - 平均值: - 川小蛋: 1.00 平均值: 1.00	■ polaris-0 [ins-feteb637] Polaris 部大倍: 100 最小信: 100 平均倍: 1.00 ■ polaris-0 [ins-feteb637] default 最大信: - 最小信: - 平均信: - ■ polaris-1 [ins-feteb637] Polaris 最大信: 1.00 最小信: 1.00 平均信: 1.00	I	■ polaris-0   ins-fcfebb37   Polaris 最大信: 0.00 最小值: 0.00 平均值: 0.00 ■ polaris-0   ins-fcfebb37   default 最大值: - 最小值: - 平均值: - ■ polaris-1   ins-fcfebb37   Polaris 最大值: 0.00 最小值: 0.00 平均值: 0.00	I	■ polaris-0   ins-fcfebb37   Polaris 数大值: 0.00 最小值: 0.00 平均值: 0.00 ■ polaris-0   ins-fcfebb37   default 最大值: - 最小值: - 平均值: - ■ polaris-1   ins-fcfebb37   Polaris 最大值: 0.00 最小值: 0.00 平均值: 0.00	I
服务实例数(个) ①	<b>4</b> C ···	在线服务实列数(个) ①	<b>*</b> D	异紫猫务实例数(个) ①	<b>▲</b> □ ···	編載服务実例数(个)①	▲ □ …
2	15:41 2.00	2	15:41 2:00	1		1	
0	10-12 10-17 10-22 10-27 10-22 10-21	0	6.07 16.02 16.07	0 1442 1447 1652 1657 1502 1507 1512 1517 1500 1507	15-22 15-27		15.92 15.97
polaris-0   ins-fcfebb37   Polaris 最大语: 2.00 最	シーム 10.07 10.02 10.02 10.02 10.07	■ polaris-0   ins-fcfebb37   Polaris 最大値 200 最小値 2.00 平均値 2.00	0.6.7 10:02 10:07	■ polaris-0   ins-fcfebb37   Polaris 最大価 - 0.00 単分類 0.00 平均類 0.00	10:07	■ polaris-0   ins-fcfebb37   Polaris 最大值: 0.00 最小值: 0.00 平均值: 0.00	10.07
polaris-0   ins-fcfebb37   default 最大值: - 最小信   polaris-1   ins-fcfebb37   Polaris 最大值: 2.00 最/	信: - 平均值: - 时/值: 2.00 平均值: 2.00	■ polaris-0   ins-fcfebb37   default 最大值: - 最小值: - 平均值: - ■ polaris-1   ins-fcfebb37   Polaris 最大值: 2.00 最小值: 2.00 平均值: 2.00		■ polaris-0   ins-fcfebb37   default 最大值: - 最小值: - 平均值: - ■ polaris-1   ins-fcfebb37   Polaris 最大值: 0.00 最小值: 0.00 平均值: 0.00		■ polaris-0   ins-fcfebb37   default 最大值: - 最小值: - 平均值: - ■ polaris-1   ins-fcfebb37   Polaris 最大值: 0.00 最小值: 0.00 平均值: 0.00	
配置中心							
■文件总数(个) ①	<b>▲</b> D ···	配置分组总数(个) ①	♣ □ …	已发布配置文件总数(个) ①	♣ □ …		
		40		40			
) ) 1.42 14:47 14:52 14:57 15:02 15:07 1	15:12 15:17 15:22 15:27 15:32 15:37	40 20 0 14:42 14:47 14:52 14:57 15:02 15:07 15:12 15:17 15:22 15	5.27 15:32 15:37	40 20 0 4.42 14.47 14.52 14.57 15.02 15.07 15.12 15.17 15.22 15.27	15:32 15:37		
) ) 4.42 14.47 14.52 14.57 15:02 15:07 1 polaris の目ins-fetebb37 [Polaris 慶大臣:一慶小師師 日のにからりになるたちおわろ7 (Advit 豊大郎) 一時間	15:12 15:17 15:22 15:27 15:32 15:37	40 20 0.4.4.2 M-47 M-52 M-57 15:02 15:07 15:12 15:17 15:22 15 ■ politin-0 line-fetebb23 / polaris 第大語: ●以何道: 平均通: ■ politin-0 line-fetebb23 / polaris 第大語: ●以何道: 平均通: ■ politin-0 line-fetebb23 / polaris 第大語: ●以何道: 平均通:	5.27 15.32 15.37	40 0 0 14.2 14.47 14.52 14.57 15.02 15.07 15.12 15.17 15.22 15.27 ■ polaris-0 ins-febb237 (shou B-476,	15:32 15:37		
0 0 4.42 16-47 14:52 14:57 15:02 15:07 1 1 polaris-0 (ins-fclebb37) Polaris 副大龍: 最小儀 1 polaris-0 (ins-fclebb37) fotault 最大龍: 最小儀 1 polaris-1 (ins-fclebb37) Polaris 最大龍: 最小儀	15:12 15:17 15:22 15:27 15:32 15:37 面:- 平均值:- 面:- 平均值:- 面:- 平均值:-	40 20 0 44-2 44-27 44-22 44-27 15-02 15-07 15-12 15-17 15-22 17 10 patient (0) (modelski27) (Patient 第756 - 後の後 - 平平田 - 10 patient (0) (modelski27) (Patient 第756 - 後の後 - 平平田 - 11 patient-01 (modelski27) (Patient 第756 - 最少後 - 平平田 -	5.27 15.32 15.37	40 50 64 64 64 64 64 64 64 64 64 64	15.32 15.37		
0 1 2 2 2 2 2 2 2 2 2 2 2 2 2	15:00 15:07 15:22 15:27 15:32 15:37 18 - 47948 - 16 - 47948 - 16 - 47948 -	00 0 0 0 0 0 0 0 0 0 0 0 0	5.27 15.32 15.37	40 50 64.2 M-47 M-82 M-67 M-50 M-57 M-12 M-17 M-22 M-27 <b>B</b> paties () In-640627 (Paties B/KB, B-166, -1968) 5 paties () In-640627 (Paties B/KB, B-268), -1968 - 8 paties () In-640627 (Paties B/KB, B-268), -1968 -	15.32 15.37		
New York Hard Land Tology Barry positive () Into Active 27 () Policy B 274 - 1940 positive () Into Active 27 () Policy B 2748 - 804 positive () Into Active 27 () Policy B 2748 - 804 policy () Policy B 2748 - 804	1610 1617 1622 1627 1633 1637 14. 中容明 - # PRA - # PRA - # PRA -	00 0 0 444 5 444 7 862 8457 862 867 863 867 862 867 0 90440 () n= 44827 [Paula 27:8] 4878 - 8708 - 9708 9 paints () n= 4502 () 4044 8 878 - 878 - 978 - 0 paints () n= 45020 () 9946 878 - 8798 - 9798 -	8.27 15.32 15.37	40 142 1447 1452 1457 1502 1507 1512 157 1522 1527 ■ data of Ins 458237 1512 157 1522 1527 ■ data of Ins 458237 1512 1527 1528 1528 ■ data of Ins 45837 1512 1529 1528 1528 1528 1528 1528 1528 1528 1528	15.32 15.37		
Aug Lutz Aug Lutz 1000 000 postacio (non-classiz) Posta 87.4 - 0.4 postacio (non-classiz) postacio (non-classiz) Posta 87.4 - 0.4 postacio (non-classiz) Posta 87.4 - 0.4 postacio (non-classiz) Posta 87.4 - 0.4	1610 1617 1622 1627 1632 1637 G. 中796 - a. 中796 - a. 中796 -	00 04 04 04 04 04 04 04 04 05 04 05 04 05 04 05 04 05 04 05 04 05 04 05 04 05 04 05 04 05 04 05 04 05 05 05 05 05 05 05 05 05 05 05 05 05	827 15.92 15.97 D	40 50 44.2 14.0 14.0 14.0 14.0 16.0 16.0 16.0 16.0 16.0 16.0 16.0 17 <b>#</b> paties of particlessor (Patient BK/RE - BU/RE - PR) <b>#</b> paties of particlessor (Patient BK/RE - BU/RE - PR) <b>#</b> paties of particlessor (Patient BK/RE - BU/RE - PR) <b>#</b> paties of particlessor (Patient BK/RE - BU/RE - PR) <b>#</b> paties of particlessor (Patient BK/RE - BU/RE - PR) <b>#</b> paties of particlessor (Patient BK/RE - BU/RE - PR) <b>#</b> paties of particlessor (Patient BK/RE - BU/RE - PR) <b>#</b> paties of particlessor (Patient BK/RE - BU/RE - PR) <b>#</b> paties of particlessor (Patient BK/RE - BU/RE - PR) <b>#</b> paties of particlessor (Patient BK/RE - BU/RE - PR) <b>#</b> paties of particlessor (Patient BK/RE - BU/RE - PR) <b>#</b> paties of particlessor (Patient BK/RE - BU/RE - PR) <b>#</b> paties of particlessor (Patient BK/RE - BU/RE - PR) <b>#</b> paties of particlessor (Patient BK/RE - BU/RE - PR) <b>#</b> paties of particlessor (Patient BK/RE - BU/RE - PR) <b>#</b> paties of particlessor (Patient BK/RE - BU/RE - PR) <b>#</b> paties of particlessor (Patient BK/RE - BU/RE - PR) <b>#</b> paties of patiest (Patient BK/RE - BU/RE - PR) <b>#</b> paties of patiest (Patient BK/RE - BU/RE - PR) <b>#</b> paties of patiest (Patient BK/RE - BU/RE - PR) <b>#</b> patiest (Patient BK/RE - BU/RE - BU/R	15.32 15.37		
A LET USS UST SOL UST province 0 (nor-factory) (norm EVR - 040 province 0) (nor-factory) (norm EVR - 040 province 1) (Norm EVR - 040 province	1612 1617 1622 1627 1632 1637 ■ 7782 · ■ 7782 · ■ 7782 · ■ 7872 · ■ 7872 ·	1 1 1 1 1 1 1 1 1 1 1 1 1 1	5.27 15.32 15.37 0	40 50 40 50 40 50 50 50 50 50 50 50 50 50 5	15.32 15.37		
ALC UND	828 827 822 827 832 837 ■ * 7984 - ■ * 7984 - 说明 单击可查看监招	************************************	6.27 16.32 16.37 U	40 0 0 0 0 0 0 0 0 0 0 0 0 0	16.92 16.97		
A AF HES LAT DOL 051 BO partice () Inc. Actabase) Plants 87.8 Brid partice () Inc. Actabase) () Adda 87.8 Brid partice () Inc. Actabase) () Adda 87.8 Brid partice () Inc. Actabase) () Adda 87.8 Brid Adda 87.8 Brid Adda 87.8 Brid Partice () Adda 87.8 Brid Adda 87.8 Brid Partice () Adda 87.8 Brid Brid Adda 87.8 Brid Adda 87.8 Brid Brid Adda 87.8 Brid Adda 87.8 Brid Brid Adda 87.8 Brid Brid Brid Adda 87.8 Brid Brid Brid Brid Brid Brid Brid Brid	1812 1817 1822 1827 1833 1837 ▲ - 4784 - ■ - 7874 - ■ - 7874 - ● - 前日 単击可查看监招	1 1 1 1 1 1 1 1 1 1 1 1 1 1	5.27 15.32 19.37	40 42 442 447 442 447 452 457 552 557 1512 157 552 557 <b>B</b> data (a) In effector (find) 878 - 878 - 878 - 878 5 data (i) ne (find) (find) 878 - 878 - 878 - 878 - 878 5 data (i) ne (find) (find) 878 -	15.02 15.07		
ALE	20 0 07 022 027 032 037 a. +700. a. +700. a. +700. b. +700. i. +700. i. +700. i. +700. i. +700. ii. +700. iii. +70	2 2 2 2 2 2 2 2 2 2 2 2 2 2	827 1632 1637	40 9 9 9 9 9 9 9 9 9 9 9 9 9	15.22 15.27		
AB MAT MAD MAT NOT FOR AT A STATE AND A	100 107 102 107 103 1037 ▲ ***** ▲ ***** ▲ ***** ● ·**** ·**** 単击可查看监控 単击可刷新获取	2 2 2 2 2 2 2 2 2 2 2 2 2 2	5.27 15.32 15.37 U	40 42 442 442 442 447 448 447 448 447 448 447 448 447 448 447 448 447 448 447 448 448	10.02 10.07		



✔ 显示图例	勾选后可在图表上显示图例信息。
	跳转至 腾讯云可观测平台控制台 配置告警策略。

#### 系统指标

选择好**指标、节点**和接口,设置好时间范围(支持近1小时、近12小时、近1天、近30天和自定义),可查看对应的监控数据。

	21 LL 10				
所有指标 > 所有节点	→ 所有接口 ・	~			
1/Julitj É	⑤ 时间粒度: 1分钟 / 〇 关闭、	/ 🗹 显示图例			
请求数					
请求数(次) ①	<b>▲</b> □ …	请求成功数(次) ①	▲ D ···	请求失败数(次) ①	▲ □ ···
210	15:37 <b>197.00</b>	210	15:37 <b>197.00</b>	1.2	
140		140		0.8	
70		70		0.4	
0	E40 1E47 1E00 1E07 1E00 1E07	0	15-10 15-17 15-00 15-07 15-00 15-07	0	15.17 15.00 15.07 15.00 15.07
	10.12 10.17 10.22 10.27 10.02 10.07	=	n 174548, 2.02		10.17 10.22 10.27 10.02 10.07
■ polaris-0   ins-icrebb37 最大值: 4:00 最小值: 2:00	5.00 平均值: 195.08	polaris-0   ins-fcfebb37 最大值: 4.00 最小值: 2.00 平均值: 2.03		■ polaris-0   ms-fcfebb37 最大值: 0.00 最小值: 0.00 平均值: 0.00	
1.2 0.8 0.4 1 4.42 14.47 14.52 14.57 15.02 15.07 11 14.42 10.47 11.452 14.57 15.02 15.07 11 m. polaris-1 [ins-fc1ebb37 最大電: 0.00 最小電: 0.00	5-12 15-17 15-22 15-27 15-32 15-37 平均键: 0.00 平均键: 0.00	12 0.8 0.4 0 14-42 14-47 14-52 14-57 15:02 15:07 1 14-42 14-47 14-52 44-57 15:02 05:07 1 章 polaris-1 (ins-fc1ebb37 最大值: 0.00 最小值: 0.00	5.12 15.17 15.22 15.27 15.32 15.37 9 坪均值: 0.00 9 平均值: 0.00	1.2 0.8 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4	15:17 15:22 15:27 15:32 15:37 ffr: 0.00 ff: 0.00
请求延时					
请求平均延时(ms) ①	♣ E …	请求最大延时(ms) ①	▲ D …	请求最小延时(ms) ①	▲ D …
1.2 15:02 1.09 0.8		2.7 1.8 0.9 0	5:10 2.49		15:38 0.06
0.4 0.1 14:42 14:47 14:52 14:57 15:02 15:07 15	5:12 15:17 15:22 15:27 15:32 15:37	14:42 14:47 14:52 14:57 15:02 15:07 1	15:12 15:17 15:22 15:27 15:32 15:37	14:42 14:47 14:52 14:57 15:02 15:07 15:12	15:17 15:22 15:27 15:32 15:37

### 系统指标

选择好**时间范围**(支持近1小时、近12小时、近1天、近3天和自定义)、**指标**和节点后,查看相应的监控数据。


新御御橋 ~ 所有节点	*						
1/Jaj 🗂 🖸	1111111度 1分钟 ~ ② 美術~	🗹 显示批判					
系统指标							
崔盘读取带宽(MBit/s) ①	<b>▲</b> □ …	磁盘读取次数(Count/s) ①	▲ □ …	磁盘写入带宽(MBit/s) ①	4 [] ··· 磁盘写入次数(	:ount/s) ①	♣ ⊡
1.2 0.8 0.4 0 14:41 14:46 14:51 14:56 15:01 15:06 15:11 15	16 15:21 15:26 15:31 15:36	12 0.8 0.4 0 14:41 14:46 14:51 14:56 15:01 15:06 15:11	15:16 15:21 15:26 15:31 15:36	0.0012 0.0008 0.0004 14:41 14:46 14:51 14:56 15:01 15:08 15	1522 0.00 0.15 0.1 0.5 0 11 15:16 15:21 15:26 15:31 15:36 0 14:41 14:0	46 14:51 14:56 15:01 15:06 15:11 15:16	15:22 0.13
■ ins-fcfebb37   system   polaris-0 最大值: 0.00 最小值: 0.00 ■ ins-fcfebb37   system   polaris-1 最大值: 0.00 最小值: 0.00	平均值: 0.00 平均值: 0.00	■ ins-fcfebb37  system   polaris-0 最大值: 0.00 最小值: ■ ins-fcfebb37  system   polaris-1 最大值: 0.00 最小值:	0.00 平均值: 0.00 0.00 平均值: 0.00	■ ins-fcfebb37   system   polaris-0 最大信: 0.00 最小信 ■ ins-fcfebb37   system   polaris-1 最大信: 0.00 最小信	0.00 平均値: 0.00 0.00 平均値: 0.00 目ins-fcfebb: ins-fcfebb:	.7   system   polaris-0 最大值: 0.12 最小值: 0.05 平均值 17   system   polaris-1 最大值: 0.13 最小值: 0.04 平均值	: 0.08  : 0.08
网络入流量(MBytes/s) ①	A D ···	网络入包量(Count/s) ①	A D	网络出流量(MBytes/s) ①	♣ [] ···· 网络出包量(Co	unt/s) ()	<b>*</b> D ·
6 15:04 4.47 4 2 0 14:41 14:46 14:51 14:56 15:01 15:08 15:11 15	18 15:21 15:26 15:31 15:38	400 14:44 333,04 200 0 14:41 14:58 14:58 15:01 15:08 15:01	15:18 15:21 15:28 15:31 15:38	2.4 1.6 0.8 0.8 0.4 1.451 1456 15.01 15.06 15.01	15:18 15:21 15:28 15:31 15:38	6 357,04 6 14:51 14:56 15:01 15:06 15:11 15:18	15:21 15:26 15:31 15:38
■ ins-fcfebb37   system   polaris-0 最大值: 4.47 最小值: 2.74 ■ ins-fcfebb37   system   polaris-1 最大值: 4.45 最小值: 2.71	平均值: 3.58 平均值: 3.61	<ul> <li>ins-fcfebb37   system   polaris-0 最大值: 310.00 最小</li> <li>ins-fcfebb37   system   polaris-1 最大值: 333.04 最小</li> </ul>	直: 297.96 平均值: 303.65 直: 317.56 平均值: 324.39	■ ins-fcfebb37   system   polaris-0 最大信: 2.22 最小值 ■ ins-fcfebb37   system   polaris-1 最大信: 2.25 最小值	1.35 平均値: 1.77 1.37 平均値: 1.83	17   system   polaris=0 最大值: 343.04 最小值: 327.52 % 17   system   polaris=1 最大值: 357.04 最小值: 338.52 %	平均值: 335.00 平均值: 346.89
こPU使用率(%)①	<b>▲</b> D …	内存使用率(%) ①	♣ D …				
	15:36 0.35	2.1 1.4 0.7 0.4 1.4 0.7	15:34 1.90				
ins-fcfebb37   system   polaris-0 最大量: 0.34 最小量: 0.20	平均值: 0.26	■ ins-fcfebb37   system   polaris-0 最大值: 186 最小值:	1.81 平均值: 1.84				

### 运行日志

### 支持查看实时日志和CLS日志。

### 1. 实时日志

1.1 选择好节点后,可查看对应节点的运行日志。

1.2 在日志页面的搜索框,可以通过关键字查询相关日志。输入关键词查询,例如:"info",注意日志检索区分大小写。

← test (ins-fcfebb3	n	🗅 Hirr
- 引擎發現	<b>文所28 文明24 x5 祭王 <u>副行在室</u> 学部区面</b>	
· #628		
注册中心	RNBS CLSBS	
- 服务管理	Tin point-9 · BIRRET into O Q	
治理中心	X#ree	
- 約5路由		
<ul> <li>就近路由</li> </ul>	1 2025-04-2013 A2 20-0720720-0488 2024-0-2013 A2 20-0402 (mode service-contract, apr) [Cabil Environmenta, Cabil Environmenta,	<b>4</b>
- 焙香降级	3 2025-05-2015-0427 20734400-0480 2024-05-2015-0427 204952 100 code service/law-gala [Cale] [Land] Element are rise rise rise rise rise (*gala**, f_solat**, f_solat*	
- <b>这问题说</b>	5 2025-08-2015-0427 20289721-048 2025-0-2015-0427 209728 (20) Gold public polymorphills Disferred Taxes 2025-05-31 53:042 -0488 (37), humitiae spatier (nn x 2025-05-31 11):044 -0488 (37) 5 2025-0-2015-0427 202721-048 2025-05-31 11):044 -0488 (37) 6 2025-0-2015-0427 202721-048 2025-05-31 11):044 -0488 (37) 6 2025-0-2015-0427 2025-05-31 11):044 -0488 (37) 6 2025-0-2015-0427 2025-05-31 11):044 -0488 (37) 6 2025-0-2015-0427 2025-0421-01):045-0488 (37) 6 2025-0-2015-0427 2025-0421-01):045-0488 (37) 6 2025-0-2015-0427 2025-0421-0428 6 2025-0-2015-0427 2025-0428 6 2025-0-2015-0427 6 202	
<ul> <li>服务될权</li> </ul>	7 2025-05-2012-0427 202025-04215-0427 202025-045-045-0425-0425-0425-0425-0425-	
. 金額請衣度	5 2027-45-0112-6023 002582-0140 021-6-2112-612 0027 0140 021 021-6-2112-612 021-6	<b>4</b>
配置中心	1 362-45-3413142.1 5985101-0418 392-45-34131428 395462 min or 1 (2014) [Instan] pt mer rola (2014) [Instan]	41.1
- 配置管理	1. 2019 - FILL AND	
可观测性	s adopt-biliseda monounement work-p-biliseda annual (b) care writen	
- 引擎运行监控	5. 4.0-p-3H1/412. WALLSHORE W 20-p-3H1/412.	
<ul> <li>注册配置监控</li> </ul>	a double information and or international and the case and international parts (case)	
· 服务调用监控	25, 262-6- 38733-0473, 36974820-6488 253-6-38733-0478, 2019 code: agi/yesp-pi84 (Codelloaria Sarte-Grazia Sard-6-29 Sard-29 Milloi (Codelloaria Sarte-Grazia Sard-6-29 Sard-29 Milloi (Codelloaria Sard-6-29 Sard-29 Milloi (Codelloaria)))	
. Prometheux道控	22.225-6-31751-01.2002/176-01.01.2025-6-31753-01.20040021 and exercise fractional sequence fractional sequ	
東鉄市心	24.2825-05-10115:44:29.95187450+08:00 2025-05-30115:44:29.9951382 info cache service/circuitbreaker.go:97 [Cache][CircuitBreaker] get more rules {"sull-from-store": 0, "delete": 0, "last": "1970-01-01108:00:00.0000002", "used": "402.95103"}	

### 2. CLS日志

PolarisMesh 持久化日志、事件中心、操作记录、告警数据存储在CLS服务中,开启CLS日志服务后可查看。

实时日志 <b>CLS日志</b>					
<ol> <li>开启CLS日志服务后, polarismesh 创建日志集 toe_logse</li> </ol>	t,用于储存事件、操作记录与告誓记录,您在polarismesh控制台上可以查询到	《具体事件、操作记录和告誓。 CLS 日志服务为独立计费产品,例如:1007	5条事件和操作记录产生费用约10.29元/月,北极星费	用中不包括这部分费用,如无需使用可选择解绑。具体计量信息量看 <u>费用谨慎</u> 13	
					MWCLS
日志英型	日志株	日志主願	保存时长	状态	操作
室控告署	c0be4765-b281-413b-94f1-7d33d0f6040b tse_logset	e61e4561-fbaa-45b3-b14a-a77e1Qaa06be ins-3d934ta3_polarismesh_alarm	30	正常	编辑: 终刻
事件	c0be4765-b281-413b-94f1-7d33d0f6040b tse_logset	e78ec306-0cdb-4713-b9d2-450464f948fe ins-3d934fa3_polarismesh_event	30	正常	编辑 经准
操作记录	c0be4765-b28f-413b-94f1-7d33d0f6040b tse_logset	47fbef12-1008-4f6e-a488-20f8405dbe9a ins-3d934fa3_polarismesh_operation	30	正常	<b>编辑</b> 伊波
共 3 质					10 ⊻ ∦/д н « 1 /1д н н



# 注册配置监控

最近更新时间: 2025-04-17 16:00:53

Polaris(北极星)针对运行中实例的注册配置能力提供了多项监控指标,例如:服务概览数据、服务及配置数据、服务端请求数据,用以监测该实例及服务 的运行情况。您可以根据这些指标实时了解北极星的运行状况,针对可能存在的风险及时处理,保障系统的稳定运行。本文为您介绍通过 TSF 控制台查看北 极星注册配置监控数据的操作方法。

## 监控指标及说明

### 概览

指标	说明
客户端节点数	客户端总节点数。
客户端连接数	包含总连接数、注册中心连接数、配置中心连接数。
总请求数	客户端访问北极星服务端请求数。包含总请求数、成功请求数、失败请求数、请求成功率。
请求时延	客户端访问北极星服务端请求时延。包含均值、最大值、最小值、P99、P95。
服务数	包含总服务数、在线服务数、异常服务数、离线服务数。
实例数	包含总实例数、在线实例数、隔离实例数、异常实例数。
配置分组数	配置分组总数。
配置数	配置文件数、已发布配置文件数。

## 服务和配置统计

指标	说明
服务数	包含总服务数、在线服务数、异常服务数、离线服务数。
实例数	包含总实例数、在线实例数、隔离实例数、异常实例数。
配置分组数	配置分组总数。
配置数	包含配置文件数、已发布配置文件数。

## 北极星服务端请求统计

指标	说明
请求数	客户端访问北极星服务端请求数。包含总请求数、成功请求数、失败请求数、请求成功率。
请求时延	客户端访问北极星服务端请求时延。包含均值、最大值、最小值、P99、P95。
失败请求数	包含总失败请求数、5xx失败请求数、4xx失败请求数、网络失败数。
返回码分布	包含 200、5xx、4xx 返回码。

# 查看监控

- 1. 登录 TSF 控制台。
- 2. 在左侧导航栏选择Polaris (北极星) 后进入引擎实例列表页,单击目标实例 ID。
- 3. 在左侧页签选择**注册配置监控**,根据您的需求选择要查看的资源、设置好时间范围,即可查看对应的监控数据。



← hjx-test (ins-fee	d5390)	C #
- 引擎管理	概寬 服务和配置统计 北极星服务端请求统计	
。 命名空间		
注册中心	命名空间 全部命名空间汇总 > 时间范围 1h 1d 3d 7d 2025-04-15 20:32:09 ~ 2	2025-04-15 21:32:09
。 服务管理		
治理中心	客户端节点数	客户端连接数
。动态路由	总节点数	总连接数 注册中心连接数 配置中心连接数
。 就近路由	0 个	0 0 0
。 熔断降级		
。访问限流	10	10
。 服务鉴权	8	8
。 全链路灰度	6	6
配置中心	4	4
- 配置管理		
可观测性	2	2
• 注册配置监控	2025-04-15 20:32:09 2025-04-15 20:52:51 2025-04-15 21:13:10	2025-04-15 20:32:09 2025-04-15 20:52:51 2025-04-15 21:13:10
。 服务调用监控	—— 总节点数	<ul> <li>一 总连接数</li> <li>一 注册中心连接数</li> <li>一 配置中心连接数</li> </ul>
权限中心		
。 权限管理		
	<b>总请求数</b> 蜜户端访问北极是服务端请求数	<b>请求时延</b> 客户编访问北极星服务编请求时延



# 服务调用监控

最近更新时间: 2025-04-17 16:00:53

Polaris(北极星)针对运行中实例的服务调用情况提供了多项监控指标,例如服务服务请求数、服务请求时延、返回码统计等指标,用以监测该实例及服务 的运行情况。您可以根据这些指标实时了解北极星的运行状况,针对可能存在的风险及时处理,保障系统的稳定运行。本文为您介绍通过 TSF 控制台查看北 极星服务调用数据的操作方法。

## 监控指标及说明

### 概览

指标	说明
服务请求数	服务请求数量,包含总请求数、成功请求数、限流请求数、熔断请求数。
服务请求时延	服务请求时延,包含均值、最大值、最小值、P99、P95。
返回码统计	服务请求的返回码,包含 200、5xx、4xx 等返回码的次数及占比。
服务请求成功率分布	展示服务请求成功率在 100%、90%~100%、75%~90%、50%~75%、<50% 的分布情况。
服务请求时延分布	展示 P99 时延在 <5ms、5ms~10ms、10ms~50ms、50ms~100ms、>100ms 的分布情况。
服务列表	展示服务列表信息,包含服务名称、命名空间、健康/总实例数、成功率、总请求数、失败请求数等信息。

## 服务/接口/实例监控

指标	说明
服务请求数	根据筛选,按命名空间、服务、接口、实例等维度,查看服务/接口的请求数量,包含总请求数、成功请求数、 限流请求数、熔断请求数。
服务请求时延	根据筛选,按命名空间、服务、接口、实例等维度,查看服务/接口的请求时延,包含均值、最大值、最小值、 P99、P95。
返回码统计	包含 200、5xx、4xx 等返回码的次数及占比。
服务实例监控	展示服务实例监控信息,包含实例 IP、端口、监控状态、成功率、总请求数等信息。
调用者监控	展示服务调用者的监控信息,包含调用者 IP、服务名、命名空间、总请求数、成功率等信息。

## 查看监控

1. 登录 TSF 控制台

2. 在左侧导航栏选择Polaris (北极星)后进入引擎实例列表页,单击目标实例ID。

3. 在左侧页签选择**注服务调用监控**,根据您的需求选择要查看的资源、设置好时间范围,即可查看对应的监控数据。



间 全部命名空间》	E总 > 时间范围 1h	1d 3d 7d	2025-04-15 20:32:09 ~ 2025	5-04-15 21:32:09	时间粒度 1秒 >			
服务请求数				服务请求时延				
总请求数 0	成功请求数 <b>0</b>	限流请求数 <b>O</b>	增断请求数 O	<sup>均值</sup> O ms	最大值 O ms	最小值 Oms	P99 Oms	P95 Oms
10				10				
8				8				
6				6				
4				4				
2				2				
		2025.04	15 01:14:00	2025 04 15 20	2.22.00 2025 04	15 20:52:00	2025 04 15 21-1	4:00



# 监控排障

最近更新时间:2025-04-09 11:24:22

## 排障概述

本文主要介绍使用北极星进行排障的基础思路和操作步骤,方便运维人员在使用北极星的过程中进行问题定位和处理。 当您的客户端访问北极星报错时,可以查看北极星网格的监控,初步定位发生了什么错误。

# 1. 找到报错的接口

首先根据客户端的日志,定位到访问的是北极星的哪一个接口。然后进入北极星的监控页面,选择具体的接口,查看监控。如下图,选择<mark>运行监控</mark> ,在下拉列 表中选择您需要查看的接口。

所有指标 ▼ 所有节点	▼ 所有接口 ▼	
1小时 🖬	③ 时间粒度: 1分钟 ▼ ④ 关闭 ▼ ••• ▼	2 显示图例
请求数		
5d286/2m ()	清·史·#15₩//20 ① ● 「1	清史生的新(か)③ ▲「1
	ing-rid-vise(0/) ① ÷ L1 ····	
1	1	1
1.5	0.5	0.5
0 09:44 09:54 10:04 10:14 10:24 10:34 10:44	0 09:44 09:54 10:04 10:14 10:24 10:34 10:44	0 09:44 09:54 10:04 10:14 10:24 10:34 10:4
polaris-0   ins-a301aa81 最大值: - 最小值: - 平均(	■ polaris-0   ins-a301aa81 最大值: - 最小值: - 平均(	polaris-0   ins-a301aa81 最大值: - 最小值: - 平均
polaris-1   ins-a301aa81 最大值: - 最小值: - 平均(	■ polaris-1   ins-a301aa81 最大值: - 最小值: - 平均(	■ polaris-1   ins-a301aa81 最大值: - 最小值: - 平均
< > >	· · · · · · · · · · · · · · · · · · ·	
违龙生败细分		
(统失败数(次) 🚯 🔹 🛄 …	网络失败数(次) ① 🔶 [] …	其他失敗数(次) ① 🌲 门 …
1	1	1
5	0.5	0.5
	0	0
0	09:44 09:54 10:04 10:14 10:24 10:34 10:44	09:44 09:54 10:04 10:14 10:24 10:34 10:4
0 09:44 09:54 10:04 10:14 10:24 10:34 10:44		
0 09:44 09:54 10:04 10:14 10:24 10:34 10:44 ■ polaris-0   ins-a301aa81 最大值: - 最小值: - 平均1	■ polaris-0   ins-a301aa81 最大值: - 最小值: - 平均(	■ polaris-0   ins-a301aa81 最大值: - 最小值: - 平均
0 09:44 09:54 10:04 10:14 10:24 10:34 10:44 polaris-0] ins-a301aa81 最大值: - 最小值: - 平均加 polaris-1] ins-a301aa81 最大值: - 最小值: - 平均加	polaris-0   ins-a301aa81 最大值: - 最小值: - 平均( polaris-1   ins-a301aa81 最大值: - 最小值: - 平均(	■ polaris-0   ins-a301aa81 最大值: - 最小值: - 平均 ■ polaris-1   ins-a301aa81 最大值: - 最小值: - 平均
0 09x44 09:54 10:04 10:14 10:24 10:34 10:44 polaris-0 j ins-a301aa81 最大值 - 最小值 - 平均间 polaris-1 j ins-a301aa81 最大值 - 最小值 - 平均间 <	■ polaris-0   ins-a301aa81 最大值: 最小值: 平均( ■ polaris-1   ins-a301aa81 最大值: 最小值: 平均( <	■ polaris-0   ins-a301aa81 最大值: - 最小值: - 平均 ■ polaris-1   ins-a301aa81 最大值: - 最小值: - 平均
0 0944 09:54 10:04 10:14 10:24 10:34 10:34 00iaris-0 jins-a301aa81 最大值 - 最小值 - 平均 0 poiaris-1 jins-a301aa81 最大值 - 最小值 - 平均 (	■ polaris-0   ins-a301aa81 最大值 - 最小值 - 平均( ■ polaris-1   ins-a301aa81 最大值 - 最小值 - 平均( <	■ polaris-0   ins-a301aa81 最大值: - 最小值 - 平均 ■ polaris-1   ins-a301aa81 最大值: - 最小值 - 平均 (
0 09544 0954 10:04 10:14 10:24 10:34 10:44 09544 0954 10:04 10:14 10:24 10:34 10:44 00 artis-1 (ins-a301aa81 最大國 - 最小國 - 平均則 ( 補來延時)	polaris-0   ins-a301aa81 最大臣 - 最小臣 - 국가의 ■ polaris-1   ins-a301aa81 最大臣 - 국小臣 - 平均 <	■ polaris-0 jins-a301aa81 最大臣 - 最小值 - 平均 ■ polaris-1 jins-a301aa81 最大臣 - 最小值 - 平均
0 0 0 0 0 0 0 0 0 0 0 0 0 0	■ polaris-0   ins-a301aa81 最大臣 - 最小臣 - 早均( ■ polaris-1   ins-a301aa81 最大臣 - 最小臣 - 平均( 、 、 、 、 、 、 、 、 、 、 、 、 、 、 、 、 、 、 、	polaris-0 jins-a301aa81 最大臣 - 最小臣 - 平均 polaris-1 jins-a301aa81 最大臣 - 最小臣 - 平均 ( ) 请求最小级的(ms) ① 章 〔3 ···
0 0 0 0 0 0 0 0 0 0 0 0 0 0	polaris-0   ins-a301aa81 最大臣、最小臣、平均 polaris-1   ins-a301aa81 最大臣、最小臣、平均 《 《 》 》	polaris-0   ins-4301ta81 最大臣 - 最小臣 - 平均 polaris-1   ins-4301ta81 最大臣 - 最小臣 - 平均 承年小班时(ms) ①
0 0 0 0 0 0 0 0 0 0 0 0 0 0	■ polaris-0   ins-a301aa81 最大值 - 最小值 - 平均 ■ polaris-1   ins-a301aa81 最大值 - 最小道 - 平均 (	■ polaris-0   Ins-4301aa81 最大值、最小值、平均 ■ polaris-1   Ins-4301aa81 最大值、最小值、平均 ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●
0 0 0 0 0 0 0 0 0 0 0 0 0 0	■ polaris-0   ins-a301aa81 最大信 - 最小值 - 平均 ■ polaris-1   ins-a301aa81 最大信 - 最小低 - 平均 《 ● )	■ polaris-0   Ins-a301aa81 最大值 - 最小值 - 平均 ■ polaris-1   Ins-a301aa81 最大值 - 最小值 - 平均 ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●
0 0 0 0 0 0 0 0 0 0 0 0 0 0	■ polaris-0   ins-a301aa81 最大信 - 最小语 - 平均 ■ polaris-1   ins-a301aa81 最大信 - 最小语 - 平均 (	■ polaris-0 [Ins-a301aa81 最大道 - 最小道 - 平平 ■ polaris-1 [Ins-a301aa81 最大道 - 最小道 - 平平
0 0 0 0 0 0 0 0 0 0 0 0 0 0	■ polaris-0   ins-a301aa81 最大信 - 最小值 - 平均1 ■ polaris-1   ins-a301aa81 最大信 - 最小值 - 平均1 ( ) ) ( ) ( ) ( ) ( ) ( ) ( ) ( ) ( ) (	polaris-0 [Ins-a301a831 最大值 · 最小值 · 平归 polaris-1 [Ins-a301a831 最大值 · 最小值 · 平归 ( 请求最小级距對(ms) ①   〔 ] · · · · · · · · · · · · · · · · · ·

目前北极星网格监控告警系统支持的接口如下表:

接口名	接口解释
POST:/eureka/apps/{application}	eureka 客户端服务实例注册。
GET:/eureka/apps	eureka 客户端查询全量服务实例。
GET:/eureka/apps/delta	eureka 客户端增量拉取服务实例。
GET:/eureka/apps/{application}	eureka 客户端查询某个服务全部实例。
GET:/eureka/apps/{application}/{instanceid}	eureka 客户端查询某个服务下的某个实例。
PUT:/eureka/apps/{application}/{instanceid}	eureka 客户端服务实例心跳上报。
DELETE:/eureka/apps/{application}/{instanceid}	eureka 客户端服务实例反注册。
PUT:/eureka/apps/{application}/{instanceid}/status	eureka 客户端服务实例状态变更。
DELETE:/eureka/apps/{application}/{instanceid}/status	eureka 客户端删除服务实例变更。



/v1.PolarisGRPC/RegisterInstance	grpc 客户端服务实例注册。
/v1.PolarisGRPC/DeregisterInstance	grpc 客户端服务实例反注册。
/v1.PolarisGRPC/Heartbeat	grpc 客户端服务实例心跳上报。
/v1.PolarisGRPC/Discover	grpc 客户端服务发现。

例如使用下面的命令请求北极星,访问 eureka 接口注册实例。

curl -v -X POST -H "Accept: application/json" -H "Content-type: text/plain" http://{\$polaris\_ip}:8761/eureka/apps/test -d '{"instance":{"instanceId":"test-1","securePort": {"\$":"0","@enabled":"false"},"ipAddr":"{\$client\_ip}}","status":"UP","port":{"\$":" {\$client\_port}}","@enabled":"true"},"hostName":"{\$client\_ip}"}'

可以在监控页面,选择接口 POST:/eureka/apps/{application} ,看到这个请求的监控信息。

## 2. 根据监控信息定位问题

根据监控信息中的 请求失败细分 分组,可以初步定位报错的原因,目前包含以下错误类型:

- 其他错误:即北极星网格服务返回的错误码为 4xx 的错误,这类错误是入参校验失败,您需要检查您传入的参数和 http header 是否正确。如果您使用 非 eureka 的方式接入,且打印了北极星网格服务返回的日志,可以检查日志,北极星网格返回包中带了入参校验失败的原因。
- 系统错误:即北极星网格服务返回的错误码为 5xx 的错误,出现这类错误表明北极星网格出现了异常,您需要提工单,联系腾讯云工程师解决。
- 网络错误:即北极星网格服务返回的错误码为负数的错误。典型的场景是在北极星网格接收客户端数据后,向客户端写数据时,客户端断开了连接,会 产生这类错误。您需要检查下您的代码逻辑,是否提前关闭了连接。



# 配置告警

最近更新时间: 2025-04-17 16:00:53

## 操作场景

**注册配置治理中心**提供一些关键指标的配置告警功能,配置告警可帮助您及时发现问题并进行处理。本文为您介绍通过控制台配置告警的操作。

## 操作步骤

### 配置告警规则

创建的告警会将一定周期内监控的指标与给定阈值的情况进行比对,从而判断是否需要触发相关通知。当指标状态改变而导致告警触发后,您可以及时进行相 应的预防或补救措施,合理地创建告警能帮助您提高应用程序的健壮性和可靠性。

- 1. 登录 TSF 控制台。
- 2. 在**服务治理中心**下的实例列表页面,单击目标引擎的"ID",进入基本信息页面。
- 3. 在顶部页签单击运行监控,在请求指标页面单击下图告警按钮跳转至 腾讯云可观测平台控制台 配置告警策略。



4. 在告警策略页面,选择好策略类型和要设置告警的实例,设置好告警规则和告警通知模板。

- 策略类型:选择微服务引擎TSE/polarismesh。
- 告警对象:选择需要配置告警策略的 TSE 资源。
- **触发条件:** 支持选择模板和手动配置,默认选择手动配置,手动配置参见以下说明,新建模板参见 新建触发条件模板 。

```
() 说明:
```

- 指标:例如"平均请求延迟",选择统计粒度为1分钟,则在1分钟内,平均请求延迟连续N个数据点超过阈值,就会触发告警。
- 告警频次:例如"每30分钟警告一次",指每30分钟内,连续多个统计周期指标都超过了阈值,如果有一次告警,30分钟内就不会再次 进行告警,直到下一个30分钟,如果指标依然超过阈值,才会再次告警。
- **通知模板**:选择通知模板,也可以新建通知模板,设置告警接收对象和接收渠道。

### 5. 单击**完成**,完成配置。

() 说明:



有关告警的更多信息,请参见 腾讯云可观测平台告警服务 。

### 新建触发条件模板

- 1. 登录 腾讯云可观测平台控制台。
- 2. 在左侧导航栏中,单击**告警管理**,在**策略管理**页签,点击新建告警策略。
- 3. 在配置告警规则下的触发条件,点击选择模板,点击新增触发条件模板。
- 4. 在新建模板页,配置策略类型。
  - 策略类型:选择微服务引擎TSE/polarismesh,然后根据需要从请求指标(接口级)/请求指标(节点级)/系统指标中选择一个策略。
  - 使用预置触发条件:勾选此选项,会出现系统建议的告警策略。
- 5. 确认无误后,单击**保存**。

新建		×
模板名称	1-100个中英文字符或下划线	
备注	选绳,可输入100个以内字符,包含中英文字符和下 划线	
策略类型	微服务引擎 TSE / Polarismesh / 请求指标(接口级) ▼	
触发条件	✓ 指标告警	
	满足 任意 ▼ 条件时,触发告警 启用告警分级功能	
	添加	
	(R-72 RC78)	

6. 返回新建告警策略页,单击刷新,就会出现刚配置的告警策略模板。

告警管理										
告警大盘	告警历史	策略管理	基础配置							
() 如有	任何问题或建议,	请扫码加技术交流群,	我们将竭诚为您服务。							
新建策略	制除	更多操作 ▼					高级筛选 多个关键的	"用竖线 " " 分隔,多个过》	悲标签用回车键分隔	Q¢¢ ±
策略名	马称	监控类型	策略类型	告警规则	策略所属項目 ▼	关联实例数	通知模板 ▼	最后修改 ↓	告警启停 ▼	操作
	الله نار	云产品监控	TSF-部署组告警	部署细节点健康率 > 10%,统计粒度…	-	1个	-	100019696805 2024/01/15 15:58:51		复制 删除 告警历史
		云产品监控	TSF-服务告警	接收请求平均耗时 > 1毫秒,统计粒度… 接收请求量 > 1次,统计粒度1分钟,… 接收请求失败率 < 100%,统计粒度1…	-	2个	più anna a	100019696805 2024/01/15 15:58:46		复制 删除 告誓历史



# 事件中心

最近更新时间: 2025-06-04 11:46:02

## 场景说明

事件中心将 Polaris(北极星)的服务状态变更事件数据进行统一管理和展示,您可以在事件中心查看服务状态变化详情,同时您也可以为服务事件配置告警 通知规则,可帮助您及时发现问题并进行处理。

Polaris (北极星)当前支持的服务事件类型有:

- 实例上线/下线
- 实例恢复健康
- 实例出现异常
- 实例开启、关闭隔离
- 无损上线开始、结束
- 无损预热开始、结束
- 无损下线开始
- 进程结束
- 熔断触发、恢复
- 熔断到半开
- 限流恢复、恢复

本文介绍如何在 TSF 控制台上查看服务事件详情和配置事件告警规则。

### 前提条件

Polaris(北极星) 事件中心、操作记录、告警数据存储在 CLS 日志服务。使用该能力前,您需要在 Polaris 中开启该能力。开启后自动创建日志集 tse\_logset,及主题: polarismesh\_event、 polarismesh\_operation、polarismesh\_alarm,您在控制台上可以查询到这些事件、操作记录和告 警配置。

开启方案如下,选择其一即可:

• 方案一:您可在购买 Polaris (北极星)实例时开启。



 方案二:直接在事件中心页面开启,登录 TSF 控制台,在引擎实例列表页面,单击目标引擎的"ID",进入基本信息页面,在顶部页签选择事件中心, 单击立即开启。



● 方案三: 创建引擎实例后,在左侧导航栏单击的目标引擎实例列表,进入详情页后,在左侧标签栏选择**引擎管理 > 运行日志 > CLS日志**,单击**立刻开启**按 钮开启 CLS 日志服务。

- 引擎管理	实例信息 实例架构 K8s 集群 运行日志 参数配置
。 命名空间	
注册中心	实时日志 CLSH志
。服务管理	暂未开启CLS日志服务。
治理中心	1. PolarisMesh 持久化日志、事件中心、操作记录、告警数据存储在CLS服务中。如需使用上述能力,建议开启。
动太败山	2. 请注意 CLS日志服务为独立计费产品,例如:100万条事件和操作记录产生费用约10.29元/月, 北极星费用甲个包括这部分费用。具体计费信息宣看费用详情 IC
	4.北极星引擎突例将立即重启
。	
。 熔断降级	立刻开启
。 访问限流	

## 操作步骤

### 查看事件详情

- 1. 登录 TSF 控制台。
- 2. 在引擎实例列表页面,单击目标引擎的"ID",进入基本信息页面。
- 3. 在顶部页签选择事件中心,选择好时间范围(支持近7天、近30天和自定义时间范围),可查看发生的事件详情,包括时间发生服务对应实例 IP 和事件发 生的时间。



近7天 近30天	2022-02-24 16:49:19 ~ 2022-03-03 16:49:19			请选择条件进行过滤 Q 💠 🗘 新建告警配置 🗹
事件类型 ▼	命名空间 🔻	服务名	影响对象	更新时间
实例关闭隔离状态	default	EUREKA-CONSUMER-SERVICE	10. 02	2022-03-03 15:13:42
实例开启隔离状态	default	EUREKA-CONSUMER-SERVICE	10. 02	2022-03-03 15:10:24
实例下线	default	EUREKA-CONSUMER-SERVICE	10. 02	2022-03-02 21:56:42
实例下线	default	EUREKA-CONSUMER-SERVICE	10. 2	2022-03-02 21:56:39
实例下线	default	EUREKA-PROVIDER-SERVICE	10. 1	2022-03-02 21:56:31
实例下线	default	EUREKA-PROVIDER-SERVICE	10. 1	2022-03-02 21:56:28

## 配置告警规则

- 1. 登录 TSF 控制台。
- 2. 在引擎实例列表页面,单击目标引擎的"ID",进入基本信息页面。
- 3. 在顶部页签选择**运行日志**,选择**CLS日志**。
- 4. 选择日志类型为"事件"对应的**日志集**,跳转至**日志服务**控制台。

例信息 K8s 集群 事件中心	运行监控 运行日志 参数百	乙置				
实时日志 CLS日志						
<ol> <li>开启CLS日志服务后, polarism 这部分费用, 如无需使用可选择</li> </ol>	esh 创建日志集 tse_logset,用于储存事件、操作词 解绑。具体计费信息查看 <u>费用详情</u> ピ	记录与告警记录,您在polarismesh控制台上可以查讨	甸到具体事件、操作记录和告警。	CLS 日志服务为独立计费产品,例如:100万条事(	件和操作记录产生费用约10.29元/月,北极星费用中不包排	£
					解	绑CLS服务
日志类型	日志集	日志主題	保存时长	状态	操作	
监控告鑒	55a2fb95-34a3-43dc-8c40- 684996899da6 tse_logset	31214573-758d-4c30-82de- b80eb0de245d ins-fe8900f8_polarismesh_alarm	30	正常	<b>编辑</b> 修复	
操作记录	55a2fb95-34a3-43dc-8c40- 684996899da6 tse_logset	cd75300e-cece-46c9-8b14- 9e35e4228f72 ins-fe8900f8_polarismesh_operation	30	正常	<b>编辑</b> 修复	
事件	55a2fb95-34a3-43dc-8c40- 684996899da6 tse_logset	0826d85a-63c0-4488-ad9f- 8673cf9c9836 ins-fe8900f8_polarismesh_event	30	正常	编辑 修复	

### 5. 跳转至 CLS 控制台进行告警策略配置。

6. 选择日志主题名称后缀为"polarismesh\_even"的日志主题,在操作栏单击更多,选择:监控高级-新建告警策略。

日志主题 🔇 圣保罗 3 其他地域 326 ~							用户之声 13	CLS技术交流群	微信公众号 产品文档 (
创建日志主题 编辑标签 管理日志集						日志集ID: 55a2fb95-34a			୦ ମ ଝ
日志主题名称/ID	检索	监控(昨天)	日志集名称/ID 了	存储类型 🖤	日志保留时	间 描述	标签	操作	
		搜索 "日	志集ID:55a2fb95-34a3-43dc-8c40-684996899d	la6",找到 3 条结果 返回	回原列表				
ins-fe890018_polarismesh_event TSE 0826d85a+63c0-4488-ad9f-8673cf9c9836	٩	🛯 (写:314.578 存:314.578)	tse_logset 55a2fb95-34a3-43dc-8c40-684996899da6	标准存储	标准: 30天	-	0	编辑 检索分析	更多 ~
ins-fe890018_polarismesh_operation TSE cd75300e-cece-46c9-8b14-9e35e4228f72	٩	0](写:838.86B存:838.86B)	tse_logset 55a2fb95-34a3-43dc-8c40-684996899da6	标准存储	标准: 30天	-	Ø	编辑 检索分析	接入数据 f 查看采集配置
ins-fe8900f8_polarismesh_alarm TSE 312f4573-758d-4c30-82de-b80eb0de245d	٩	0](写:0MB 存:0MB)	tse_logset 55a2fb95-34a3-43dc-8c40-684996899da6	标准存储	标准: 30天	-	0	编辑 检索分析	查看索引配置 (仪表盘(0) >
共3条							2 新: 查:	建告警策略 看告警策略(0)	监控告警(0) > 数据处理(0) >
									投递消费(0) > 编辑标签 删除日志主题
									autor serveral.KO

7. 在新建告警策略页面进行告警策略配置,请参考日志服务--配置告警策略进行告警策略配置。



		Q、支持通过实例ID、IP、名称等搜索资源 体	快捷键/集	团账号	备案	工具 客服支持
● 热门新品 边缘:	安全加速平台EdgeO	ne,规则引擎支持爆存策略、自定义Cache Key等多种场景 宣看详情 >				
← 新建告警策	寶略 圣保罗					
告警名称	test					
白田状态						
称並 ()	标签键	▶ 标签值 > 🛇				
	+ 添加 🕟 键值	粘贴板				
监控对象						
地域	⑤ 圣保罗 3	低他地域 326 ~				
监控对象	日志主题	指标主题 在每个执行语句中单独选择监控对象				
	圣保罗/1个(ins-fe	8900f8_p ¥				
监控任务						
执行语句	4					
	1 34718101					
	执行语句	1 event_type:InstanceOnline				\$
	查询时间范围	近15分钟 🖌				
	预览结果 💭	原始日志				
		11.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1				
	<ul> <li>热门新品 边缘</li> <li>新建告警算</li> <li>告警名称</li> <li>启用状态</li> <li>応知文列象</li> <li>地域</li> <li>追控对象</li> <li>执行语句</li> </ul>		● 作業工業業務()         ● 作業工業業         ●	② 2014/21.26,00%, Pc. 4089/26,27.3         位 101/2         何日日         何日         何日日         何日         何日日         何日         何日	② 免疫型反映的、PC 必要發展表面         ● 校相/         ● 通知           ● 新聞 感 回線全加速年台Edgeone, 展開引電支持磁行環惑、自定文Cache Key等多种速度         意電計画         ●           ● 新聞書         ● 新聞書         ● <td< th=""><th>● 公共協議工会院的C, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,</th></td<>	● 公共協議工会院的C, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,

各事件对应的 event\_type 如下:

事件大类	事件子类	事件名称	event_type
		实例上线	InstanceOnline
	肥冬京周車件	实例下线	InstanceOffline
	版方关时事件	实例出现异常	InstanceTurnUnHealth
		实例恢复健康	InstanceTurnHealth
计四中心		无损上线开始	LosslessOnlineStart
注册中心		无损上线结束	LosslessOnlineEnd
	无损上下线及预热	服务预热开始	LosslessWarmupStart
		服务预热结束	LosslessWarmupEnd
		无损下线开始	LosslessOfflineStart
		进程结束	InstanceThreadEnd
		熔断触发	CiruitBreakerOpen
	故障熔断	熔断到半开	CircuitBreakerHalfOpen
服务治理		熔断恢复	CircuitBreakerClose
	服冬阳流	限流触发	RateLimitStart
	服务限流	限流恢复	RateLimitEnd



# 操作记录

最近更新时间: 2025-04-17 16:00:53

操作记录将 Polaris(北极星)的控制台操作以及 OpenAPI 调用产生的操作信息进行统一管理和展示,您可以在操作记录查看详情,同时您也可以为操作记 录配置告警通知规则,可帮助您及时发现问题并进行处理。

#### ▲ 注意:

此文档仅针对北极星引擎实例版本 1.14.0.0 以及更高版本。

Polaris(北极星)当前支持的操作记录资源类型有:

- 命名空间
- 服务
- 服务实例
- 路由规则
- 用户
- 用户组
- 鉴权策略
- 配置分组
- 配置文件
- 限流规则
- 熔断规则

本文介绍如何在 TSF 控制台上查看操作记录详情和配置操作记录告警规则。

### 操作步骤

## 查看操作记录详情

- 1. 登录 TSF 控制台。
- 2. 在左侧导航栏选择 Polaris (北极星),进入引擎实例列表页。
- 3. 单击目标实例引擎操作栏的控制台,输入用户名和密码,进入 POLARIS MESH 控制台页面。

新建							默认开公网,如需关闭	,请前往访问控制页中关闭。
ID/名称	地域 🕇	版本	产品版本	规格	状态	计费模式	默认账号:polaris 默认密码:polaris@ins	-(
ins-	广州	1.0.0	标准版	实例配额数:50	运行中	按量付费	2022-03-01 16:17:06	控制台 删除 更多 ▼ 😳

4. 在左侧导航栏选择操作记录。您可以查看通过北极星控制台或者 OpenAPI 的操作记录信息。

🕸 POLARIS MESH							文档	polaris *
北极星服务治理	操作记录							
② 命名空间	近7天 近30天 2022-1	2-21 16:23:47 ~ 2022-12-28 16:23:47 📋					请选择条件进行过滤	Q Ø
注册中心	操作时间	资源关型	命名空间 平	资源名称	操作类型	操作详情	操作人	
<ul> <li>         ·        服务列表         ·      ·        ·        服务别名         ·        ·</li></ul>				繁无数据				
服务网格	2022-12-28 16:23:28	服务	default	EchoServer	创建	{"name":("value":"EchoServer"),"name	polaris	
王 动态路由 🔹								
计访问跟流								
ET MERITE								
三 配置分组								
☰ 发布历史								
可说测性								
🖸 业务监控								
权限投制								
2 用户								
△ 策略								



## 配置告警规则

- 1. 登录 TSF 控制台。
- 2. 在左侧导航栏选择 Polaris (北极星),进入引擎实例列表页。
- 3. 单击目标引擎实例,进入实例信息页后单击引擎管理。
- 4. 单击运行日志,切换到 CLS 日志的页面 tab 后,单击日志类型为操作记录的日志主题。

← 🛄	(ins-6ec2ac96)					
基本信息	访问控制 K8s集群	运行监控 运行日志				
实时日志	CLS日志					
日志类型		日志集	日志主题	保存时长	状态	操作
事件		tso_logset	ins-6ec2ac96_polarismesh_event	30	正常	编辑 修复
监控告警		tse_logset	ins-6ec2ac96_polarismesh_alarm	30	正常	编辑 修复
操作记录		tse_logset	) ins-6ec2ac96_polarismesh_operation	30	正常	编辑 修复
共3条						10 ¥ 条/页 H K 1 /1页 F H

5. 单击日志服务的告警策略,单击**新建策略**,然后针对操作记录日志主题进行配置告警。

~	告警名称	请输入告暨名称	
析	启用状态		
 警 ^	标签()	标签键 <b>v</b> 标签值 <b>v</b> X	
— 略		<ul> <li>+ 添加</li> <li>② 键值粘贴板</li> </ul>	
史	收纳对象		
道组	加速	◎ 广州 154   耳他地域 29 ▼	
容模板	ille determina dita		
<b>T</b>	监控对象	日本王題 指称王題 在等个执行途句中単强选择监控对象	建议使用CQL(日志服务专用语法),针对日志检索 进行了针对性优化,使用更加简单。语法规则详见
		/ m/rl/uns-asulaasi_polark *	帮助文档 2。
题	监控任务		UCUL (日志融秀专用培法) Lucene
题	执行语句	1 执行语句1	4月72 以又1月
管理		执行语句 1 level:error   select count(*) as errCount	CQL
群管理		③ resource_type 子叔名 请输入者 ④ resource_name 字段名	
		查询时间范围 近15 〔3] namespace 字段名	
I		20 operation_type     子校名       預览結果     直     ① operator       字段名     字段名	
lL分析		+ 添加执行语句 ③ operation_detail 字段名	
务	触发条件③	③ happen_time     学校名       交互模式 ▼     ④ _SOURCE_     学校名	
		○ 本 · · · · · · · · · · · · · · · · · ·	ά.
	分组触发①		
	执行周期	固定频率 🔻 🖶 🛨 1 🕂 分钟	
		测试监控任务	
	多维分析		
	多维分析①	类型	8
		相关原始日志	X检索分析
		获取者会执行语句检索条件的原由日志,例如     行对建发童智河高超的含金田日志,按照定学     日かさ Enror日表过多被发音警讨,在音響中童者对应的     投始在学士を表示。     校知道の学生の表示。     校知道の学生の表示。     校知道の学生の表示。     校知道の学生の表示。     校知道の学生の表示。     マックロー     マックロー	総省醫費/阿范語/P的全部日志,执行自定 SV折徑司,例如: 通过! select imeCost) as time.URL group by URL 就取 食口请求耗好,通过 status>499 获取错误



# 权限控制(权限中心)

最近更新时间: 2025-04-17 16:00:53

# 基本概念

北极星场景下与账号和用户相关的概念如下:

- **主账号**:也可称为主用户,是所有北极星资源的拥有者,对所有北极星资源有操作权限。在创建 TSF 引擎的时候,TSF 默认会创建名称为 polaris 的主账户。
- 子账号: 主账户创建的协作账户。
- 用户组:北极星的用户组概念,是一组具有相同权限的用户。主账号可以通过创建用户组批量对用户进行授权与管理。用户组只能由主账号创建和授权。
- 授权: 主用户可以管理所有子用户以及用户分组的写权限。子用户可将自己拥有写权限的资源权限分配给其他子用户或者用户组。
- token: 资源权限将通过 token 进行控制和管理,用户与用户组均可生成。

## 操作场景

使用 Polarismesh(北极星) 的鉴权功能,您可以清晰地管理资源与用户的访问权限。北极星基于命名空间和服务资源维度实现权限管控。 北极星引入策略的概念,将资源的访问权限与不同的用户角色联系到一起。例如下图,



子账号和用户组默认没有写资源的权限,需要主账号为其授权,才能使子账号/用户组具有指定资源的操作权限。 本文指导您在 TSF 控制台 通过开启资源鉴权和新建策略,在主账号、子账号、资源视角下,为用户或用户组授予资源权限。

## 前提条件

主账号至少拥有一个 Polarismesh 引擎实例。若您没有,请前往 创建引擎。

## 操作步骤

## 步骤1:开启鉴权模式

1. 登录 TSF 控制台。

2. 在北极星网格下的 polarismesh 页面,单击已有的引擎实例的"ID",进入基本信息管理页面。



### 3. 在基本信息页面的鉴权模块,单击服务鉴权的开关按钮,在弹窗中单击确定开启服务鉴权。

		test	(ins-	)			
本信息	<b>રે</b> દ	方问控制	K8s集群	事件中心	运行监控	运行日志	
基本信	息						
ID	ins-						
名称		test					
地域	广州						
规格	1核1G						
状态	运行中						
鉴权							
服务鉴相	权 🔿						
	此开	关控制Polaris	mesh(北极星)的	鉴权能力。开启后,	服务注册等能力需	要客户端使用token,控制台访问也会开启鉴权。	

4. 开启服务鉴权后,子账号或用户组进行访问控制台、访问北极星资源等操作时,都会需要使用 token 鉴权,没有配置 token 的客户端将无法注册服务。

### 步骤2: 创建子账号

1. 在 polarismesh 列表页面,单击目标实例引擎操作栏的控制台,输入用户名和密码,登录北极星控制台。

新建							默认开公网,如需关闭	, 请前往访问控制页中关闭	].
ID/名称	地域 👅	版本	产品版本	规格	状态	计费模式	默认账号:polaris 默认密码:polaris@ins-	-	
ins-	广州	1.0.0	标准版	实例配额数:50	运行中	按量付费	2022-03-01 16:17:06	控制台 删除 更多 ▼	J

- 2. 进入北极星控制台后,在左侧导航栏选择**用户**,单击新建。
- 3. 输入子账户信息后,单击确定,完成子账户创建。

用户				
新建				¢
用户名	用户类型	用户id	创建时间	操作
polaris (当前登录)	主账号	16	2022-03-29 10:47:02	授权 <b>查看Token 更多 ▼</b>
user-1	子账号	e4	2022-03-29 12:16:20	授权 查看Token 更多 ▼
共 2 条			1	0▼条/页 ⊣ ◀ 1 /1页 ▶ ⊨

4. 主账号可在权限控制页面中查看所有用户或用户组的 token,并且对用户、用户组和权限策略做管理和编辑。

# 步骤3:关联用户组

说明
 只有主账号才可对用户组进行编辑操作。



1. 在北极星控制台的左侧导航栏选择**用户组**,进入用户组列表页面,单击**新建用户组**。

	用户组					
(	新雄用户组					φ
	用户组名称	用户数量	备注	创建时间	操作	
			暂无数据			
	共 0 条				10 * 条/页   4 4 1 /1页	► F

### 2. 根据您的需要,为用户组命名,并且关联子账号。

1907			
请选择用户		已选择 (1)	
输入关键字搜索	Q		C
	+	*	

# 步骤4:授权操作

您可以通过以下列表查看不同视角的授权操作。

主账号视角
<ol> <li>说明</li> <li>请确保已 开启服务鉴权。</li> </ol>
1. 使用主账号登录北极星控制台。 2. 在左侧导航栏选择 <b>权限管理 &gt; 策略</b> ,进入权限管理页面,单击新建策略。



用户用户组		
<b>请选择用户</b>	0	已选择 (0)
	4	
		**

4. 在资源栏可选择北极星的资源类型,包括命名空间、服务等资源,主账号可对所有资源进行操作。

请选择命名空间			已选择 (1)	
输入关键字搜索	C	2	Polaris	
Polaris				
default				
		$\leftrightarrow$		

5. 单击**下一步**,进入预览界面,详细展示该策略涉及的用户、用户组以及资源。确认信息无误后,单击**完成**。



6. 主账号可在权限策略列表查阅现有的权限策略,可单击编辑进行授权或删除等操作。

新建策略			¢
	Q	1.000	编辑 删除
▼ 默认策略 (2)			
策略会	•••	备注	
約默认策略 🚬			
▼ 自定义策略 (0)		用户丨用户组	
		<b>用户(1)</b> 用户组(0)	
		Mar 102	
		<b>资</b> 源	
		<b>命名空间</b> 服务	
		全部命名空间(含后续新增)	

#### 子账号视角

- 1. 仅在主账号开启了鉴权模式下,使用子账户进入北极星控制台,左侧的导航栏才会出现策略选项。
- 2. 子用户**无法**新建和编辑权限策略,只能查看现有权限策略。

<b>传略</b>				
	Q			
▼ 駅以策略(1)		备注		
▼ 自定义策略 (0)				
		用户丨用户组		
		<b>用户(1)</b> 用户组(0)		
		资源		
		命名空间 服务		
		名称	权限	
			暂无数据	

3. 子用户可将自己拥有写权限的资源权限分配给其他子用户或者用户组。具体操作请参见资源视角。

### 资源视角

- 1. 仅在主账号开启了鉴权模式下,用户在创建、编辑命名空间或服务时,才可将该资源授权给其他用户或者用户组。接下来以命名空间为例,指导您如何在资源视角给用户授权。
- 2. 在 TSE 控制台 > polarismesh,选择已存在的北极星引擎实例,进入北极星控制台。在左侧导航栏中,选择**命名空间**,进入命名空间列表管理页 面。

ξ.	描述	服务数	健康实例/总实例数	创建时间	修改时间	操作
	1000.000	2	3/3			编辑 删除
	10000	0	0/0			编辑 删除
~					20 ▼ 条/页	
<b>建命名</b> 名空间* 述	<b>空间</b> 允许数字、英文字母、 长度不超过1024个字	、-、_, 限制128 符	个字符			
	<ul> <li>用户</li> <li>用户组</li> <li>① 授予权限的用</li> </ul>	)户或者用户组,具 <sup>;</sup>	有该命名空间的读写权	限,所有用户具有读标	又限	
	请选择用户			已选择 (0)		
	输入关键字搜索		Q			
			$\leftrightarrow$			
			↔			
	□ <p< td=""><td></td><td></td><td></td><td></td><td></td></p<>					

# 注意事项

🔗 腾讯云



- 1. 只有主账号开启服务鉴权功能后,才能在控制台查看**策略**列表。
- 2. 资源隔离性:
  - 所有子用户,对主账号的资源都默认具备读权限。
  - 所有子用户或者用户分组,都默认获得对所创建的资源的写权限。
  - 子用户可以获得所在的用户分组的所有资源写权限。
- 3. 每个用户初始都有一个策略默认策略。
  - 子用户没有创建资源的情况下,默认策略中资源为空。
  - 用户创建资源后,自动添加资源至默认策略。
  - 删除资源时,引擎将自动把该资源从默认策略中删除。
- 4. 如果之前向 Polarismesh 引擎实例导入过腾讯云子账户,建议尽快使用北极星主账户修改已有子账户的登录密码。已有子账户的登录信息如下。
  - 用户名: { 腾讯云子账户用户名称 }
  - 密码: polarismesh@2022



# Java 应用开发 Spring Cloud 应用开发 Spring Cloud Tencent 概述

最近更新时间: 2024-02-21 14:51:31

目前 Spring Cloud Tencent 主要提供了微服务领域常见的服务注册与发现、配置中心、服务路由、限流熔断等服务治理能力。 Spring Cloud Tencent 已对外开源,更多信息请参见 Spring Cloud Tencent 。



# Spring Cloud Tencent 功能与版本对应列表

功能大项	功能点	功能分组	支持版本号
	服务注册		1.0.0+
	服务发现		1.0.0+
	心跳上报		1.0.0+
	自定义健康检查接口		1.1.4+
服务注册与发现	服务列表		1.1.4+
	<b>夕</b> 注 — 夕 台 — 句	Consul	1.2.0+
	Ул:шУХЖ	Nacos	1.9.0+
	合栽均编:	权重随机	1.0.0+
	[x1C+0+X	哈希环	1.11.0+
	动大配罢兹即与审实	远端配置	1.2.0+
	헤장마티카세기주체	本地配置	1.8.0+
动态配置	默认文件分组读取(bootstrap 和 application)		1.5.0+
	配置变更监听器		1.6.0+
	配置更新配置方式		1.2.0+
服务限流	阳流方式	单机限流	1.0.0+
	סד רעיזואא	分布式限流	1.4.0+
	限流行为	快速失败	1.0.0+



		匀速排队	1.5.0+
		Servlet	1.0.0+
	<b>熙友力式</b>	Reactive	1.1.0+
	标签限流		1.3.0+
	第二代标签解析		1.8.0+
	动态标签解析		1.5.0+
	白中ツ阳本响点。	本地静态配置	1.4.0+
	自走又限流响应	代码SPI	1.11.0+
	<b>外学 第七 齐川 第七</b>	错误码	1.0.0+
	「「「「」」「「」」「「」」「」」「「」」「」」「「」」「」」「」」「」」「」	时延	1.11.0+
		实例级熔断	1.0.0+
	熔断级别	接口级熔断	1.10.0+
甲之杨阳		服务级熔断	1.10.0+
加大之子		Feign	1.0.0+
		RestTemplate	1.6.0+
	触发方式	WebClient	1.11.0+
		SCG	1.11.0+
		Zuul	1.11.3+
		自定义路由	1.0.0+
	路由方式	元数据路由	1.5.0+
		就近路由	1.5.0+
叩々吃山	动态标签解析		1.5.0+
版芳哈田	第二代标签解析		1.8.0+
	SCG 支持服务路由		1.6.0+
	Zuul 支持服务路由		1.7.1+
	spring-retry 支持服务路由		1.8.0+
	Feign 调用增强		1.7.0+
RPC增强	RestTemplate 调用增强		1.11.0+
	WebClient 调用增强		1.11.0+
	SCG 转发增强		1.11.0+
	Zuul 转发增强		1.11.3+
元数据传递	法丽 <del>는 -}</del>	本地配置读取	1.0.0+
	Ⅰ <del>洪</del> 4Ⅹ <i>/</i> □ エレ	代码 SPI 读取	1.0.0+
	传递方式	链路传递	1.0.0+



		单跳传递	1.7.0+
		原始头部传递	1.8.0+
		Servlet	1.0.0+
	接收方式	Reactive	1.0.0+
		Feign	1.0.0+
		RestTemplate	1.0.0+
	传递途径	Zuul	1.1.0+
		SCG	1.1.0+
		WebClient	1.11.0+
	测试环境路由插件		1.7.0+
场景化插件	SCG 流量染色插件		1.7.0+
	其他注册中心适配器插件		1.11.0+
	服务实例列表		1.6.0+
Actuator Endpoint扩	动态配置		1.6.0+
展	服务限流规则		1.6.0+
	服务路由规则		1.7.0+
		Feign	1.7.0+
监控数据上报		RestTemplate	1.7.0+
	采集方式	WebClient	1.11.0+
		SCG	1.11.0+
		Zuul	1.11.3+
	F据方式	主动上报	1.7.1+
	0-1-CART	被动拉取	1.7.1+



# Spring Cloud 版本支持

最近更新时间: 2025-04-15 16:59:52

## 背景

Spring Cloud TSF/Tencent 为实现了 Spring Cloud 标准微服务 SPI 的服务治理组件,因此其不包含服务调用组件相关的功能。换句话说,用户自身的 应用本身可能带有服务调用组件,Spring Cloud TSF/Tencent 能提供基于 Spring Cloud 标准接口的服务调用中的服务治理功能。因为 Spring Cloud TSF/Tencent 实现的是 Spring Cloud 定义的标准接口,而开源的 Spring Cloud 有不同版本的提供,因此 Spring Cloud TSF/Tencent 针对不同版 本的 Spring Cloud 也需要对应的支持。

与此同时,Spring Cloud TSF/Tencent 也存在自身的版本规划,以便用户根据自身应用和北极星服务端版本,选择对应的 Spring Cloud TSF/Tencent 版本。所以需要该文档指定 Spring Cloud TSF/Tencent 与 Spring Cloud 的版本对应关系。

# Spring Cloud TSF/Tencent 版本适配规范

Spring Cloud TSF/Tecnent 每1-2年会发布一个大版本(前2位版本号),例如: 2022年发布 Spring Cloud TSF 1.46, 2024年发布 Spring Cloud Tencent 2.0。Spring Cloud TSF/Tencent 对 Spring Cloud 开源版本的适配策略:

- 默认适配发布年份前两年的 Spring Cloud 开源版本,例如: 2024年发布的 Spring Cloud Tencent 2.0 默认适配 Spring Cloud 2022。
- 如有必要,最早适配发布年份前五年的 Spring Cloud 开源版本,例如: 2024年发布的 Spring Cloud Tencent 2.0 最早适配 Spring Cloud Hoxton。
- 在每个版本 EOS 之前(不含 EOS 当年),按需适配上一年到发布年份的 Spring Cloud 开源版本。

## Spring Cloud TSF/Tencent 与服务端版本适配

- 1.x: 支持 TSF-Consul。
- 2.x: 支持北极星和 TSF-Consul。

## Spring Cloud TSF/Tencent 版本适配列表

## Spring Cloud TSF 1.46

Spring Cloud 版本	2022年	2023年	2024年	2025年(EOS)
Finchley	按需适配	按需适配	按需适配	按需适配
Greenwich	按需适配	按需适配	按需适配	按需适配
Hoxton	按需适配	按需适配	按需适配	按需适配
2020	默认适配	默认适配	默认适配	默认适配
2021	按需适配	按需适配	按需适配	按需适配
2022	_	按需适配	按需适配	按需适配
2023	_	_	按需适配	按需适配
2024	_	_	_	不适配

## Spring Cloud Tencent 2.0

Spring Cloud 版本	2024年	2025年	2026年	2027年(EOS)
Hoxton	按需适配	按需适配	按需适配	按需适配
2020	按需适配	按需适配	按需适配	按需适配
2021	按需适配	按需适配	按需适配	按需适配
2022	默认适配	默认适配	默认适配	默认适配
2023	按需适配	按需适配	按需适配	按需适配





2024	-	按需适配	按需适配	按需适配
2025(如有)	-	-	按需适配	按需适配
2026(如有)	-	_	_	不适配

# Spring Cloud TSF/Tencent 版本号规范

# Spring Cloud TSF

Spring Cloud TSF 采用3位版本号,即 a.b.c-xyz-RELEASE,其中每位版本号含义如下:

- a.b: 和集成的北极星服务端的前两位版本号保持一致。
- C:需求开发和缺陷修复版本号。如果有需求开发或缺陷修复,增加此版本号。

• xyz: 和集成的 Spring Cloud 大版本号保持一致。

示例:

- ●版本号 1.46.11-SpringCloud2021-RELEASE 表示对接 1.46.x版本的 TSF Consul 服务端和 Spring Cloud 2021版本。
- 如果 Spring Cloud TSF 有需求开发或缺陷修复,TSF Consul 服务端没有变化,版本号变为 1.46.12-SpringCloud2021-RELEASE。

# **Spring Cloud Tencent**

Spring Cloud Tencent 采用4位版本号,即 a.b.c.d-xyz,其中每位版本号含义如下:

- a.b: 和集成的北极星服务端的前两位版本号保持一致。
- C: 需求开发版本号。如果有需求开发,增加此版本号。
- d: 缺陷修复版本号。如果仅有缺陷修复,增加此版本号。

• xyz: 和集成的 Spring Cloud 版本号保持一致。

示例:

- 版本号 2.0.0.0-2022.0.5 表示对接 2.0.x.x 版本的北极星服务端和 Spring Cloud 2022.0.5 版本。
- 如果 Spring Cloud Tencent 有需求开发,北极星服务端没有变化,版本号变为 2.0.1.0-2022.0.5。
- 如果 Spring Cloud Tencent 有缺陷修复,北极星服务端没有变化,版本号变为 2.0.1.1-2022.0.5。
- 如果 Spring Cloud Tencent 没有变化,北极星服务端需求开发版本号变化:版本号变为 2.0.2.0-2022.0.5。
- 如果 Spring Cloud Tencent 没有变化,北极星服务端缺陷修复版本号变化:版本号变为 2.0.2.1-2022.0.5。

# 相关链接

Spring Cloud Tencent 版本管理 Spring Cloud Tencent 版本发布页



# Spring Cloud 接入

最近更新时间: 2025-04-17 16:00:53

## 操作场景

本文通过一个 demo 进行 Spring Cloud 应用接入微服务引擎托管的 PolarisMesh 治理中心的全流程操作演示,帮助您快速了解如何使用北极星网格。

## 前提条件

- 已创建 Polaris 北极星实例,请参见 创建 PolarisMesh 治理中心。
- 下载 Github 的 demo 源码 到本地并解压。
- 本地编译构建打包机器环境已安装了Java JDK、Maven,并且能够访问 Maven 中央库。
- 根据您自身的业务,已准备好业务部署的资源, 虚拟机部署 、 容器化部署 选择其中一种方式即可。
  - 虚拟机部署已创建 CVM 虚拟机,请参见 创建 CVM 虚拟机。
  - 容器化部署已创建 TKE 容器集群,请参见 创建 TKE 集群。

## 操作步骤

- 1. 登录 TSF 控制台。
- 2. 在左侧导航栏选择 Polaris (北极星),进入北极星实例引擎列表页后单击目标实例进入实例信息页。
- 3. 在引擎管理-实例信息页查看访问地址,Spring Cloud 应用访问使用 gRPC 端口(8091):

### Polaris Server访问地址

客户端接入端口	注册发现	8091 (service-grpc)	
	配置管理	8093 (config-grpc)	
	监控上报	9091 (pushgateway)	
OpenAPI 端口	8090 (http)		
协议兼容端口	7779 (I5)	8761 (eureka) 15010 (:	xdsv3)
	15020 (xds)	v2)	
内网负载均衡 🛈	私 子師	网 内 访问控制	操作
	sh-gz gz-	-4( 1 🎤	删除
	添加内网负载	均衡	
公网负载均衡 🛈	- 开启		

### 4. 修改 demo 中的注册中心地址。

#### 4.1 在下载到本地的 demo 源码目录 下,分别找到

spring-cloud-tencent-examples/polaris-discovery-example/discovery-calleeservice/src/main/resources/bootstrap.yml

# 🔗 腾讯云

### 和

spring-cloud-tencent-examples/polaris-discovery-example/discovery-caller-

service/src/main/resources/bootstrap.yml

两个文件。

### 4.2 添加微服务引擎北极星网格地址到项目配置文件中(以

spring-cloud-tencent-examples/polaris-discovery-example/discovery-callee-

service/src/main/resources/bootstrap.yml

## 为例)。

```
server:
  port: 0
spring:
  application:
    name: DiscoveryCallerService
  cloud:
    polaris:
    enabled: true
    address: grpc://{北极星引擎内网地址 IP}:8091
```

### 5. 打包 demo 源码成 Jar 包。

5.1 在 polaris-discovery-example 源码根目录下,打开 cmd 命令,执行 mvn clean package 命令,对项目进行打包编译。

### 5.2 编译成功后,生成如下表所示的2个 Jar 包。

软件包所在目录	软件包名称	说明
polaris-discovery-example/discovery-callee- service/target	discovery-callee-service-\${version}- SNAPSHOT.jar	服务生产者
polaris-discovery-example/discovery-caller- service/target	discovery-caller-service-\${version}- SNAPSHOT.jar	服务消费者

6. 部署 provider 和 consumer 微服务应用,根据您业务实际选择虚拟机部署、容器化部署以及 TEM 部署中的一种部署方式即可。

### 6.1 虚拟机部署

- 上传 Jar 包至 CVM 实例。
- 执行启动命令进行启动:

nohup java -Djava.security.egd=file:/dev/./urandom -jar [jar包名称] &

#### 6.2 容器化部署

○ 编写 dockerfile 生成镜像,参考:

FROM java:8 ADD ddiscovery-caller-service-\${version}-SNAPSHOT.jar /root/app.jar ENTRYPOINT ["java","-Djava.security.egd=file:/dev/./urandom","-jar","/root/app.jar"]

### ○ 通过 TKE 部署并运行镜像

### 7. 确认部署结果。

- 7.1 进入前面提到的北极星实例页面。
- 选择**服务管理 > 服务列表**, 查看微服务 DiscoveryCallerService 和 DiscoveryCalleeService 的实例数量:
- 若实例数量值不为0,则表示已经成功接入微服务引擎。
- 若实例数量为0,或者找不到 DiscoveryCallerService 和 DiscoveryCalleeService 服务名,则表示微服务应用接入微服务引擎失败。

DiscoveryCalleeService	default		1/1
DiscoveryCallerService	default	-	1/1



7.2 调用 consumer 的 HTTP 接口。执行 HTTP 调用,其中\${app.port}替换为 consumer 的监听端口,\${add.address}则替换为 consumer 暴露的地址:

curl -L -X GET 'http://\${add.address}:\${app.port}/echo?value="hello-world'
预期返回值: hello-world



# 注册发现

最近更新时间: 2025-04-17 16:00:53

## 操作场景

本文介绍在本地开发 Java 应用,通过 Spring Cloud 的方式接入 Polaris (北极星)实例,并实现服务注册与发现。

## 前提条件

1. 在开始开发前,请确保您已经参见 下载 Maven下载安装了 Java 和 Maven。

2. 已创建 Polaris (北极星)实例,操作步骤详细参见 引擎管理。

## 操作步骤

为了方便您快速接入,我们为您准备了 Demo 应用, <mark>点击下载</mark> 。

### 步骤1:引入服务注册与发现的依赖

### 1. 引入 spring cloud tencent 依赖

修改应用根目录下的 pom.xml,添加 dependencyManagement :

```
<dependencyManagement>

<dependencies>

<dependency>

<groupId>com.tencent.cloud</groupId>

<artifactId>spring-cloud-tencent-dependencies</artifactId>

<version>${version}</version>

<type>pom</type>

<scope>import</scope>

</dependency>

</dependencies>

</dependencyManagement>
```

### △ 注意:

请根据项目 Spring Boot 和 Spring Framework 的版本,选择合适的 Spring Cloud Tencent 版本。

### 2. 引入 spring cloud tencent starter

```
• 方式一: 只引入 spring-cloud-starter-tencent-polaris-discovery 。
```

• 方式二: 通过 spring-cloud-starter-tencent-all 引入 SCT 所有 starters。

```
<dependency>
    <groupId>com.tencent.cloud</groupId>
    <artifactId>spring-cloud-starter-tencent-all</artifactId>
    <!-- 注意需要指定 type=pom-->
    <type>pom</type>
</dependency>
```

如果使用 Spring Cloud 2020 及以后版本,还需要添加如下依赖:



```
<dependency>
<groupId>org.springframework.cloud</groupId>
<artifactId>spring-cloud-starter-bootstrap</artifactId>
</dependency>
```

## 步骤2:添加北极星配置文件

- 1. 在项目的 main/resources 目录下创建 bootstrap.yml 文件。
- 2. 在 bootstrap.yml 文件中配置应用名、polaris(北极星)服务端地址等信息。服务端地址查看详细参见: 引擎管理 > 客户端访问地址。

```
spring:
    application:
    name: ${application.name}
    cloud:
        polaris:
        enabled: true
        address: grpc://${修改为Polaris 服务地址}:8091
        namespace: default
```

## 步骤3: 启动应用

应用成功启动后,到 北极星控制台 查看服务注册信息。如下图所示:



## 步骤4: 服务调用

在 Spring Cloud 中可通过 RestTemplate 或者 Feign 发起服务调用。

• RestTemplate

只需要在实例化 RestTemplate 的地方加上 @LoadBalanced 注解即可。



• Feign

通过 Feign 框架调用,按照标准的 Feign 方式即可。



# 配置管理

最近更新时间: 2025-04-17 16:00:53

## 操作场景

本文介绍在本地开发 Java 应用,通过 SpringCloud 的方式接入北极星,使用配置管理功能。

## 前提条件

1. 在开始开发前,请确保您已经参见 下载 Maven下载安装了 Java 和 Maven。

2. 已创建了 Polars (北极星) 实例,操作步骤详细参见 引擎管理。

## 操作步骤

## 步骤1: 引入配置管理依赖

### 1.1 引入 spring cloud tencent 依赖

修改应用根目录下的 pom.xml, 添加 dependencyManagement :

### △ 注意:

请根据项目 Spring Boot 和 Spring Framework 的版本,选择合适的 Spring Cloud Tencent 版本。

### 1.2 引入 spring cloud tencent starter

```
• 方式一: 只引入 spring-cloud-starter-tencent-polaris-config 。
```

```
<dependency>
<groupId>com.tencent.cloud</groupId>
<artifactId>spring-cloud-starter-tencent-polaris-config</artifactId>
</dependency>
```

• 方式二: 通过 spring-cloud-starter-tencent-all 引入 sct 所有 starters。

```
<dependency>
    <groupId>com.tencent.cloud</groupId>
    <artifactId>spring-cloud-starter-tencent-all</artifactId>
    <!-- 注意需要指定 type=pom-->
    <type>pom</type>
</dependency>
```

如果使用 Spring Cloud 2021 版本,还需要添加如下依赖:

<dependency>



<proupId>org.springframework.cloud</groupId> <artifactId>spring-cloud-starter-bootstrap</artifactId dependency>

### 步骤2: 增加配置文件

### △ 注意:

- 应用启动强依赖配置,所以 Spring Cloud Config 是在 Bootstrap 阶段加载配置文件,所以需要把北极星相关的配置放在 bootstrap.yml 里。
- bootstrap.yml 核心配置内容为配置北极星服务端地址以及注入的配置文件信息。

### 2.1 配置 Polaris(北极星) 配置中心地址

- 1. 在项目的 main/resources 目录下创建 bootstrap.yml 文件。
- 2. 在 bootstrap.yml 文件中配置应用名、polaris(北极星)服务端地址等信息。服务端地址详细参见: 引擎管理 > 客户端访问地址 。
- 如果北极星配置中心和注册中心是同一套北极星集群,则只需配置 spring.cloud.polaris.address 即可。
- 如果部署了两套北极星集群,分别用于注册中心和配置中心,则需配置 spring.cloud.polaris.address 用于指定注册中心集群的地址,配置 spring.cloud.polaris.config.address 用于指定配置中心的地址。如下所示:

```
spring:
    application:
    name: ${application.name}
    cloud:
        polaris:
        enabled: true
        address: grpc://${修改为 Polaris 服务地址}:8091 # 必填
        namespace: default # 全局 namespace 参数
        config:
        address: grpc://${独立的配置中心}:8093 # 选填,只有在配置中心和注册中心是两个不同的地址时才需要配置
        auto-refresh; true # 选填,当配置发布后,动态刷新 Spring 上下文,默认值为 true
```

### 2.2 注入配置文件

我们推荐的实践教程是在北极星管控端创建一个名为当前应用名 \${spring.application.name} 的配置分组,Spring Cloud Tencent Config 会自动 注入当前应用名分组下的:

- application-\${activeProfile}.properties
- application-\${activeProfile}.yml
- application-\${activeProfile}.yaml
- application.properties
- application.yml
- application.yaml
- bootstrap-\${activeProfile}.properties
- bootstrap-\${activeProfile}.yml
- bootstrap=\${activeProfile}.yaml
- bootstrap.properties
- bootstrap.yml
- bootstrap.yaml

#### 🕛 说明:

加载顺序依次从上到下,即越先加载的配置,优先级越高。与此同时,远端配置优先级大于本地配置优先级。优先级高的配置覆盖优先级低的配置。

自动注入以上配置文件符合 Spring Boot 的规范,能够满足绝大部分应用场景了。只有当您需要注入额外自定义的配置文件时,才需要在 yml 里配置 spring.cloud.polaris.config.groups ,如下所示:



spring:
cloud:
polaris:
config:
groups:
- name: \${spring.application.name} <b># 选填,注入自定义配置的配置分组</b>
files: [ "config/application.properties", "config/bootstrap.yml" ] # 注入自定义配置文件列表,当
key 冲突时,排在前面的配置文件优先级高于后面

## 步骤3: 在代码中使用配置

用户可通过两种方式更新代码中的配置信息:使用配置类 @ConfigurationProperties 和 @Value 注解。

- @Value 比较适用于配置比较少的场景
- @ConfigurationProperties 则更适用于有较多配置的情况
- 1. 通过 @Value 注入

@Value("\${timeout:1000}")
private int timeout;

2. 通过 @ConfigurationProperties 注入

```
@Component
@ConfigurationProperties(prefix = "teacher")
@RefreshScope //如果使用反射模式,则不需要加这个注解
public class Person {
    private String name;
    private int age;
    String getName() {
        return name;
    }
    void setName(String name) {
        this.name = name;
    }
    int getAge() {
        return age;
    }
    void setAge(int age) {
        this.age = nage;
    }
    void setAge(int age) {
        this.age = age;
    }
    @Override
    public String toString() {
        return "User{" + "name='" + name + '\'' + ", age=" + age + '}';
    }
}.
```

### 步骤4: 在北极星控制台添加配置文件

1. 创建命名空间



命名空间为北极星里核心的概念,通过命名空间逻辑隔离资源,例如通常用于标识不同的环境、不同的集群。

# ▲ 注意:

这里的命名空间需要和应用配置中的命名空间保持一致。

#### 2. 创建配置文件分组

北极星的配置文件分组概念为一组配置文件的集合,推荐把应用名作为一个分组名,例如在我们的示例中,新建一个 polaris-config-example 的分 组。把 polaris-config-example 应用的配置文件都放在 polaris-config-example 分组下,这样便于配置管理。

#### 3. 创建并发布配置文件

北极星配置中心的控制台,配置文件名可以通过"/"来按树状目录结构展示,树状结构可以清晰的管理配置文件。例如一个应用下分不同的模块,每个模 块都有独立的一组配置文件,则可以创建 module1/file1.properties, module1/file2.yaml, module2/file3.yaml。

### ▲ 注意:

配置文件名强烈建议带文件后缀,例如 .properties .yaml .yml .json 等。因为客户端会通过文件名后缀来解析文件内容,如果客户端发现不认 识的后缀名则默认当做 .properties 文件处理。

#### 配置好的实例如下图所示:

← test_c	onfig(default)					
配置文件	配置版本 发布历史					
	新增					
	请输入文件名搜索	Q	app2.json			查看发布历史 🖸
	app2.json		状态 发布成功	最后修改人	10 3	
	app1.json		最后修改时间 2024-01-08 16:17:39	最后发布人	10 3	
			最后发布时间 2024-01-08 16:17:55	备注	-	
			标签 -	格式	json	
			发布 编辑			
						53
			1 adc:123			

### 步骤5: 启动应用

到此接入 Spring Cloud Tencent Config 即已完成。

### 步骤6: 动态刷新配置

#### 🕛 说明:

- Spring Cloud Tencent 1.7.0 之前的版本通过 Spring Cloud 标准的 @RefreshScope 实现配置动态刷新。 @RefreshScope 存在以下局限性:
  - 每次刷新配置,需要重新构建整个 Spring Context 重建 Bean。
  - 代码里层面需要引入 @RefreshScope 注解。
- 1.7.1 及之后版本, Spring Cloud Tencent 优化了动态刷新的实现方式, 建议您使用最新版本。

### 6.1 刷新 @Value 属性

应用启动时扫描所有的 Bean,构建 <sup>@Value</sup> 属性对应的 Bean 的映射关系。当属性配置更新时,找到所有的待更新的 Bean 并利用 Java 的反射机制更 新属性。

### 6.2 刷新 @ConfigurationProperties 配置类

相比于 @RefreshScope 重建所有的 Bean,优化之后的刷新机制只需要重建更新的配置对应的 @ConfigurationProperties Bean ,影响面更小。 实现原理请看 PolarisRefreshAffectedContextRefresher 和 AffectedConfigurationPropertiesRebinder 。 如果您想使用传统的 @RefreshScope 方式刷新配置,可以配置 spring.cloud.polaris.config.refresh-type=refresh\_context 。


## 6.3 关闭动态刷新能力

 ${f \epsilon}$  bootstrap.yml  ${f ERE}$  spring.cloud.polaris.config.auto-refresh=false .

# 服务限流



最近更新时间: 2025-01-15 12:06:52

## 操作场景

**服务限流是最常见的一种服务自我保护措施之一,防止流量洪峰打垮服务。**Spring Cloud Tencent Rate Limit **模块内置了针对** Spring Web 和 Spring WebFlux 场景的限流 Filter,结合北极星的限流功能帮忙业务快速接入限流能力。

## 前提条件

1. 在开始开发前,请确保您已经参见 下载 Maven下载安装了 Java 和 Maven。

2. 已创建 TSE 治理中心(北极星网格)实例,操作步骤详细参见 引擎管理。

## 操作步骤

## 步骤1: 引入依赖

## 1.1 引入 spring cloud tencent 依赖

修改应用根目录下的 pom.xml,添加 dependencyManagement :



## ▲ 注意:

请根据项目 Spring Boot 和 Spring Framework 的版本,选择合适的 Spring Cloud Tencent 版本。

#### 1.2 引入 spring cloud tencent starter

```
• 方式一: 只引入 spring-cloud-starter-tencent-polaris-ratelimit。
```



• 方式二: 通过 spring-cloud-starter-tencent-all 引入 sct 所有 starters。

如果使用 Spring Cloud 2020 及以后版本,还需要添加如下依赖:

```
<dependency>
<groupId>org.springframework.cloud</groupId>
<artifactId>spring-cloud-starter-bootstrap</artifactId>
```



## </dependency>

## 步骤2: 增加配置文件

- 1. 在项目的 main/resources 目录下创建 bootstrap.yml 文件。
- 2. 在 bootstrap.yml 文件中配置应用名、polaris (北极星) 服务端地址等信息。服务端地址详细参见: 引擎管理 > 客户端访问地址。

spring:
application:
<pre>name: \${application.name}</pre>
cloud:
polaris:
enabled: true
address: grpc://\${ <b>修改为第一步部署的</b> Polaris <b>服务地址</b> }:8091
namespace: default

## 步骤3:控制台配置限流规则

#### 请参见 服务治理指南中服务治理 > 服务限流 相关文档,配置限流规则,相关界面如下:

限意规则名称•
显长64个字符
現說类型• <b>单机限流</b> 分布式限流
限流现则详情
目标服务您可以对目标服务的指定接口设置限流规则。当该接口被调用时,符合匹配规则的请求、则会触发限流规则
命名空间・ 请选择命名空间 ▼
而冬安街•
· 通知名称 · 斯尔全部 · 斯尔全部 · <b>至比此 · ▼</b>
请求匹配规则 ① 自定义 ▼ 请输入Key值 全匹配 ▼ 请输入Value值 ×
限流规则
<b>環議条件</b> 满足以下任意条件即可触发限流
段流调值 统计窗口时长 请求数调值
+ 液加 删除所有
合并计算阈值 🕢 如果目标请求匹配到多个接口及参数。则将匹配到的所有请求汇合、合并计算编值、具体规则查看
<b>院高万条</b> 佛龙斑虎骶灰条件员的处理万条
限溫效果 快速失败 匀速排队
是否命用

## 步骤4:启动应用

应用启动成功后,访问应用的 HTTP 接口,测试限流是否生效。如果请求被限流将会返回以下内容:

The request is deny by rate limit because the throttling threshold is reached



# 服务熔断

最近更新时间: 2025-04-17 16:00:53

## 熔断场景

Spring Cloud Tencent CircuitBreaker 结合 Polaris 的熔断能力提供了标准的熔断实现,并提供服务级、接口级、节点级三种熔断级别,可以根据需 求组合搭配,来实现期望的熔断效果。

本文介绍在本地开发 Java 应用,通过 Spring Cloud 的方式接入 Polaris(北极星),使用服务熔断功能。

## 前提条件

1. 在开始开发前,请确保您已经参见 下载 Maven下载安装了 Java 和 Maven。

2. 已创建 Polaris (北极星)实例,操作步骤详情参见 引擎管理。

#### 操作步骤

## 步骤1: 引入依赖

#### 1.1 引入 spring cloud tencent 依赖

修改应用根目录下的 pom.xml,添加 dependencyManagement :

#### △ 注意:

- 请根据项目 Spring Boot 和 Spring Framework 的版本,选择合适的 Spring Cloud Tencent 版本。
- 服务级熔断支持版本号大于1.10.0的 Spring Cloud Tencent 版本。

## 1.2 引入 spring cloud tencent starter

#### 🕛 说明:

因为熔断能力依赖服务发现能力,所以需要主调方和被调方都引入服务注册与发现依赖。

• 方式一: 只引入 spring-cloud-starter-tencent-polaris-circuitbreaker 。

```
<dependency>
```

<groupId>com.tencent.cloud</groupId>

<artifactId>spring-cloud-starter-tencent-polaris-circuitbreaker</artifactId>

</dependency

• 方式二: 通过 spring-cloud-starter-tencent-all 引入 sct 所有 starters。

dependency>

- <groupId>com.tencent.cloud</groupId>
- <artifactId>spring-cloud-starter-tencent-all</artifactId>



**F** = 1 = 1 = 1

<!-- **注意需要指定** type=pom--> <type>pom</type> pendency>

如果使用 Spring Cloud 2020 及以后版本,还需要添加如下依赖:

## 步骤2: 增加相关配置文件并设置熔断方法

Spring Cloud Tencent 支持 Feign、RestTemplate、WebClient、Spring Cloud Gateway 四种组件的熔断,并会自动拉取您在北极星上为被调 服务配置的熔断规则进行熔断。选择您适合的方式配置即可。

ге	ign
完 1.	整代码可查看 Demo: 添加 Feign 依赖
	<dependency> <groupid>org.springframework.cloud</groupid> <artifactid>spring-cloud-starter-openfeign</artifactid> </dependency>
2.	添加 Feign 配置文件
	spring: application: name: polaris-circuitbreaker-feign-example

```
openfeign:
circuitbreaker:
enabled: true
polaris:
address: grpc://${修改为 Polaris 服务地址}:8091
namespace: default
enabled: true
loadbalancer:
enabled: true
circuitbreaker:
enabled: true
```

#### 3. 设置 Feign 熔断方法

○ 声明 Feign 调用

当 Feign fallback 不配置时,会在熔断发生时,从 polaris server 拉取降级配置,进行降级,降级的 response body 会序列化成您在 Feign 中配置的返回对象,请确保能正确的序列化。







○ 声明本地 Feign 熔断方法(可选)

```
@Component
public class ProviderBFallback implements ProviderB {
    @Override
    public String info() {
        return "fallback: trigger the refuse for service b";
    }
}
```

○ 发起 Feign 调用

```
@RestController
@RequestMapping("/example/service/a")
public class ServiceAController {
    @Autowired
    private ProviderB polarisServiceB;
    /**
    * Get info of Service B by Feign.
    * @return info of Service B
    */
    @GetMapping("/getBServiceInfo")
    public String getBServiceInfo() {
        return polarisServiceB.info();
    }
}
```

RestTemplate

#### 完整代码可查看 Demo:

#### 1. 添加 RestTemplate 配置文件

```
spring:
    application:
    name: polaris-circuitbreaker-resttemplate-example
    cloud:
    polaris:
        address: grpc://${修改为 Polaris 服务地址}:8091
        namespace: default
        enabled: true
        loadbalancer:
        enabled: true
        circuitbreaker:
```



#### enabled: true

#### 2. 设置 RestTemplate 熔断方法

设置 RestTemplate bean, 添加 @PolarisCircuitBreaker 注解。



```
ekequestMapping('/example/service/a')
public class ServiceAController {
    @Autowired
    private RestTemplate restTemplate;
    @GetMapping("/getBServiceInfo")
    public String getBServiceInfo() {
        return restTemplateFallbackFromPolaris.getForEntity("/example/service/b/info",
    String.class);
    }
}
```

## WebClient

#### 完整代码可查看 demo:

1. 添加 WebFlux 依赖

#### 2. 设置 WebClient 配置文件



```
spring:
    application:
    name: polaris-circuitbreaker-webclient-example
    cloud:
    polaris:
        address: grpc://${修改为 Polaris 服务地址}:8091
        namespace: default
        enabled: true
        loadbalancer:
        enabled: true
        circuitbreaker:
        enabled: true
        circuitbreaker:
        enabled: true
        circuitbreaker:
        enabled: true
        circuitbreaker:
        enabled: true
        o 设置 WebClient 熔断方法
        O 设置 WebClient bean
```

# @LoadBalanced @Bean WebClient.Builder webClientBuilder() { return WebClient.builder() .baseUrl("http://polaris-circuitbreaker-callee-service"); }

#### ○ 发起 WebClient 调用



Spring Cloud Gateway

#### 完整代码可查看 demo:

1. 添加 Spring Cloud Gateway 依赖

#### 2. 设置 Spring Cloud Gateway 配置文件

○ 以 gateway.discovery.locator 形式配置

```
○ 以普通路由形式配置形式配置
```

```
spring:
application:
  name: GatewayScgService
cloud:
  polaris:
    address: grpc://${修改为 Polaris 服务地址}:8091
    namespace: default
    enabled: true
  gateway:
```





## 步骤3:验证

启动应用并在 北极星控制台 设置被调服务的服务级熔断规则;请参见 服务治理指南 > 服务治理 > 熔断降级 > 服务熔断与主动探测 相关文档,相关界面如 下:

炮断降级									
服务级增新	节点级编断	主动探测							
BROSER								诸选择条件进行过渡	Q
10/规则名			RD T	主调服务	被调服务	创建时间/推改时间	应用时间	Shift	
* test			EAR	命名空间:" 服务:"	命名空间: defauit 服务: polaris-circuitbreaker- calies-service 第口:	2023-02-20 23:11:21 2023-03-13 15:18:42	2023-03-13 15:18:42	57/H 1618 (818	
HEE									
相误利用条件	返回码不够于200								
1012	连续错误数>=1个								
的新拓度	服务								
10/01/16	6019								
55,52,52,55	当课程9个连续成功请	1水后恢复							
主动原则	开启								
自定义纳应	未开启								
共1条							10 * 张/页	H - 1 /13	ж

- •为了验证熔断能力,被调端至少启动两个实例,一个返回成功,一个返回失败。
- 测试主调方向被调方发起服务调用,前几个请求会出现往有问题的服务发起调用,当问题数达到您在北极星控制台配置的熔断阈值时,即会触发熔断,主调 方不再发起调用,直接返回熔断降级方法。



# 动态路由

最近更新时间: 2025-04-17 16:00:53

## 操作场景

本文介绍在本地开发 Java 应用,通过 Spring Cloud 的方式接入 Polaris(北极星),并实现服务路由。

## 前提条件

1. 在开发前,请确保您已经参见 下载 Maven下载安装了 Java 和 Maven。

2. 已创建 Polaris (北极星)实例,操作步骤详见 引擎管理。

## 操作步骤

## 步骤1: 引入依赖

## 1.1 引入 spring cloud tencent 依赖

修改应用根目录下的 pom.xml, 添加 dependencyManagement :

## ▲ 注意:

- 请根据项目 Spring Boot 和 Spring Framework 的版本,选择合适的 Spring Cloud Tencent 版本。
- 服务路由在 1.5.0版本之后,提供了完善的服务路由能力。所以版本号需大于1.5.0。

## 1.2 引入 spring cloud tencent starter

```
• 方式一: 同时引入 spring-cloud-starter-tencent-polaris-discovery 和 <math>spring-cloud-starter-tencent-polaris-router 。
```

```
<!-- 注意需要指定 type=pom-->
```



dependency>

如果使用 Spring Cloud 2020 及以后版本,还需要添加如下依赖:

## 步骤2:添加北极星配置文件

- 1. 在项目的 main/resources 目录下创建 bootstrap.yml 文件。
- 2. 在 bootstrap.yml 文件中配置应用名、polaris (北极星)服务端地址等信息。服务端地址查看详见 引擎管理 的客户端访问地址部分。



## 步骤3:路由规则下发

启动应用并在 北极星控制台 设置动态路由规则,操作步骤详细参见 服务治理指南 > 服务治理 > 动态路由,相关界面如下:

← 新建服务路由规则					
10.01/2.53					
发现中的。 最长64个字符					
描述					
优先级 ①・ 📃 🛛 🛛	+				
匹配条件	主调服务	<del>()</del>	→ 被调服务		
主调	请求按照匹配规则匹配成功后,将按照当前规则进行	<sup>一</sup> 目标服务路由 点击切换主被调服务	请求会按照规则路由到目标服务分组		
命名空间。	全部命名空间	▼ 命名3	间• 请选择命名空间	*	
服务名称	全部服务	服务者	称•		
路由策略 按照以下规则顺序进行匹配, 7	J拖动调整顺序。若规则全未匹配上,则请求报 	巨绝。			+ X
来源服务的请求满足以	下匹配条件				
自定义	▼ 请输入标签键		<b>等于 ▼</b> 请输入标签值		×
+ 添加					
将转发至目标服务的以	下实例分组,按优先级转发,可拖动调!	整优先级:			
Ⅲ 第【 <b>0】</b> 优先级 (1	先级常用于容灾场景,高优先级实例故障后,	路由至次优先级实例。相同优先级下	分组权重加和为100。		×
实例分组名称	权重 ()	是否隔离 实例标签			
分组1	100	+			×
+添加实例分组					
1 20 to No 40 H- 40					
〒 湯川山八九元3級					
+ 添加规则					



# 全链路灰度

最近更新时间: 2025-04-17 16:00:53

## 操作场景

本文介绍在本地开发 Java 应用,通过 Spring Cloud 的方式接入 Polaris(北极星),并实现全链路灰度能力。

## 前提条件

1. 在开始开发前,请确保您已经参见 下载 Maven下载安装了 Java 和 Maven。

2. 已创建 Polaris (北极星)实例,操作步骤详细参见 引擎管理。

## 操作步骤

为了方便您快速接入,我们为您准备了 Demo 应用, <mark>点击下载</mark> 。

## 步骤1: 引入服务注册与发现的依赖

## 1. 引入 spring cloud tencent 依赖

修改应用根目录下的 pom.xml,添加 dependencyManagement :

```
<dependencyManagement>

<dependencies>

<dependency>

<groupId>com.tencent.cloud</groupId>

<artifactId>spring-cloud-tencent-dependencies</artifactId>

<version>${version}</version>

<type>pom</type>

<scope>import</scope>

</dependency>

</dependencies>

</dependencyManagement>
```

#### △ 注意:

请根据项目 Spring Boot 和 Spring Framework 的版本,选择合适的 Spring Cloud Tencent 版本。

#### 2. 引入 spring cloud tencent starter

• 方式一: 同时引入 spring-cloud-starter-tencent-polaris-discovery 和 <math>spring-cloud-starter-tencent-polaris-router 。





'dependency>

如果使用 Spring Cloud 2020 及以后版本,还需要添加如下依赖:

## 步骤2:添加北极星配置文件

- 1. 在项目的 main/resources 目录下创建 bootstrap.yml 文件。
- 2. 在 bootstrap.yml 文件中配置应用名、polaris (北极星)服务端地址等信息。服务端地址详细参见: 引擎管理 > 客户端访问地址。



## 步骤3: 启动应用

应用成功启动后,到 北极星控制台 查看服务注册信息。如下图所示:

~	POLINISPESI											
86	教治部中心	+ public-service-polaris	gitewayigened)									
	1218	BARN BARN S	NUL RIGHT AREA									
0	SATIS.	1.9.9	80 0	82	16.0	88115 88	- BRID 5	· ·				
۲	#452H											
6.00	190		VI IN ACTO									0
=	ACRE IN CO.	C RBP	80	92	<b>三</b> キ	6.8	120H5	RRKD	108709	\$52319	8.9	
₿	****	+ 1837/KL144	80	inte		100		+16.4	2822-05-21 17.40-25	2522-65-31 17:40:05	/ 宜	
100	291	818								10 × 0 / 11	+ + 1 218 I	
=	8092											
8	\$\$1212											
22	#22.22											

#### 步骤4: 控制台配置全链路灰度规则

完成客户端接入后,您可在北极星控制台配置您全链路灰度的规则,详情请参见: 全链路灰度配置。



# 无损上下线

最近更新时间: 2025-04-17 16:00:53

## 使用场景

微服务发布过程中,可能会因为服务的变更造成流量的错误或中断。Spring Cloud Tencent 提供了插件来实现无损上下线,其原理为:

- 无损上线:在服务启动后,等待服务完全就绪后再注册到注册中心(无配置健康探测接口时为延迟注册),并对外提供服务。然后再结合 K8s 生命周期, 滚动更新下一个节点。
- 无损下线:在服务停止前,先从注册中心注销,拒绝新的请求,等待旧的请求处理完毕后再下线服务。

## 无损上线

Polaris (北极星)支持以下2种方式支持服务无损上线:服务就绪/延迟注册和服务注册就绪检查。

## 方案一: 服务就绪/延迟注册

在一些场景下,服务支持延迟加载,启动后异步加载一些资源,例如:服务需要从文件存储 COS 获取数据或者文件,待数据或者文件拉取完成后才能对外提 供服务。如果在应用启动后直接注册服务,会导致服务实际没有就绪而调用失败。因此通过确保服务就绪之后,再进行注册从而对外提供服务,可以确保服务 平滑、无损上线。

北极星支持以下两种服务就绪/延迟注册场景:

- 场景一: 业务暴露健康检查接口,接口探测成功后,服务注册。
- 场景二: 业务没有暴露健康检查接口,则延迟一段时间服务注册,默认延迟注册时长30s,可通过配置自定义延迟注册的时长。



#### 操作步骤

**步骤1**:服务通过 Spring Cloud Tencent 接入北极星服务注册与发现能力。具体步骤请参见:Spring Cloud Tencent 接入北极星。 步骤2:引入 spring cloud tencent 无损上下线插件依赖。 在 pom.xml 中添加依赖:

<dependencies></dependencies>			
<dependency></dependency>			
<groupid>com.tencent.c</groupid>	loud <th></th> <th></th>		
<artifactid>spring-clc</artifactid>	ud-tencent-lo	ossless-plug:	in
步骤3:在应用北极星配置文件中添加无损。	上下线相关配置项	。具体参数如下	:
配置项 Key	默认值	是否必填	配置项说明



spring.cloud.polaris.lossless. enabled	false	否	无损上下线开关。
spring.cloud.polaris.lossless. health-check-path	无	否	业务程序健康探测接口。
spring.cloud.polaris.lossless. delay-register-interval	30000	否	无配置业务程序健康探测接口,延迟注册时间。默认 30000(单位ms)。
spring.cloud.polaris.lossless. health-check-interval	5000	否	配置业务程序健康探测接口后,健康探测间隔。默认 5000(单位ms)。
spring.cloud.polaris.admin.ho st	0.0.0.0	否	无损上下线服务绑定的 IP 地址。
spring.cloud.polaris.admin.po rt	28080	否	无损上下线服务绑定的端口号。

## 配置示例:

在项目的 main/resources 目录下 bootstrap.yml 文件配置:

```
spring:
    application:
    name: ${application.name}
    cloud:
        polaris:
        enabled: true
        address: grpc://${修改为Polaris 服务地址}:8091
        namespace: default
        # 以下为无损上下线的配置
        lossless:
        enabled: true
        health-check-path: /
        delay-register-interval: 300000
        health-check-interval: 10000
        admin:
        host: 0.0.0.0
        port: 28080
```

## 方案二: 服务注册就绪检查

通常 K8s 提供就绪检查机制来对实例在就绪前进行健康检查,它一般会简单地认为端口起来了应用即处于就绪状态。但是实际上,应用端口起来了和服务成 功注册之间存在 gap,这样会造成服务没有成功注册,旧的应用实例就被下线,下一个节点开始部署。最终导致消费端调用异常。 TSE 北极星提供服务注册状态的接口和端口,以配合 K8s 就绪检查,当应用注册完成返回200状态码,帮助 K8s 判定应用已就绪;未完成注册返回500状 态码,帮助 K8s 判定应用未就绪。尤其在实例滚动更新的过程中,等实例就绪后才滚动更新下一个节点。

#### 操作步骤:

步骤1:服务通过 Spring Cloud Tencent 接入北极星服务注册与发现能力。具体步骤参见: Spring Cloud Tencent 接入北极星。 步骤2:引入 spring cloud tencent 无损上下线插件依赖,并添加无损上下线相关配置。具体参见 方案一。 步骤3:在容器服务 TKE 等 K8s 应用部署平台中配置就绪检查,如下图所示。

- 路径: /online。
- 端口: 28080。



容器健康检查()	<u> </u>	存活检查 检查容器是召 就绪检查 检查容器是召	5正常,不正常则重启实例 5就绪,不就绪则停止转发流量到当前实例
		检查方法	HTTP请求检查
		检查协议	HTTP *
		host	默认为 Pod IP, 一般不需要修改
			大多数情况下不需要填 host 字段,请谨慎填写防止探测失败,更多请参考文档 🗹
		检查端口	28080
		检查端口	28080 端口范围: 1~65535,支持使用端口名
		检查端口 httpHeader	28080 端口范围: 1~66535, 支持使用端口名
		检查端口 httpHeader 请求路径	28080 端口范围: 1~65535, 支持使用端口名
		检查端口 httpHeader 请求路径 启动延时①	28080 端口范围: 1-65535,支持使用端口名 /online /時礼名忠政証 <b>秒</b> 启动延时最小值为0秒, 默认为不设置

## 无损下线

应用在滚动发布或者下线过程中,被调方服务实例向注册中心发起反注册,主调方从注册中心更新 IP 的过程中,存在时间 gap,导致依然有可能调用到已经 下线的实例,从而调用失败。

TSE 北极星提供服务无损下线的接口: /offline,结合 K8s 的生命周期,实现服务无损下线。整体流程如下:



#### 操作步骤:

步骤1: 服务通过 Spring Cloud Tencent 接入北极星服务注册与发现能力。具体步骤参见: Spring Cloud Tencent 接入北极星。

步骤2:引入 spring cloud tencent 无损上下线插件依赖,并添加无损上下线相关配置。具体参见方案一。

步骤3:在容器服务 TKE 等 K8s 应用部署平台中配置 preStop 生命周期检查。

preStop 配置检查指令: curl -X PUT http://localhost:28080/offline && sleep 20

# 使用 Java Agent Java Agent 概述

最近更新时间: 2025-06-11 15:55:02

## 功能概述

Spring Cloud Tencent 支持 Java Agent 的方式进行接入。对于 Spring Cloud 应用(原生 Spring Cloud 应用或者其他 Spring Cloud 扩展的应用),可以不用对应用代码及依赖进行任何变更的情况下,即可使用 Spring Cloud Tencent 的功能。

## Spring Cloud 各版本及功能在 Java Agent 上的支持情况

## () 说明:

使用 Java Agent 的方式进行接入时,只支持 Open JDK 和 Kona JDK,不支持 Oracle JDK。

市能十米	市台市	Spring Cloud 2023	Spring Cloud 2022	Spring Cloud Hoxton
功能入关	和日日本	JDK 17	JDK 17	JDK 8
lava Agont注入	虚拟机注入	支持	支持	支持
	容器化注入	支持	支持	支持
注册中心	服务注册	支持	支持	支持
	服务发现	支持	支持	支持
	健康检查(心跳上 报)	支持	支持	支持
	Nacos&北极星服务 双注册	不支持	不支持	不支持
	动态路由	支持	支持	支持
	就近路由	支持	支持	支持
流量调度	服务限流	不支持	支持	支持
	无损上下线	支持	支持	支持
	全链路灰度	支持	支持	支持
	故障熔断	不支持	支持	支持
故障容错	主动探测	不支持	支持	支持
	自定义降级	不支持	支持	支持
配置管理	配置文件订阅	支持	支持	支持



# 使用 Java Agent 接入

最近更新时间: 2025-06-11 15:55:02

## 功能概述

Spring Cloud Tencent 的 Java Agent 支持针对容器部署的应用以及虚拟机部署的应用进行注入。对于 Spring Cloud 应用(原生 Spring Cloud 应 用或者其他 Spring Cloud 扩展的应用),可以不用对应用代码及依赖进行任何变更的情况下,即可使用 Spring Cloud Tencent 的功能。下文将分别对 容器场景和虚机场景的使用方法进行说明。

## 前提条件

- 已创建 Polaris (北极星)引擎实例。
- 登录 TSF 控制台。
- 在服务治理中心下的实例列表页面,单击目标引擎的 ID,进入基本信息页面。

## 使用步骤

- 1. 登录 TSF 控制台。
- 2. 在左侧导航栏选择Polaris (北极星)后,进入北极星引擎实例列表页。
- 3. 单击目标引擎的 ID,进入基本信息页面。

#### 容器场景

## 步骤1: 安装 polaris-controller

#### () 说明:

- 1. 添加容器集群会安装 Polaris-Controller 组件,请确保您的集群至少有 1C2G 的空闲资源,否则会添加失败。
- 2. 容器集群中安装 Polaris-Controller 组件后默认支持 K8s Service 服务同步(按需同步)和 Java Agent 自动注入能力。您也可以在 控制台选择**全量同步**使用 k8s 全量同步方式。
- 1. 在目标引擎基本信息页顶部页签单击 K8s 集群,单击关联集群,单击查看关于 K8s 集群关联到 Polaris 的相关介绍。
- 2. 根据自身业务需求选择目标关联的 K8s 集群,支持 TKE (容器集群) / EKS (弹性容器集群)。即在目标集群中完成 polaris-controller 的安装。
- 3. K8s Service 同步接入:
- 全量同步:集群内全部 service 自动同步至北极星。

<ol> <li>添加容器 容器集群 以选择下</li> </ol>	集群会安装Polaris-Controller组件,请确保您的集群至少有1C2G的空闲资源,否则会添加失败 中安装Polaris-Controller组件后默认支持K8s Service服务同步(按需同步)和Java Agent自动注入能力。 您也可 方"全量同步"按钮使用k8s 全量同步方式。
集群 🛈 🔸	tke         ✓         ●
K8s Service 同步	按需同步 全量同步 集群内全部 service 自动同步至北极星。您可根据需要添加 annotation,使用指引 [2]
Java Agent接入	用户可以通过添加注解的方式,在 Spring Cloud 应用里注入 Java Agent。 Java Agent 自动引入 Spring Cloud Tencent 依赖,接入北极星注册配置治理中心。 <b>使用指引 2</b>

• 按需同步: 仅指定的 namespace 或者 service 自动同步至北极星。



<ol> <li>添加容器 容器集群 以选择下</li> </ol>	集群会安装Polaris-Control 中安装Polaris-Controller组 方"全量同步"按钮使用k8s :	er组件,请确保您的集群至少 件后默认支持K8s Service服∰ ≧量同步方式。	有1C2G的空闲资源,否则 务同步(按需同步)和Java	会添加失败 a Agent自动注入能力。 您也i	IJ
集群 🛈 🔸	tke 如果现有的网络不合适,	> → → → → → → → → → → → → → → → → → → →	(t ×	0	
K8s Service 同步	<mark>按需同步</mark> 全量 K8s 集群中,仅指定的 n 您需要在指定的 namesp	<b>同步</b> Imespace 或者 service 将自 Ice 或者 service 上添加 ann	动同步至北极星。 otation, <b>使用指引 <sup>[2]</sup></b>		
Java Agent接入	用户可以通过添加注解的 Tencent 依赖,接入北极	方式,在 Spring Cloud 应用 重注册配置治理中心。 <b>使用指</b>	팉注入 Java Agent。 Java 引 ☑	Agent 自动引入 Spring Clou	ıd

## 步骤2:在命名空间开启自动注入功能

• 在目标应用所在的 K8s 命名空间,通过以下命令开启自动注入:

kubectl label namespace <命名空间名> polaris-injection=enabled

• 通过以下命令查询命名空间是否开启了自动注入:

kubectl get	namespace -L	polaris	-injection
输出结果:			
NAME	STATUS	AGE	POLARIS-INJECTION
default	Active	3d2h	enabled

#### • 注入范围控制列表

label 键	范围	label 值	含义	polaris-controller 版本
polarismesh.cn/inje ct	Pod	disable d	声明当命名空间开启默认注入时,作为黑名单功能 使用,指定 pod 不注入 javaagent	v1.7.3及以上版本
polarismesh.cn/inje ct	Pod	enabled	声明当命名空间未开启默认注入时,作为白名单功 能使用,单独指定 pod 注入 javaagent	v1.7.3及以上版本
polarismesh.cn/inje ct	Nam espa ce	enabled	声明该命名空间下的 pod,默认注入 javaagent	v1.7.3及以上版本
polaris-injection	Nam espa ce	enabled	声明该命名空间下的 pod,默认注入 javaagent	所有版本支持,待废弃,使用 polarismesh.cn/inject 代 替

#### 具体yaml示例

```
apiVersion: v1
kind: Namespace
metadata:
name: java-agent-test
labels:
polarismesh.cn/inject: enab
```

• 黑名单功能,不注入java agent。

apiVersion: apps/v1 kind: Deployment



metadata:	
name: label-block	
namespace: java-agent-test	
spec:	
selector:	
matchLabels:	
app: label-block	
template:	
metadata:	
labels:	
app: label-block	
polarismesh.cn/inject: disabled	
annotations:	
polarismesh.cn/javaagent: "true"	
spec:	

• 白名单功能,命名空间未开启自动注入的情况下,实现指定 pod 按需注入。

```
apiVersion: apps/v1
kind: Deployment
metadata:
   name: label-allow
spec:
   selector:
    matchLabels:
        app: label-allow
template:
        metadata:
        labels:
            app: label-allow
            polarismesh.cn/inject: enabled
        annotations:
            polarismesh.cn/javaagent: "true"
        spec:
        .....
```

## 步骤3: SpringCloud 应用 yaml 文件中声明 Java Agent 标签

• 找到需要注入 Java Agent 的应用(一般是 deployment 或者 statefulset),编辑 yaml 并增加以下标签:

Annotation 键	Annotation值	含义	polaris−controller版 本
polarismesh.cn/j avaagent	支持: "true"、"false",必填	声明需要往这个 POD 中注入 Java A gent	所有版本支持
polarismesh.cn/ workloadNames paceAsServiceN amespace	支持:"true"、"false",缺省	声明使用 k8s 命名空间作为服务命名空 间注册到北极星,若和 polarismesh.cn/javaagentConfig 中的配置冲突,此配置不生效	v1.7.3及以上版本
polarismesh.cn/ workloadNameA sServiceName	支持:"true"、"false",缺省	声明使用 k8s 工作负载名称作为服务名 注册到北极星,若和 polarismesh.cn/javaagentConfig 中的配置冲突,此配置不生效	v1.7.3及以上版本
polarismesh.cn/j avaagentFrame workName	缺省,则自动识别应用框架的类型。 支持:spring-cloud	声明应用的框架类型,对于 SpringCloud 应用则填写 spring-	所有版本支持



		cloud,目前仅支持 SpringCloud 类型。	
polarismesh.cn/j avaagentFrame workVersion	缺省,则自动识别应用框架的版本。 也支持手动声明框架版本,例如: 2023、2022、hoxton 等。	声明应用的框架版本。	所有版本支持
	如:2.0.1.0。		
polarismesh.cn/j avaagentVersio n	♪ 注意: 生产环境请指定验证后的版本, 若不填则自动会在每次实例发生 重启时自动注入最新的版本,可 能会有新版本的兼容性问题,导 致实例启动异常。	声明 java−agent 包的镜像版本。	所有版本支持
polarismesh.cn/j avaagentConfig	如指定命名空间、服务名并配置无损上 线: '{"spring.cloud.polaris.lossless.e nabled":"true","spring.cloud.pola ris.lossless.delay-register- interval":"30000","spring.cloud.p olaris.discovery.namespace":"cu stom- ns","spring.application.name":"c ustom-service"}'	支持用户自定义的 Java Agent 配置, 支持指定命名空间、服务名、服务实例 元数据、无损上线配置等。格式为 JSON。 不填写的配置则使用默认配置。	所有版本支持

## • 配置 yaml 样例:

默认配置: \${spring.application.name}的值作为服务名, default作为北极星命名空间。

```
apiVersion: apps/v1
kind: Deployment
metadata:
    name: default-ns-name
    namespace: java-agent-test
spec:
    selector:
    matchLabels:
        app: default-ns-name
    template:
        metadata:
        labels:
            app: default-ns-name
        annotations:
            polarismesh.cn/javaagent: "true"
    spec:
        .....
```

使用 k8s 工作负载名称作为服务名, k8s 命名空间作为北极星服务命名空间注册。

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: k8s-info-inject
  namespace: java-agent-test
spec:
  selector:
```



matchLabels:
app: k8s-info-inject
template:
metadata:
labels:
app: k8s-info-inject
annotations:
polarismesh.cn/javaagent: "true"
polarismesh.cn/workloadNamespaceAsServiceNamespace: "true"
polarismesh.cn/workloadNameAsServiceName: "true"
spec:

• 使用自定义配置,下面的例子为使用 config 里面的配置,使用 custom-service 作为服务名,custom-ns 作为北极星服务命名空间注册。



修改 yaml 后,需要重新部署应用。重新部署后,会自动完成 java-agent 的注入,可以观察应用日志,如果启动的时候出现以下关键字,则证明 java-agent 注入成功。



虚拟机场景

#### 步骤1:下载 java agent

获取最新的 polaris-java-agent 的版本,并下载二进制 zip 包。点击下载。

步骤2:修改配置文件



1. 解压下载后到的 zip 包,进入 polaris-java-agent 的目录,并修改 conf/polaris-agent.conf 文件,填写需要使用的插件名称(需要按照应用 的 spring-cloud 版本进行填写),内容示例如下:

plugins.enable=spring-cloud-2020-plugin

2. 进入 conf/plugin/spring-cloud-2020(不同插件有不同目录,需要按照上一步填写的插件来选择目录)目录中,修改 application.properties 文件:

2.1 填写正确的应用名(spring.application.name)。

2.2 填写您购买的北极星服务端的地址信息。服务端地址查看详细参见: 引擎管理 > 客户端访问地址。

```
# 应用名称(必填)
spring.application.name=testSvcName
# 配置北极星服务端地址
```

```
spring.cloud.polaris.address=grpc\://9.134.5.52\:8
```

## 步骤3:添加应用的启动参数

在应用的启动参数中,添加 -javaagent:<java-agent安装目录>/polaris-agent-core-bootstrap.jar ,然后重启应用,即可完成java-agent的注入。

#### () 说明:

如果使用 IDEA 进行本地调试,则需要在 VM Option 下添加 -javaagent 的参数。

# Java Agent 支持无损上下线

最近更新时间: 2024-06-20 16:22:31

## 使用场景

微服务发布过程中,可能会因为服务的变更造成流量的错误或中断。Spring Cloud Tencent 提供了插件来实现无损上下线,其原理为:

- 无损上线:在服务启动后,等待服务完全就绪后再注册到注册中心(无配置健康探测接口时为延迟注册),并对外提供服务。然后再结合K8s生命周期,滚动更新下一个节点。
- 无损下线:在服务停止前,先从注册中心注销,拒绝新的请求,等待旧的请求处理完毕后再下线服务。

## 无损上线

TSE 北极星支持以下 2 种方式支持服务无损上线: 服务就绪/延迟注册 和 服务注册就绪检查 。

## 方案一: 服务就绪/延迟注册

在一些场景下,服务支持延迟加载,启动后异步加载一些资源,例如:服务需要从文件存储 COS 获取数据或者文件,待数据或者文件拉取完成后才能对外提 供服务。如果在应用启动后直接注册服务,会导致服务实际没有就绪而调用失败。因此通过确保服务就绪之后,再进行注册从而对外提供服务,可以确保服务 平滑、无损上线。

TSE 北极星支持以下两种服务就绪/延迟注册场景:

• 场景一: 业务暴露健康检查接口,接口探测成功后,服务注册。

场景二:业务没有暴露健康检查接口,则延迟一段时间服务注册,默认延迟注册时长30s,可通过配置自定义延迟注册的时长。



#### 操作步骤

步骤1: 服务通过 Java Agent 接入北极星。具体步骤参见: Spring Cloud 使用 Java Agent。

步骤2:在上一步接入操作中 步骤3 SpringCloud 应用 yaml 文件中声明 Java Agent 标签 中配置无损上下线相关配置项。具体参数如下:

配置项Key	默认值	是否必填	配置项说明
spring.cloud.polaris.lossless. enabled	false	否	无损上下线开关
spring.cloud.polaris.lossless. health-check-path	无	否	业务程序健康探测接口



spring.cloud.polaris.lossless. delay-register-interval	30000	否	无配置业务程序健康探测接口,延迟注册时间。默认 30000(单位ms )
spring.cloud.polaris.lossless. health-check-interval	5000	否	配置业务程序健康探测接口后,健康探测间隔。默认 5000(单位ms )

配置示例:

apiVersion: apps/v1
kind: Deployment
spec:
template:
metadata:
annotations:
# <b>声明需要往这个</b> POD <b>中注入</b> javaagent
polarismesh.cn/javaagent: "true"
# <b>声明应用的框架类型,对于</b> SpringCloud <b>应用则填写</b> spring-cloud
polarismesh.cn/javaagentFrameworkName: spring-cloud
# <b>声明应用的框架版本,当前支持</b> hoxton, 2023
polarismesh.cn/javaagentFrameworkVersion: hoxton
# <b>声明</b> java-agent <b>包的镜像版本,可用版本:</b> https://github.com/polarismesh/polaris-java-agent/releases
polarismesh.cn/javaagentVersion: 1.7.0-RC3
# 用户自定义的JavaAgent配置,不填写的配置则使用默认配置,格式为JSON。具体配置查看:
https://github.com/polarismesh/polaris-
controller/blob/main/deploy/kubernetes_v1.22/kubernetes/javaagent-configmap.yaml
# 无损上线配置示例:
<pre># polarismesh.cn/javaagentConfig: "{\"spring.cloud.polaris.lossless.enabled\": \"true\",</pre>
\"spring.cloud.polaris.lossless.delay-register-interval\": \"30000\"}"

## 方案二: 服务注册就绪检查

通常 K8s 提供就绪检查机制来对实例在就绪前进行健康检查,它一般会简单地认为端口起来了应用即处于就绪状态。但是实际上,应用端口起来了和服务成 功注册之间存在 gap,这样会造成服务没有成功注册,旧的应用实例就被下线,下一个节点开始部署。最终导致消费端调用异常。 TSE 北极星提供服务注册状态的接口和端口,以配合 K8s 就绪检查,当应用注册完成返回200状态码,帮助 K8s 判定应用已就绪;未完成注册返回500状 态码,帮助 K8s 判定应用未就绪。尤其在实例滚动更新的过程中,等实例就绪后才滚动更新下一个节点。

#### 操作步骤:

步骤1: 服务通过 Java Agent 接入北极星。具体步骤参见 Spring Cloud 使用 Java Agent。

- 步骤2:在上一步接入操作中 步骤3 SpringCloud 应用 yaml 文件中声明 Java Agent 标签 中打开无损上下线的开关。具体参数参见 方案一。 步骤3:在容器服务 TKE 等 K8s 应用部署平台中配置就绪检查,如下图所示。
- 路径: /online。
- 端口: 28080。



<ul> <li>存活检查 检查容器</li> <li>✓ 就绪检查 检查容器</li> </ul>	是否正常,不正常则重启实例 是否就绪,不就绪则停止转发流量到当前实例
检查方法	HTTP请求检查
检查协议	HTTP *
host(j)	默认为 Pod IP, 一般不需要修改
	大多数情况下不需要填 host 字段,请谨慎填写防止探测失败,更多请参考文档 🗹
检查端口	28080 端口范围: 1~66535,支持使用端口名
httpHeader	
请求路径	/online
启动延时(3	清输入启动延移
	<ul> <li>              み張检査 检查容器          </li> <li>             秋绪检查             松查容器         </li> <li>             松童方法         </li> <li>             や宣訪议         </li> <li>             host④         </li> <li>             ればpHeader         </li> <li>             崩水路径         </li> <li>             島动庭时④         </li> </ul>

## 无损下线

应用在滚动发布或者下线过程中,被调方服务实例向注册中心发起反注册,主调方从注册中心更新 IP 的过程中,存在时间 gap,导致依然有可能调用到已经 下线的实例,从而调用失败。

TSE 北极星提供服务无损下线的接口: /offline,结合 K8s 的生命周期,实现服务无损下线。整体流程如下:



## 操作步骤:

步骤1: 服务通过 Java Agent 接入北极星。具体步骤参见 Spring Cloud 使用 Java Agent。

步骤2:在上一步接入操作中 步骤3 SpringCloud 应用 yaml 文件中声明 Java Agent 标签 中打开无损上下线的开关。具体参数参见 方案一。 步骤3:在容器服务 TKE 等 K8s 应用部署平台中配置 preStop 生命周期检查。

preStop 配置检查指令: curl -X PUT http://localhost:28080/offline && sleep 20



# Java Agent 支持配置管理

最近更新时间: 2025-06-11 15:55:02

## 场景说明

该文档指导您通过 Java Agent 接入北极星时,使用配置管理能力的操作步骤。

## 操作步骤

#### () 说明:

Java Agent 的版本要 2.0.1.0及以上。

## 步骤1: 通过 Java Agent 接入北极星

服务通过 Java Agent 接入北极星。具体步骤参见: Spring Cloud 使用 Java Agent。

## 步骤2:配置管理

在上一步接入操作中 步骤3 Spring Cloud 应用 yaml 文件中声明 Java Agent 标签 中配置相关项。具体参数如下:

配置项	配置值	是否必填	配置项说明
spring.config.import	optional:polaris	是	使用北极星的配置中心
spring.cloud.polaris.config.enabled	true	是	开启北极星配置中心功能
spring.cloud.polaris.config.address	例如 grpc://127.0.0.1:8093	是	北极星的配置中心地址
spring.cloud.polaris.config.groups[0].na me	例如 service-provider-2022	是	北极星配置分组名称
spring.cloud.polaris.config.groups[0].files [0]	config/callee.properties	是	指定北极星配置分组文件名称

#### 配置示例:

apiVersion: apps/v1
kind: Deployment
metadata:
labels:
app: service-provider-2022-a
name: service-provider-2022-a
namespace: default
spec:
replicas: 1
selector:
matchLabels:
app: service-provider-2022-a
template:
metadata:
labels:
app: service-provider-2022-a
annotations:
# <b>声明需要往这个</b> POD <b>中注入</b> javaagent
polarismesh.cn/javaagent: "true"
# <b>声明应用的框架类型,对于</b> SpringCloud <b>应用则填写</b> spring-cloud



```
# 用户自定义的JavaAgent配置,不填写的配置则使用默认配置,格式为JSON。
 imagePullPolicy: Always
```

# 使用 Java SDK Polaris – Java 接入

最近更新时间: 2025-04-17 16:00:53

## 操作场景

本文通过一个 demo 进行应用使用 polaris-java 接入微服务引擎托管的 Polaris(北极星)的全流程操作演示,帮助您快速了解如何使用北极星网格。

## 前提条件

- 已创建 Polaris (北极星)实例,请参见 创建 Polaris (实例)。
- 下载 Github 的 demo 源码 到本地并解压。
- 本地编译构建打包机器环境已安装了Java JDK、Maven,并且能够访问 Maven 中央库。
- 根据您自身的业务,已准备好业务部署的资源,虚拟机部署、容器化部署和 TEM 部署选择其中一种方式即可。
  - 虚拟机部署已创建 CVM 虚拟机,请参见 创建 CVM 虚拟机。
  - 容器化部署已创建 TKE 容器集群,请参见 创建 TKE 集群。
  - TEM部署已创建 TEM 环境,请参见 创建 TEM 环境。

## 操作步骤

- 1. 登录 TSF 控制台。
- 2. 在左侧导航栏选择Polaris(北极星),进入北极星实例引擎列表页后单击目标实例进入实例信息页。
- 3. 在引擎管理-实例信息页查看访问地址,polaris-java 应用访问使用 gRPC 端口(8091):

Polaris Server)	方问地址	
客户端接入端口	注册发现	8091 (service-grpc)
		8092 (service-trpc)
	配置管理	8093 (config-grpc)
	监控上报	9091 (pushgateway)
OpenAPI 端口	8090 (http)	
协议兼容端口	7779 (I5)	8761 (eureka) 15010 (xdsv3)
	15020 (xdsv	(2)
内网负载均衡 🕄	私 子阿	网 内 访问控制 操作
	sh-gz gz-	4( 1. 产 删除
	添加内网负载地	均衡
公网负载均衡 🛈	- 开启	

4. 修改 demo 中的注册中心地址。



#### 4.1 在下载到本地的 demo 源码目录 下,分别找到

quickstart-example\quickstart-example-provider\src\main\resources\polaris.yml  $m{n}$ 

quickstart-example\quickstart-example-consumer\src\main\resources\polaris.yml 两个文件。

#### 4.2 添加微服务引擎北极星网格地址到项目配置文件中(以

quickstart-example\quickstart-example-consumer\src\main\resources\polaris.yml 为例)。

global:				
serverConr	ector:			
addresses:				
- 10.0.4.6	5:8091			

#### 5. 打包 demo 源码成 jar 包。

5.1 在 quickstart-example 源码根目录下,打开 cmd 命令,执行 mvn clean package 命令,对项目进行打包编译。

#### 5.2 编译成功后,生成如下表所示的2个 Jar 包。

软件包所在目录	软件包名称	说明
\quickstart-example\quickstart-example-	quickstart-example-provider-\${version}-	服务生产
provider\target	SNAPSHOT.jar	者
\quickstart-example\quickstart-example-	quickstart-example-consumer-\${version}-	服务消费
consumer\target	SNAPSHOT.jar	者

#### 6. 部署 provider 和 consumer 微服务应用,根据您业务实际选择虚拟机部署、容器化部署以及 TEM 部署中的一种部署方式即可。

#### 6.1 虚拟机部署

- 上传 Jar 包至 CVM 实例。
- 执行启动命令进行启动:

nohup java -jar [jar**包名称**] 🌡

#### 6.2 容器化部署

○ 编写 dockerfile 生成镜像,参考:

FF	COM java:8
ADE	/quickstart-example-provider-\${VERSION}.jar /root/app.jar
ENTRYPC	DINT ["java","-jar","/root/app.jar"]

○ 通过 TKE 部署并运行镜像。

## 6.3 TEM 部署



○ 选择 TEM 环境,注意所选择的环境,其依赖的 VPC,必须和上面已经创建的北极星网格实例所依赖的 VPC 一致:

基本信息	资源管理 访问管理 配置管理
基本信息	
环境名称	andr-test-1 💉
ID	env-
所在地域	广州
集群网络	VPC( 【)  子网( 【)
创建时间	2021-12-21 20:08:21
描述	1

○ 在已选择的环境中,新建 TEM 应用,相关参数填写参考:

新建应用		
名称 <b>*</b>	polaris-java-provider	
描述 (选埴)	请输入描述	
开发语言	O JAVA ○ 其他语言	
程序包上传方式	○ 镜像   ○ JAR包      WAR包	
	将为您上传的程序包目动制作镜像,并存储至镜像仓库 著您是第一次使用镜像仓库,请先开通镜像仓库,个人版 🖸 企业版 🖸	
启用组件	<b>勝讯云应用性能观测</b> 应用性能管理平台、基于OpenTracing协议,采用探针采集,主 要支持「基础监控」、「应用级别监控」、以及「链路追踪」等 能力。	
	提交关闭	



○ 部署应用,相关参数填写请参考(端口号映射,consumer 默认端口号为15700, provider 默认端口号为15800):

应用名	polaris-java-provider
发布环境	andr-test-1 / env-
	如无合适环境,可前往创建 🖸
JDK版本	KonaJDK 8 💌
上传程序包	quickstart-example-provider- 重新上传 Demo下载 ▼
版本号	20211222172508 使用时间戳作为版本号
	最长32个字符,支持英文字毋a-z、A-Z、横杠"-"、下划线"_"、点"."
版本描述	请输入版本描述
启动参数	-Xms128m -Xmx512m -XX:MetaspaceSize=128m - XX:MaxMetaspaceSize=512m
▼ 资源配置	
▲ 访问配置	•
访问方式	● 环境内访问 ○ VPC内网访问(四层转发) ○ 公网访问(四层转发) 提供—个可以被环境内其他应用访问的入口,支持 TCP/UDP 协议, 如您需要配置公网的 HTTP/HTTPS 协议转发规则,可能往应用路由页面 Ⅰ 转发规则配置。
端口映射	协议①         容器端口③         应用监听端口④
	TCP • 15800 15800
	添加满口映射
▼ 应用启停管理	在环境启动后和应用结束前设置处理任务,例如:环境初始化、应用优雅退出朝
▼ 配置设置	
▼ 环境变量	
▼ 健康检查	
▼ 持久化存储	为容器提供存储,目前支持器讯云文件存储CFS,还需挂载到容器的指定路径中。使用前,请先前往环境关联CFS存储资源。
▼ 安全组	
▼ 日志配置	
部署	取消

○ 查看访问路径,consumer 应用部署完后,可以在基本信息 > 访问配置中查看访问地址,如需公网访问,可以编辑并更新开启公网访问:

基本信息 日志 监控 实例列表 发布记录	
基本信息	访问配置 编辑并更新
应用名称 polaris-java-consumer	访问方式 <b>公网访问 (四层转发)</b>
应用描述 -	访问地址 公网IP
私有网络 vpc 🖸	转发配置 前往设置 🛛
子网 subnet [2] (可用IP数206个)	
运行/目标实例数 2/2	配置信息
可用区 广州三区	启动参数 -Xms128m -Xmx512m -XX:MetaspaceSize=128m -XX:MaxMetaspaceSize=512m
	环境变量 暫未设置
弹性伸缩	持久化存储 智未设置
规格   1核 2G	日志信息 日志主题:智未设置 采集路径:智未设置
初始化实例数 2个 编辑并更新	配置信息 <b>智未设置</b>
弹性伸缩 未启用 编辑并更新	

7. 确认部署结果。



- 7.1 进入前面提到的微北极星网格实例页面。
- 选择服务管理 > 服务列表, 查看微服务 EchoServerGRPC 的实例数量:
- 若实例数量值不为0,则表示已经成功接入微服务引擎。
- 若实例数量为0,或者找不到 EchoServerGRPC 服务名,则表示微服务应用接入微服务引擎失败。



curl -L -X GET 'http://\${add.address}:\${app.port}/echo?value=hello\_world' 预期返回值: echo: hello\_world



## 注册发现

最近更新时间: 2025-04-17 16:00:53

## 操作场景

本文介绍在本地开发 Java 应用,通过 polaris-sdk 的方式接入 Polaris(北极星),并实现服务注册与发现。

## 前提条件

在开发前,请确保您已经下载并安装了 Java 和 Maven。

## 操作步骤

为了方便您快速接入,我们为您准备了 Demo 应用, 点击下载 。

## 步骤1: 引入北极星依赖

## 1. 引入 polaris sdk 依赖

修改应用根目录下的 pom.xml ,为 polaris-java 添加 dependencyManagement :

#### 🕛 说明:

polaris-sdk 版本信息请参见:版本信息。

## 2. 引入 polaris starter

## 步骤2:添加北极星配置文件 polaris.yml

- 1. 在项目的 main/resources 目录下创建 polaris.yml 文件用于初始化 polaris-java SDK。
- 2. 在 polaris.yml 文件中配置应用名、polaris(北极星)服务端地址等信息。服务端地址查看详见: 引擎管理 > 客户端访问地址 。





#描述:监控及日志数据上报相关配置
statReporter:
#描述:是否启用上报
enable: true
plugin:
prometheus:
type: push
# <b>描述: 设置</b>
# 地址需要替换成您创建的北极星引擎的客户端访问地址。
address: 127.0.0.1:9091
# <b>描述:设置</b> metric <b>数据推送到</b> pushgateway <b>的执行周期</b>
# <b>范围:</b> [1s:] <b>,默认值:</b> 10s
pushInterval: 10s

更多 polaris.yml 配置信息,请 default-config.yml。

## 步骤3:应用开发

1. 服务注册(服务生产者)

#### 1.1 初始化 DiscoveryAPI 实例

import com.tencent.polaris.factory.api.DiscoveryAPIFactory;
public static void main(String[] args) throws Exception {
 ProviderAPI providerAPI = DiscoveryAPIFactory.createProviderAPI();
}

#### 1.2 注册请求体,设置健康检查、metadata 等信息。

```
// InstanceRegisterRequest 注册服务请求
type InstanceRegisterRequest struct {
    // 必选, 服务名
    Service string
    // 必选, 命名空间
    Namespace string
    // 必选, 服务实例监听port
    Port int
    // 少选, 服务实例监听port
    Port int
    // 可选, 资源访问Token, 即用户/用户组访问凭据, 仅当服务端开启客户端鉴权时才需配置
    ServiceToken string
    // U下字段可选, 默认ALL表示客户端不配置, 使用服务端配置
    // 服务协议
    Protocol *string
    // 服务协议
    Protocol *string
    // 服务协议
    Protocol *string
    // 或选, 范围0~10000
    Weight *int
    // 实例据供服务版本号
    Version *string
    // 或能务实例呈示性重要。
    Kudeata map[string]string
    // 该服务实例是否键重, 默认健康
    Healthy *bool
    // 该服务实例是否确重, 默认不隔离
    Isolate *bool
    // 设置の挑键廉检查, 客户端必须以TTI间隔上报心跳
    // 设置心跳链廉检查, 客户端必须以TTI间隔上报心跳
    // 正下。
    // Tu *int
    // 算检查服务器:%TTI未要到心跳明将实例置为不健康
    // Tu *int
    // Sundata map(string)
    // 按正常端的公式TTI和表示。
    // 如告。
    // 如告。你们有一个问题上很心能
    // 如告。
    // 如告。你们有一个问题。
    // 如告。你们有一个问题是我们有一个问题。
    // 如告。你们有一个问题。
    // 如告。
    // 如告。
    // 如告。
    // 如告
    //
```




#### 1.3 发起注册请求

在初始化完 InstanceRegisterRequest 结构体后,只需要调用 ProviderAPI.RegisterInstance 方法即可完成实例注册,并且 RegisterInstance 方法内部会自动维护实例的心跳上报。

InstanceRegisterResponse registerResp = providerAPI.registerInstance(registerRequest

#### 2. 服务发现(服务消费者)

## 2.1 初始化 polaris sdk 实例



#### 2.2 获取服务提供方的实例信息

北极星提供三种获取目标服务实例的方法:GetAllInstances、GetHealthyInstances、GetOneInstances,分别获取全部实例、健康实例和一个实例。详情如下:

#### (1) GetAllInstances

直接返回目标服务下的所有实例,包括不健康、隔离、权重为0、被熔断的实例,都会在返回的实例列表中。



#### (2) GetHealthyInstances

每次获取一批可用服务提供者实例。该方法默认会过滤掉不健康、隔离、权重为0、被熔断的实例。





// 可选,设置主调服务信息,只用于路由规则匹配
SourceService serviceInfo = new SourceService();
// 设置主调服务命名空间
serviceInfo.setNamespace(String namespace); // 设置主调服务名称
<pre>serviceInfo.setService(String service);</pre>
// 设置主调方的请求标签信息
<pre>serviceInfo.setArguments(Set<routeargument> arguments);</routeargument></pre>
<pre>request.setServiceInfo(serviceInfo);</pre>
// 设置超时时间
<pre>request.setTimeoutMs(long timeoutMs);</pre>
// <b>调用</b> ConsumerAPI 执行该请求

#### (3) GetOneInstances

每次仅获取一个可用服务提供者实例,该方法会依次执行路由、负载均衡流程。该方法默认会过滤掉不健康、隔离、权重为0、被熔断的实例。

#### ▲ 注意:

#### 执行路由流程的条件

- 配置了 GetOneInstanceRequest.ServiceInfo.Metadata 属性,会触发自定义路由流程。
- 设置了 GetOneInstanceRequest.Metadata 属性,会触发元数据路由流程。



// 设置主调服务命名空间
<pre>serviceInfo.setNamespace(String namespace);</pre>
// 设置主调服务名称
<pre>serviceInfo.setService(String service);</pre>
// 设置主调方的请求标签信息
<pre>serviceInfo.setArguments(Set<routeargument> arguments);</routeargument></pre>
request.setServiceInfo(serviceInfo);
// 设置超时时间
<pre>request.setTimeoutMs(long timeoutMs);</pre>
// <b>调用</b> ConsumerAPI <b>执行该请求</b>
<pre>consumerAPI.getOneInstance(request);</pre>

# 步骤4: 应用部署

Demo 应用部署可参见: Demo 应用部署。



# 负载均衡

最近更新时间: 2025-04-17 16:00:53

# 操作场景

本文介绍在本地开发 Java 应用,通过 polaris-sdk 的方式接入 Polaris(北极星),并实现负载均衡功能。

# 前提条件

在开发前,请确保您已经下载并安装了 Java 和 Maven。

## 操作步骤

## 步骤1: 引入北极星依赖

## 1. 引入 polaris sdk 依赖

修改应用根目录下的 pom.xml ,为 polaris-java 添加 dependencyManagement :



#### () 说明:

polaris-sdk 版本信息请参见:版本信息。

## 2. 引入 polaris starter

```
<dependencies>

<dependency>

<groupId>com.tencent.polaris</groupId>

<artifactId>polaris-all</artifactId>

</dependency>

</dependencies>
```

# 步骤2:添加北极星配置文件 polaris.yaml

- 1. 在项目的 main/resources 目录下创建 polaris.yml 文件用于初始化 polaris-java SDK。
- 2. 在 polaris.yml 文件中配置应用名、polaris(北极星)服务端地址等信息。服务端地址详细参见: 引擎管理 > 客户端访问地址。





#捆还: 走省后用上版
enable: true
plugin:
prometheus:
type: push # 描述: 设置 pushgateway 的地址, 仅 type == push 时生效 # 地址需要替换成您创建的北极星引擎的客户端访问地址。 address: 127.0.0.1:9091 #描述:设置metric数据推送到pushgateway的执行周期 #范围:[1s:],默认值:10s pushInterval: 10s

更多 polaris.yml 配置信息,请参见 default-config.yml。

## 步骤3:应用开发

#### 1. 初始化 RouterAPI 实例

```
import com.tencent.polaris.factory.api.RouterAPIFactory;
public static void main(String[] args) throws Exception {
    RouterAPI routerAPI = RouterAPIFactory.createRouterAPI();
}
```

2. 注册请求体,设置执行负载均衡的实例列表、负载均衡策略等信息。



#### 3. 负载均衡

您在使用 ConsumerAPI.getAllInstances 或者 ConsumerAPI.getInstances 获取到服务实例列表后,完成 ProcessLoadBalanceRequest 初始化,只需要调用 RouterAPI.processLoadBalance 方法即可完成负载均衡。

ProcessLoadBalanceResponse resp = routerAPI.processLoadBalance(request)



# 动态路由

最近更新时间: 2025-04-17 16:00:53

# 操作场景

本文介绍在本地开发 Java 应用,通过 polaris-sdk 的方式接入 Polaris(北极星),并实现服务路由功能。

# 前提条件

在开发前,请确保您已经下载并安装了 Java 和 Maven。

## 操作步骤

## 步骤1: 引入北极星依赖

## 1. 引入 polaris sdk 依赖

修改应用根目录下的 pom.xml ,为 polaris-java 添加 dependencyManagement :



#### () 说明:

polaris-sdk 版本信息请参见:版本信息。

## 2. 引入 polaris starter

```
<dependencies>

<dependency>

<groupId>com.tencent.polaris</groupId>

<artifactId>polaris-all</artifactId>

</dependency>

</dependencies>
```

# 步骤2:添加北极星配置文件 polaris.yaml

- 1. 在项目的 main/resources 目录下创建 polaris.yml 文件用于初始化 polaris-java SDK。
- 2. 在 polaris.yml 文件中配置应用名、polaris (北极星)服务端地址等信息。服务端地址查看详见: 引擎管理 > 客户端访问地址。





#捆还: 走台后用上报
enable: true
plugin:
prometheus:
type: push # 描述: 设置 pushgateway 的地址,仅 type == push 时生效 # 地址需要替换成您创建的北极星引擎的客户端访问地址。 address: 127.0.0.1:9091 #描述:设置metric数据推送到pushgateway的执行周期 #范围:[1s:],默认值:10s pushInterval: 10s

更多 polaris.yml 配置信息,请参见 default-config.yml。

## 步骤3:应用开发

### 1. 初始化 routerAPI 实例

```
import com.tencent.polaris.factory.api.RouterAPIFactory;
public static void main(String[] args) throws Exception {
    RouterAPI routerAPI = RouterAPIFactory.createRouterAPI();
}
```

2. 注册请求体:设置路由策略、实例列表、metedata等信息。

```
// ProcessRoutersRequest 执行路由请求结构体
type ProcessRoutersRequest struct {
    // 可选参数, 设置本次路由请求期望执行的路由插件
    // 当前支持的路由插件如下
    // - 自定义路由: ruleBasedRouter
    // - 自定义路由: ruleBasedRouter
    // - 前太近路由: nearbyBasedRouter
    // - 元数据路由: dstMetaRouter
    // - 可选参数, 主调服务信息, 你可以通过 ServiceInfo.Metadata 设置本次请求的流量标签信息
    SourceService ServiceInfo
    // 少选参数, 侍执行服务路由的实例列表
    // 1. InstancesResponse, returned from ConsumerAPI.GetAllInstances.
    // 2. DefaultServiceInstances, for user to construct manually.
    DstInstances ServiceInstances
    // 可选参数, 对应路由规则中的方法(Smethod)标签
    Method string
    // 可选, 单次查询超时间为(1+RetryCount) * Timeout
    Timeout *time.Duration
    // 可选, 重试次数, 默认直接获取全局的超时配置
    RetryCount *int
}
①    Unit:
        Duration
        // 可述, 重试次数, 默认直接获取全局的超时配置
        RetryCount *int
}
```

如果当前 ProcessRoutersRequest 还不支持 AddArgument 方法,同时服务端版本 >= 1.12.0, SourceService.Metadata 对应的 key 名称如下:

- 路径: \$path
- 方法: \$method
- 请求头: \$header.{标签键}
- 请求参数: \$query.{标签键}



- 请求COOKIE: \$cookie.{标签键}
- 主调IP: \$caller\_ip
- 自定义: {标签键}

## 3. 执行服务路由

您在初始化完 ProcessRoutersRequest 结构体后,只需要调用 RouterAPI.processRouters 方法即可完成服务路由。

ProcessRoutersResponse resp = routerAPI.processRouters(registerRequest)



# 就近访问

最近更新时间: 2024-01-22 16:19:31

# 操作场景

本文通过一个 demo 进行 Java 应用接入微服务引擎托管的 PolarisMesh 治理中心的全流程操作演示,帮助您快速了解如何使用北极星网格的就近路由能 力。

## 前提条件

- 已创建 PolarisMesh 北极星网格,请参见 创建 PolarisMesh 治理中心。
- 下载 Github 的 demo 源码 到本地并解压。
- 本地编译构建打包机器环境已安装了 Java 环境,并且能够访问 Github。
- 根据您自身的业务,已准备好业务部署的资源,虚拟机部署和容器化部署选择其中一种方式即可。
  - 虚拟机部署已创建 CVM 虚拟机,请参见 创建 CVM 虚拟机。
  - 容器化部署已创建 TKE 容器集群,请参见 创建 TKE 集群。

# 操作步骤

- 1. 登录 TSE 控制台。
- 2. 在北极星网格下的 polarismesh 页面,单击页面左上方下拉列表,选择目标地域。
- 3. 单击目标引擎的"ID",进入基本信息页面。
- 4. 查看访问地址,Java 应用访问使用 gRPC 端口(8091):

访问地址										
IIII 0         8090/http、8091/grpc、8092/trpc、7779/f5、15020/xds(v2)、8761/eureka										
内网地址	私有网络	子网	内网地址							
	pk-gz-vpc(vpc- ) 🗈	polaris-dev(subnet-	1.6							

#### 5. 修改 demo 中的注册中心地址:

- 在下载到本地的 demo 源码 目录下,分别找到 quickstart-example-provider/src/main/resources/polaris.yml 以及 quickstart-example-consumer/src/main/resources/polaris.yml 文件。
- 添加微服务引擎北极星网格地址以及就近路由相关配置到项目配置文件中(这里以 quickstart-example-provider/src/main/resources/polaris.yml 为例)。

#### 通过环境变量获取 SDK 所在地域信息

• 通过腾讯云 API 获取当前 CVM /容器 POD 的地域信息

```
export ZONE=$(curl http://metadata.tencentyun.com/latest/meta-data/placement/region)
export CAMPUS=$(curl http://metadata.tencentyun.com/latest/meta-data/placement/zone)
```

• 修改 polaris.yaml 文件

jlobal:
serverConnector:
addresses:
# 设置北极星服务:

版权所有:腾讯云计算(北京)有限责任公司



#### # 设置服务调用相关参数

onsumer:

serviceRouter

- plugin
  - # 设置就近路由插件配置
  - nearbyBasedRouter
    - # 描述:就近路由的最小匹配级别,需要显示设置
    - # 范围: zone(腾讯云地域信息,eg: ap-guangzhou)、campus(腾讯云可用区, eg: ap-guangzhou-3)
  - matchLevel: campus

#### 通过腾讯云 API 获取 SDK 所在地域信息



- 6. 部署 provider 和 consumer 微服务应用。详情请参见 Polaris-Java Demo 部署。
- 7. 确认部署结果。
- 进入前面提到的微服务引擎北极星网格实例页面。
- •选择注册中心 > 服务列表,查看微服务 EchoServerJava 的实例数量。
- 若实例数量值不为0,则表示已经成功接入微服务引擎。同时确认实例所处的腾讯云地址位置信息。
- 若实例数量为0,或者找不到 EchoServerJava 服务名,则表示微服务应用接入微服务引擎失败。

EchoServerJava(default)												
服务实例 路由规则	熔断规则	限流规则	服务信息									
3512 BIR	其他操作								请选择条件进行	立道	Q Ø	
实例IP	端口	协议	版本	权重	健康状态 ▼	隔离状态 ▼	地域/可用区	创建时间	修改时间	操作		
▶ 9.135.224.139	15901	http	1.0.0	100	98.98	不隔离	ap-guangzhou/ap- guangzhou-3	2023-01-12 18:14:	2023-01-12 18:14:	/ Ū		
共 1 条								10 + #	/页 14 4 1	/1页	F F	



• 开启 EchoServerJava 的就近路由。

服务列表									
8518									
□ 服务名									
EchoServerJava	61910 #							2023-01-12 17:46:54	
EchoSarvarGRPC	猫和新放劳						×	2023-01-12 16:58:48	
▶ polaris.checker	命名空间。	default			*			2021-09-06 15:55:09	
共名条	服务名•	EchoServerJava							
	8873								
	业务								
	开启就近访问	● 开启	目标服务就近	路由					
	服务标签③	根签名	标篮值			删除			
		internal-auto-cre	ate true			×			
		851 <b>10</b>							
	描述	长度不超过1024个	字符,标签数量不能超过64						
	▼ 高级设置								
				提交 关	闭				

• 调用 consumer 的 HTTP 接口,执行 http 调用,其中 \${app.port} 替换为 consumer 的监听端口(默认为16011), \${add.address} 则替换 为 consumer 暴露的地址。





# 服务限流

最近更新时间: 2025-04-17 16:00:53

# 操作场景

本文介绍在本地开发 Java 应用,通过 polaris-sdk 的方式接入 Polaris(北极星),并实现服务的服务限流功能。

# 前提条件

在开发前,请确保您已经下载并安装了 Java 和 Maven。

# 操作步骤

## 步骤1: 引入北极星依赖

## 1. 引入 polaris sdk 依赖

修改应用根目录下的 pom.xml ,为 polaris-java 添加 dependencyManagement :

#### () 说明:

polaris-sdk 版本信息请详细参见:版本信息。

## 2. 引入 polaris starter

```
<dependencies>

<dependency>

<groupId>com.tencent.polaris</groupId>

<artifactId>polaris-all</artifactId>

</dependency>

</dependencies>
```

## 步骤2: 添加北极星配置文件 polaris.yml

- 1. 在项目的 main/resources 目录下创建 polaris.yml 文件用于初始化 polaris-java SDK。
- 2. 在 polaris.yml 文件中配置应用名、Polaris(北极星)服务端地址等信息。服务端地址详细参见: 引擎管理 > 客户端访问地址 。





	神田还: 定省后用上报
	enable: true
F	olugin:
	prometheus:
	type: push # 描述: 设置 pushgateway 的地址, 仅 type == push 时生效 # 地址需要替换成您创建的北极星引擎的客户端访问地址。 address: 127.0.0.1:9091 #描述:设置metric数据推送到pushgateway的执行周期 #范围:[1s:],默认值:10s pushInterval: 10s

更多 polaris.yml 配置信息,请参见 default-config.yml。

#### 步骤3:应用开发

#### 1.初始化 LimitAPI 实例:

```
import com.tencent.polaris.ratelimit.factory.LimitAPIFactory;
public static void main(String[] args) throws Exception {
   LimitAPI limitAPI = LimitAPIFactory.createLimitAPI();
}
```

#### 2.请求配额

```
QuotaRequest quotaRequest = new QuotaRequest();

// 设置需要进行限流的服务信息: 设置命名空间信息

quotaRequest.setNamespace(String namespace);

// 设置需要进行限流的服务信息: 设置服务名称信息

quotaRequest.setService(String service);

// 设置本次被调用的方法信息

quotaRequest.setMethod(String method);

// 设置本次的请求标签

quotaRequest.setArguments(Set<Argument> arguments)

// 设置需要申请的请求配额数量

quotaRequest.setCount(1);
```

如果当前 QuotaRequest 还不支持 AddArgument 方法,同时服务端版本 >= 1.11.0, SetLabels 对应的 key 名称如下:

- 路径: \$path
- 方法: \$method
- 请求头: \$header.{标签键}
- 请求参数: \$query.{标签键}
- 主调服务: \$caller\_service
- 主调IP: \$caller ip
- 自定义: {标签键}

## 3.发起配额申请请求

在接收到请求之后对 QuotaRequest 结构体完成初始化后,只需要调用 LimitAPI.GetQuota 方法即可完成本次请求配额的申请。

QuotaResponse resp = limitAPI.getQuota(registerRequest);





最近更新时间: 2023-08-23 16:21:53



# 操作场景

本文针对用户接入微服务引擎托管的 PolarisMesh 治理中心希望使用北极星分布式限流能力。

#### 为何需要用户手动创建

用户接入方式存在多 VPC、公网接入两类场景。不同的接入侧用户需要看到不同的服务端 IP,无法自动创建相关北极星系统服务。

## 前提条件

已创建 PolarisMesh 服务治理中心,请参见 创建 PolarisMesh 治理中心。

## 操作步骤

- 1. 登录 TSE 控制台。
- 2. 在治理中心下的 polarismesh 页面,单击页面左上方下拉列表,选择目标地域。
- 3. 单击目标引擎的"ID",进入基本信息页面。
- 4. 根据实例信息的控制台。

# 引擎实例未开启客户端公网访问

#### 创建 polaris.limiter

- 1. 在实例的基本信息页面中,记住访问地址中的每个 VPC 内网 IP 地址以及访问端口。
- 2. 针对每一个内网 VPC 的 IP 地址,在命名空间 Polaris 下创建服务 polaris.limiter-{vpc 名称}。

新建服务					×
命名空间 *	Polaris		▼		
服务名 *	polaris.limiter-vpc-xxx				
部门					
业务					
服务标签①	标签名    标	签值		删除	
	新增				
描述	长度不超过1024个字符,标签数	全不能超过64个			
▼ 高级设置					
		摄	交关闭		

3. 单击服务 polaris.limiter-{vpc 名称},进入到服务的实例页面,根据对应上述第一步所记录的 VPC 内网 IP 地址以及端口协议 grpc 创建实例。



新建服务实例		×
实例IP ★	10.0.0.1 对应	VPC 的内网接入 IP
端□•	8101	
权重►	- 100 +	定 <del>为 8101</del> ,
协议	arac	
版本		
实例标签 🛈	标签名 标签值	删除
	新增	
地域信息	请输入 Region 请输入 Zone	请输入 Campus
健康状态		
开启健康检查①		
是否隔离①		

4. 按照上述步骤的操作,最终服务 polaris.limiter-{vpc 名称} 的实例列表如下:

FR BB Read									请选择条件进行过滤	Qφ	
英朝IP	第〇	协议	版本	权重	健康状态 ▼	隔高状态 ▼	地区/地域/可用区	\$118891/cl	修改时间	操作	
▶ 10.0.0.1	8101	grpc		100	SE IR	不隔离	//	2023-06-21 14:40:45	2023-06-21 14:40:45	/ 亩	
共1条									10 * 条/页	× < 1 /1	1页 > ×

```
△ 注意:
```

如果只是单 VPC 接入的场景,则无需参考 polaris.limiter-{vpc 名称} 这种命名规则,直接使用 polaris.limiter 即可。

## polaris-java 接入相关配置文件说明

• polaris.yaml 中按照下面示例进行配置二次寻址:

global:	
serverConnector:	
addresses:	
- '{ <b>对应</b> vpc 的内网接入地址}:8091'	
provider:	
rateLimit:	
enable: true	
limiterNamespace: Polaris	
<b>limiterService:</b> 'polaris.limiter-{vpc <b>名称</b>	

## spring-cloud-tencent 接入相关配置文件说明

• 修改 bootstrap.yaml 或者 bootstrap.properties 文件:

spring:			
cloud:			
polaris:			



# address: "{**对应** vpc 的内网接入地址}:8091"

- 在 resources 下创建 polaris.yml 文件。
- 按照下面示例进行配置二次寻址:

```
provider:
rateLimit:
enable: true
limiterNamespace: Polaris
limiterService: 'polaris.limiter-{vpc 名称}'
```

# 引擎实例开启客户端公网访问

#### 创建 polaris.limiter 服务

- 1. 在实例的基本信息页面中,记住访问地址中的公网 IP 地址以及访问端口。
- 2. 在命名空间 Polaris下创建服务 polaris.limiter。
- 3. 单击服务 polaris.limiter,进入到服务的实例页面,根据客户端公网访问 IP 地址以及端口协议 grpc 分别创建实例。

新建服务实例		×
实例IP ◆	多个IP以英文逗号、英文分号、空格或换行分隔,每次最多添加100个IP 填写公网 IP	
	请填写IP	
端口 *	8101	
	端口固定为 8101, 协议为 grpc	
权重 •	- 100 +	
协议	grpc	
版本		
实例标签 🕄	标签名 标签值 删除	
	新墙	
地域信息	请输入 Region 请输入 Zone 请输入 Campus	
健康状态		
开启健康检查 🛈		
是否隔离()		
	提交关闭	

4. 按照上述步骤的操作,最终服务 polaris.limiter 的实例列表如下:

<b>3512</b> EK 12 22 12	操作									诸选择条件进行过渡	Q Ø
实例P	開口	协议	版本	权重	健康状态 平	國高状态 平	地区/地域/可用区	包括肥甲的间	修改时间	操作	
▶ 183.0.0.1	8101	grpc	-	100	SE IR	不隔离	//	2023-06-21 14:46:03	2023-06-21 14:46:03	/ 亩	
共 1 条									10 -  条/页	K < 1 /	1页 → ×

#### polaris-java 接入相关配置文件说明

• polaris.yaml 中按照下面示例进行配置二次寻址:

global:





#### spring-cloud-tencent 接入相关配置文件说明

• 修改 bootstrap.yaml 或者 bootstrap.properties 文件。

```
spring:
cloud:
polaris:
address: '{对应公网接入地址}:8091'
```

- 在 resources 下创建 polaris.yml 文件。
- 按照下面示例进行配置二次寻址:

```
provider:
rateLimit:
enable: true
limiterNamespace: Polaris
limiterService: polaris.limite
```



# 熔断降级

最近更新时间: 2025-04-17 16:00:53

# 操作场景

本文介绍在本地开发 Java 应用,通过 polaris-sdk 的方式接入 Polaris(北极星),并实现熔断降级功能。

# 前提条件

在开发前,请确保您已经下载并安装了 Java 和 Maven。

## 操作步骤

## 步骤1: 引入北极星依赖

## 1. 引入 polaris sdk 依赖

修改应用根目录下的 pom.xml ,为 polaris-java 添加 dependencyManagement :



#### () 说明:

polaris-sdk版本信息请参见:版本信息。

## 2. 引入 polaris starter

# 步骤2:添加北极星配置文件 polaris.yaml

- 1. 在项目的 main/resources 目录下创建 polaris.yml 文件用于初始化 polaris-java SDK。
- 2. 在 polaris.yml 文件中配置应用名、polaris (北极星)服务端地址等信息。服务端地址查看详细参见: 引擎管理 > 客户端访问地址。





enable: true plugin: prometheus: type: push # 描述: 设置 pushgateway 的地址,仅 type == push 时生效 # 地址需要替换成您创建的北极星引擎的客户端访问地址。 address: 127.0.0.1:9091 #描述:设置metric数据推送到pushgateway的执行周期 #范围:[1s:],默认值:10s pushInterval: 10s	# <b>f</b>	<b>笛还: 是</b> 省后用上报
<pre>plugin:     prometheus:     type: push     # 描述: 设置 pushgateway 的地址, 仅 type == push 时生效     # 地址需要替换成您创建的北极星引擎的客户端访问地址。     address: 127.0.0.1:9091     #描述:设置metric数据推送到pushgateway的执行周期     #范围:[1s:],默认值:10s     pushInterval: 10s</pre>		nable: true
prometheus: type: push # 描述: 设置 pushgateway 的地址,仅 type == push 时生效 # 地址需要替换成您创建的北极星引擎的客户端访问地址。 address: 127.0.0.1:9091 #描述:设置metric数据推送到pushgateway的执行周期 #范围:[1s:],默认值:10s pushInterval: 10s	p]	lugin:
type: push # 描述: 设置 pushgateway 的地址, 仅 type == push 时生效 # 地址需要替换成您创建的北极星引擎的客户端访问地址。 address: 127.0.0.1:9091 #描述:设置metric数据推送到pushgateway的执行周期 #范围:[1s:],默认值:10s pushInterval: 10s		prometheus:
		type: push # 描述: 设置 pushgateway 的地址,仅 type == push 时生效 # 地址需要替换成您创建的北极星引擎的客户端访问地址。 address: 127.0.0.1:9091 #描述:设置metric数据推送到pushgateway的执行周期 #范围:[1s:],默认值:10s pushInterval: 10s

更多 polaris.yml 配置信息,请参见 default-config.yml。

#### 步骤3:使用说明

北极星支持三种维度的熔断降级:服务级、接口级、实例级。

#### 1. 控制台配置熔断规则

请参见服务治理指南 > 服务治理 > 熔断降级相关文档,相关界面如下:

服务级熔断	节点级熔断 主动探测							
新建熔断规则	9						请选择条件进行过滤	Q
ID/规则名		状态 ▼	主调服务	被调服务	创建时间/修改时间	启用时间	操作	
■ f6a08b93c ■ test	54c84570bc2055f83fe2a2e2	已启用	命名空间: * 服务: *	命名空间: default 服务: * 接口:	2023-01-26 17:58:43 2023-01-26 17:58:43	2023-01-26 17:58:43	3 禁用 编辑 删除	
描述	-							
错误判断条件	返回码全匹配500 时延全匹配500							
描述	错误率>=30% (统计周期:60,最小 连续错误数>=5个	请求数: 5)						
熔断粒度	服务							
熔断时长	60€)							
恢复策略	当满足3个连续成功请求后恢复							
主动探测	开启							
自定义响应	未开启							
共 1 条						10 ▼ 条/页 🛛 🕅	≪ 1 /1页 ▶	H

熔断规则配置示例:针对 default 命名空间下所有的服务,对于时延大于500毫秒,或者返回码为500的请求,标识为错误请求,一旦一分钟内错误率30% 及以上或连续错误数在5个以上,则对服务进行熔断。

#### 2. 客户端使用 SDK 进行熔断判断

#### 方法说明:

北极星 Java SDK 提供以下熔断相关的方法,所有的方法都在 com.tencent.polaris.circuitbreak.api.CircuitBreakAPI 接口中提供。

- check:检查资源是否可被调用,并对资源获取调用申请。对于半开的资源,如果半开的调用配额申请成功,返回 true,否则返回 false。
- report: 该方法供用户在资源调用完成后,上报调用的结果,包括返回码、时延等信息,供熔断逻辑判断。
- makeFunctionalDecorator:创建一个函数调用装饰器 FunctionalDecorator,装饰器可以对 Java 的函数接口进行装饰。装饰后的逻辑,会在函数逻辑调用前,先通过 check 方法检查资源是否可被调用,如果不能被调用,会抛出资源熔断异常(CallAbortedException)。调用完成后,会通过 report 接口上报本次调用结果。

FunctionalDecorator 包含以下方法:

- decorateSupplier: 对函数接口 Supplier 进行封装。
- decorateConsumer: 对函数接口 Consumer 进行封装。
- decorateFunction: 对函数 Function 进行封装。
- decoratePredicate: 对函数接口 Predicate 进行封装。

#### 2.2 服务级熔断



```
// 创建CircuitBreakAPI gdl
CircuitBreakAPI circuitBreakAPI = CircuitBreakAPIFactory.oreateCircuitBreakAPI();
// 通过传入服务名(testService))和命名空间(default); 创建FunctionalDecorator
FunctionalDecoratorRequest makeDecoratorRequest = new FunctionalDecoratorRequest();
makeDecoratorRequest.setService(new ServiceKey("default", "testService1"));
FunctionalDecorator decorator = circuitBreakAPI.makePunctionalDecorator(makeDecoratorRequest);
// 封装函数接口
Consumer<Integer> integerConsumer = decorator.decorateConsumer(new Consumer<Integer>() {
    @Override
    public void accept(Integer object) {
        // 执行服务调用...
        }
    });
// 通过执行函数接口,进行服务调用
// 在调用过程中,如果出现熔断,会抛出CallAbortedException异常
for (int i = 0; i < 500; i++) {
    try {
        integerConsumer.accept(i);
        } catch(CallAbortedException e) {
            e.printStackTrace();
        }
    }
```

#### 2.3 接口级熔断

```
// 创建CircuitBreakAPI实例
CircuitBreakAPI circuitBreakAPI = CircuitBreakAPIFactory.createCircuitBreakAPI();
// 通过传入服务名(testService1)、命名空间(default)和方法名(foo), 创建FunctionalDecorator
FunctionalDecoratorRequest makeDecoratorRequest = new FunctionalDecoratorRequest();
makeDecoratorRequest.setService(new ServiceKey("default", "testService1"));
makeDecoratorRequest.setMethod("foo");
FunctionalDecorator decorator = circuitBreakAPI.makeFunctionalDecorator(makeDecoratorRequest);
// 封装函数接口
Consumer<Integer> integerConsumer = decorator.decorateConsumer(new Consumer<Integer>() {
    @Override
    public void accept(Integer object) {
        // 执行服务接口调用...
        }
    });
// 通过执行函数接口,进行服务调用
// 在调用过程中,如果出现缩断,会抛出CallAbortedException异常
for (int i = 0; i < 500; i++) {
    try {
        integerConsumer.accept(i);
        } catch(CallAbortedException e) {
            e.printStackTrace();
        }
    }
}
```

#### 2.4 实例级熔断



当实例被熔断时,该实例会暂时不接收请求,原本路由到该实例的请求会路由到其他实例。这个过程在服务路由过程中自动完成,用户无需进行额外的熔断状 态判断等操作。 执行服务路由:



# 配置管理

最近更新时间: 2025-04-17 16:00:53

# 操作场景

本文介绍在本地开发 Java 应用,通过 polaris-sdk 的方式接入 Polaris(北极星),并实现配置分组管理功能。

# 前提条件

在开发前,请确保您已经下载并安装了 Java 和 Maven。

## 操作步骤

## 步骤1: 引入北极星依赖

## 1. 引入 polaris sdk 依赖

修改应用根目录下的 pom.xml ,为 polaris-java 添加 dependencyManagement :



#### () 说明:

polaris-sdk 版本信息请参见:版本信息。

#### 2. 引入 polaris starter

```
<dependencies>

<dependency>

<groupId>com.tencent.polaris</groupId>

<artifactId>polaris-all</artifactId>

</dependency>

</dependencies>
```

# 步骤2:添加北极星配置文件 polaris.yml

- 1. 在项目的 main/resources 目录下创建 polaris.yml 文件用于初始化 polaris-java SDK。
- 2. 在 polaris.yml 文件中配置应用名、polaris (北极星)服务端地址等信息。服务端地址详细参见: 引擎管理 > 客户端访问地址。





#t#	<b>时处: 定省后用上</b> 扳
	able: true
pl	ugin:
	prometheus:
	type: push # 描述: 设置 pushgateway 的地址, 仅 type == push 时生效 # 地址需要替换成您创建的北极星引擎的客户端访问地址。 address: 127.0.0.1:9091 #描述:设置metric数据推送到pushgateway的执行周期 #范围:[1s:],默认值:10s pushInterval: 10s

更多 polaris.yml 配置信息,请参见 default-config.yml。

#### 步骤3: 应用开发

#### 1.获取配置

(1) 初始化 ConfigFileService SDK 实例

```
import com.tencent.polaris.configuration.factory.ConfigFileServiceFactory;
public static void main(String[] args) throws Exception {
    ConfigFileService configFileService = ConfigFileServiceFactory.createConfigFileService();
}
```

#### (2)获取配置文件

```
// 获取特定远程的配置文件
ConfigFile getConfigFile(String namespace, String fileGroup, String fileName);
// 获取特定远程的配置文件
ConfigFile getConfigFile(ConfigFileMetadata configFileMetadata);
```

#### (3) 监听配置文件

```
//获取配置文件
ConfigFile configFile = configFileService.getConfigFile(namespace, fileGroup, fileName);
//添加变更监听器
configFile.addChangeListener(new ConfigFileChangeListener() {
    @Override
    public void onChange(ConfigFileChangeEvent event) {
    }
});
```

#### (4)监听配置分组下的已发布文件列表变化

获取到目标配置分组后,调用配置分组的 addChangeListener 方法监听改配置分组下已发布配置文件列表的变化。





# });

#### ConfigFileGroupChangedEvent 数据结构

#### 2.操作配置

(1) 初始化 ConfigFilePublishService SDK 实例。

```
import com.tencent.polaris.configuration.factory.ConfigFileServicePublishFactory;
public static void main(String[] args) throws Exception {
    ConfigFilePublishService configFilePublishService =
    ConfigFileServicePublishFactory.createConfigFilePublishService();
}
```







void updateConfigFile(ConfigFileMetadata configFileMetadata, String content);

// 发布配置文件

void releaseConfigFile(String namespace, String fileGroup, String fileName);

// 发布配置文件

void releaseConfigFile(ConfigFileMetadata configFileMetadata);



# 可观测性

最近更新时间: 2025-04-17 16:00:53

# 操作场景

本文介绍在本地开发 Java 应用,通过 polaris-sdk 的方式接入 Polaris(北极星),并实现可观测性功能。

# 前提条件

在开发前,请确保您已经下载并安装了 Java 和 Maven。

## 操作步骤

## 步骤1: 引入北极星依赖

## 1. 引入 polaris sdk 依赖

修改应用根目录下的 pom.xml ,为 polaris-java 添加 dependencyManagement :



#### () 说明:

polaris-sdk 版本信息请参见:版本信息。

## 2. 引入 polaris starter

```
<dependencies>
<dependency>
<groupId>com.tencent.polaris</groupId>
<artifactId>polaris-all</artifactId>
</dependency>
</dependencies>
```

# 步骤2:添加北极星配置文件 polaris.yml

- 1. 在项目的 main/resources 目录下创建 polaris.yml 文件用于初始化 polaris-java SDK。
- 2. 在 polaris.yml 文件中配置应用名、polaris (北极星)服务端地址等信息。服务端地址查看详细参见:引擎管理 > 客户端访问地址。





# 畑 还: 定 省 后 用 上 扳
enable: true
plugin:
prometheus:
type: push # 描述: 设置 pushgateway 的地址, 仅 type == push 时生效 # 地址需要替换成您创建的北极星引擎的客户端访问地址。 address: 127.0.0.1:9091 #描述:设置metric数据推送到pushgateway的执行周期 #范围:[1s:],默认值:10s pushInterval: 10s

更多 polaris.yml 配置信息,请参见 default-config.yml。

## 步骤3:应用开发,上报监控指标

#### 发起调用,并上报请求调用结果。

1. 初始化 ConsumerAPI SDK 实例:

```
import com.tencent.polaris.factory.api.DiscoveryAPIFactory;
public static void main(String[] args) throws Exception {
    ConsumerAPI consumerAPI = DiscoveryAPIFactory.createConsumerAPI();
}
```

2. 设置服务调用结果信息:

```
public enum RetStatus {
    // 服务调用成功
    RetSuccess,
    // 服务调用起时
    RetFail,
    // 服务调用超时
    RetTimeout,
}
ServiceCallResult result = new ServiceCallResult();
// 设置被调服务所在命名空间
result.setNamespace(String namespace);
// 设置被调服务的服务信息
result.setService(String service);
// 设置被调定例
result.setEndCode(String code);
// 设置本次请求的前起
result.setDel2(String delay);
// 设置本次请求的相对标签。格式为 key=value;key=value;
result.setLeStatus(RetStatus status);
// 设置本次请求调用的方法
result.setMetDel3(String labels);
// 设置本次请求调用的方法
```

## 3. 上报请求调用结果:

您在根据请求调用情况对 ServiceCallResult 结构体完成初始化后,只需要调用 ConsumerAPI.updateServiceCallResult 方法即可完成请求调用结果上报。SDK 内部会根据上报的调用结果信息,将其转换为相应的流量调用指标数据,上报至 prometheus。



onsumerAPI.updateServiceCallResult(ServiceCallResult)



# 二次寻址

最近更新时间: 2025-04-17 16:00:53

# 操作场景

本文针对用户接入微服务引擎托管的Polaris(北极星)实例希望使用北极星客户端的二次寻址能力。

#### △ 警告:

- 如果您使用的是腾讯自研上云(OA 版)北极星 SDK,由于 OA 版 SDK 默认开启了二次寻址机制(且无法关闭),请参见本文档接入。
- 当前仅针对腾讯自研上云(OA版)北极星 SDK 的注册发现能力做了兼容性验证,其他治理能力无兼容性保证。

# 接入机制说明

# 北极星 SDK 二次寻址

北极星 SDK 的二次寻址机制依赖 polaris.discover 以及 polaris.healthcheck 这2个服务。因此需要在北极星引擎实例创建出来后,需在控制台手动创 建 polaris.discover 以及 polaris.healthcheck 这两个服务。

## 为何需要用户手动创建

用户接入方式存在多 VPC、公网接入两类场景。不同的接入侧用户需要看到不同的服务端 IP,无法自动创建相关北极星系统服务。

## 前提条件

已创建 Polaris (北极星)实例,请参见 创建 Polaris (北极星)。

# 操作步骤

- 1. 登录 TSF 控制台。
- 2. 在左侧导航栏选择Polaris(北极星),进入实例引擎列表页。
- 3. 单击目标引擎的"ID",进入基本信息页面。
- 4. 根据实例信息的控制台。

## 引擎实例未开启客户端公网访问

创建 polaris.discover 以及 polaris.healthcheck 服务



1. 在实例的基本信息页面中,记住访问地址中的每个 VPC 内网 IP 地址以及访问端口。

Polaris Serverù	5问地址
客户端接入端口	注册发现 8091 (service-grpc)
	8092 (service-trpc)
	配置管理 8093 (config-grpc)
	监控上报 9091 (pushgateway)
OpenAPI 端口	8090 (http)
协议兼容端口	7779 (I5) 8761 (eureka) 15010 (xdsv3)
	15020 (xdsv2)
内网负载均衡 🛈	
	私 子网 内 访问控制 操作
	sh-gz gz-4( 1 🖍 删除
	添加内网负载均衡
公网负载均衡 🛈	- 开启

2. 针对每一个内网 VPC 的 IP 地址,在命名空间 Polaris 下创建服务 polaris.discover-{vpc 名称}以及 polaris.healthcheck-{vpc 名称}。

服务列表 新建 司法							
_ #55							
polaris.monitor				polaris			
polaris.checker							
	,	新建服务					×
	Ŕ	命名空间•	Polaris		v		
	В	服务名·	polaris.discover-pk-gz	服务名推荐格式	t: polaris.disc	over-{vpc 名称	R}
	1 E	部门					
	1	业务					
	8	服务标签()	标签键	标签值	操作		
				无标签			
			添加标签				
	I	苗述	长度不超过1024个字符				
				提交美国	I		

3. 单击服务 polaris.discover-{vpc 名称},进入到服务的实例页面,根据对应上述第一步所记录的 VPC 内网 IP 地址以及端口协议 grpc、trpc 分别创 建实例。



polaris.discover-pk-gz (Polaris)						
服务实例						
				923	東状态 て	
	新建服务实	PI				×
	17-811D •	10.0 >	值写对应 VP	C的内网IP地址		
	× our .					
	端口•	8091	填写对应协议的	端口		
	权重・	- 100	+			
	协议	grpc	填写端口协议,	grpc 或者 trpc		
	版本					
	实例标签①	107-108-108	振等	a	10-71	
			101444	± 1=26		
				JUMIZ		
		20.30标查				
	健康状态					
	开启健康检查	3				
	是否隔离③					
			提交	关闭		

4. 单击服务 polaris.healthcheck-{vpc 名称},进入到服务的实例页面,根据对应上述第一步所记录的 VPC 内网 IP 地址以及端口协议 grpc、trpc 分 别创建实例。

← polaris.healthcheck-pk-gz (Polaris)					
服务实例					
all Alexin					
2.66 回避 110 110 110 110 110 110 110 110 110 11				健康状态 ▼	
			智无数据		
A 0 #	新建服务实例				×
	实例IP •	10、 2 填写对应 VPC	的内网IP地址		
	端口•	8091 填写对应协议如	着口		
	权重•	- 100 +			
	协议	grpc 填写端口协议,	grpc 或者 trpc		
	版本				
	实例标签①	标签键	标签值	操作	
			无标签		
		添加标签			
	健康状态				
	开启健康检查①				
	是否隔离①				
			提交 关闭		

5. 按照上述步骤的操作,最终服务 polaris.discover-{vpc 名称} 的实例列表如下:

ſ	← polaris.discover-pk-gz (Polar 服务条例	ris)									
	<b>82</b> 88 2697									10010-88	Q 0 0
	_ R#P	HCI.	1972	18.0	6.B	● 単成化の ▼	用用状石 ▼	2089/4	10202020	85	
	· 0	8092	abo		100		不得满	2022-08-18 12:30:33	2022-03-10 12:30:33	692.809	
	+ 1012	8091	gre		100		不能商	2022-03-18 12:30:25	2022-03-18 12:30:25	658 B78	
	P1 2 15								20 × (t) /	S + 1	/1页 > ×

6. 按照上述步骤的操作,最终服务 polaris.healthcheck-{vpc 名称} 的实例列表如下:

e polaris.healthcheck-pk-ga	polaris.healthcheck-pk-gz (Polaris)										
服务实例											
RU 88 3889									10.00 C 10.00	9.00	
. R8P	1813	942	8.8	0.0	SERVICE T	RANS T	00824141	183331/14	90		
→ 🗆 K	8092	tpc		100	12.0	788	2022-00-18 13:00:14	2022-02-18 13:02:14	68 89		
> [] 16. 2	8091	grpc		100	-	不現真	2022-00-16 13:03:02	2022-08-18 13:03:02	101 AU		
共主告								20 + 1	1/X × 1	/1₿ → ×	



## ▲ 注意:

如果只是单 VPC 接入的场景,则无需参考 polaris.discover-{vpc 名称} 这种命名规则,直接使用 polaris.discover 即可, polaris.healthcheck-{vpc 名称} 也直接使用 polaris.healthcheck 即可。

#### 创建路由规则

#### 创建路由规则示例

polaris.discover-{vpc 名称}

服务安利 路森规则 熔板规则 限度规则 服务信息				
STRACE SANN				
88 88				0
当以下服务调用本服务时,遵守下列路由规则				
#82H	846	82482	18/5	
- 25	25	protocol/type	/ 亩 +	
如果迷水场馆后起,彼仅重和元光场港自到以下实例分明				
220101 01222 Points 80 point.discover 20002 protecting	the too this o shime Files			
v 28	28	protocotgrpc	と言う	
刘果康孝相签匹配,经权重和元先成准合则以下实例分组				
RECEI DEEN Pairs AS poirs.decover RESE protocolgap	5 N.H. 100 N.N.N. 0 M.S.N.N. FILM.			
共 2 余				10 × ±/1 × × 1 /12 × H

#### polaris.healthcheck-{vpc 名称}

服务实利 <b>路由规则</b> 均新规则 能流规则 服务信息			
STATES STATES			
NENATI <b>HANNI 1NNI</b>			
and max			•
当以下最贵调用本服务时,遵守下列路由规则			
#82H	848	B2018	86
▼ 2年	25	profaceships	2 ± +
如果很多场性后起,按权量和优先场通信的以下实例分布			
2010/01 0222 Points 32 points.docover 20102 protocol.type	112 100 1112 0 11128 <b>788</b>		
v 2部	28	protocolgrpc	/ 音 +
如果需求和语言是、极权重的优先依然会们以下发明分组			
SRUE1 DEEX Point AS paint.decover SRUE protocol.ppc	52 900 0.510 0 million <b>7688</b>		
共主张			10 × ⊕/Ξ × < 1 /1Ξ > ×

#### polaris-java 接入相关配置文件说明

• polaris.yaml 中按照下面示例进行配置二次寻址:

global:
serverConnector:
addresses:
- '{ <b>对应</b> vpc 的内网接入地址}:8091'
system:
discoverCluster:
namespace: Polaris
<b>service:</b> 'polaris.discover-{vpc <b>名称</b> }'
sameAsBuiltin: false
healthCheckCluster:
namespace: Polaris
<b>service:</b> 'polaris.healthcheck-{vpc <b>名称</b> }'
sameAsBuiltin: false

#### spring-cloud-tencent 接入相关配置文件说明

• 修改 bootstrap.yaml 或者 bootstrap.properties 文件:

spring:		
cloud:		
polaris:		
address:	' {对应	的内网接入地址}:8091'

• 在 resources 下创建 polaris.yml 文件。



## • 按照下面示例进行配置二次寻址:

lobal:
system:
#服务发现集群
discoverCluster:
namespace: Polaris
<b>service: '</b> polaris.discover-{vpc <b>名称</b> }'
sameAsBuiltin: false
#健康检查集群
healthCheckCluster:
namespace: Polaris
<b>service:</b> 'polaris.healthcheck-{vpc <b>名称</b> }'
sameAsBuiltin: false

#### trpc-java 接入相关配置文件说明

```
plugins:
    registry:
    polaris:
    heartbeat_interval: 3000
    protocol: grpc
    address_list: '(对应 vpc 的内网接入地址):8091'
    global:
        system:
        discoverCluster:
        namespace: Polaris
        service: 'polaris.discover-{vpc 名称}'
        sameAsBuiltin: false
        healthCheckCluster:
        namespace: Polaris
        service: 'polaris.healthcheck-{vpc 名称}'
        sameAsBuiltin: false
selector:
    polaris:
    protocol: grpc
    address_list: '{对应 vpc 的内网接入地址}:8091'
    global:
        system:
        discoverCluster:
        namespace: Polaris
        service: 'polaris.discover-{vpc 名称}'
        sameAsBuiltin: false
    healthCheckCluster:
        namespace: Polaris
        service: 'polaris.discover-{vpc 名称}'
        sameAsBuiltin: false
    healthCheckCluster:
        namespace: Polaris
        service: 'polaris.healthcheck-{vpc 名称}'
        sameAsBuiltin: false
    healthCheckCluster:
        namespace: Polaris
        service: 'polaris.healthcheck-{vpc 名称}'
        sameAsBuiltin: false
    healthCheckCluster:
        namespace: Polaris
        service: 'polaris.healthcheck-{vpc 名称}'
        sameAsBuiltin: false
    }
}
```

# 引擎实例开启客户端公网访问

创建 polaris.discover 以及 polaris.healthcheck 服务



1. 在实例的基本信息页面中,记住访问地址中的公网 IP 地址以及访问端口。

	8090/nttp 8091/grpc, 8092/th	pc、7779/I5、15020/xds(v2)、8761/eureka	
内网地址	私有网络	子网	内网地址

2. 在命名空间 Polaris下创建服务 polaris.discover 以及 polaris.healthcheck。

服务列表					
8658 BILLS					
□ 服务=					健康实例/总实例
					2/2
polaris.monitor	Polaris			polaris	0/0
	POD				3/3
		新建服务			×
		命名空间。	Polaris	Ŧ	]
		服务名•	polaris.discover		]
		部门			]
		业务			]
		服务标签()	标签键	标签值	操作
				无标签	
			源加标签		
		描述	长度不超过1024个字符		
				提交 关闭	-
				2000 - 20	



3. 单击服务 polaris.discover,进入到服务的实例页面,根据客户端公网访问 IP 地址以及端口协议 grpc、trpc 分别创建实例。

← polaris.discover (Polaris)					
				健康状态 下	周离状
共 0 条	新建服务实例				×
	实例IP•	16° 1 J. 510'	填写客户端访	问的公网IP地址	
	端口・	8091 填写网	寸应协议端口		
	权重·	- 100 +			
	协议	grpc 填写端	i口协议,grpc	:或者 trpc	
	版本				
	实例标签③	<b>新安</b> 時	經签值	10-11	
		7931.000 ME	107-334 MR	无标签	
		酒加标签			
	健康状态				
	<b>井后雄康检查()</b>				
	本日時間( <b>(</b> )				
			提交	关闭	

4. 单击服务 polaris.healthcheck,进入到服务的实例页面,根据客户端公网访问 IP 地址以及端口协议 grpc、trpc 分别创建实例。

服务实例							
NIE BIG HOUSE							
					9	康状态 〒	隔离状态
					誓无數据		
		新建服务实例					×
		実例IP・	1 ,6	填写客户端访问	间公网 IP 地址		
		180 ·	8091 填写	对应协议端口			
		权重•	- 100 +				
		物议	grpc 填写剪	<b>端口协议</b>			
		版本					
		实例标签①	标签键	标签值		操作	
					无标签		
			添加标签				
		健康状态					
		开启健康检查①					
		是否隔离①					
				提交	关闭		

5. 按照上述步骤的操作,最终服务 polaris.discover 的实例列表如下:

<ul> <li>polaris.discover (Polaris)</li> </ul>										
服务实例										
<b>50</b> 80 Reso									10.002 <b>919</b>	0 0
— RMP	端口	19-17	85.7	618.	STRACT	用用のの T	8182334	10207314	30.17	
>	8092	tepo		100	10	不能用	2022-03-18 12:52:02	2022-03-18 12:52:02	A11 313	
× 🗆 – 1.196	8091	910		100	-	不稱酒	2022-03-18 12:51:55	2022-03-18 12:51:55	84 88	
共 2 张								20 × ±/Ξ	8 4 1 71	ă → ×

6. 按照上述步骤的操作,最终服务 polaris.healthcheck 的实例列表如下:

polaria.baathchuck (Polarin)										
服务实例										
RR 25.80									10.01/2- <b>80</b>	0 0
2 RMP	NC .	897	15.0	638	BRUS T	時用状の <b>T</b>	1010291/41	#331/H	1815	
> [] 1 12	8092	trpc		100	22	不能着	2022-02-10 10:10:16	2022-03-18 13:10:15	658 858	
> 🗆 - 8.1	8091	9900		100		不稱義	2022-03-18 13:10:09	2022-03-18 13:10:09	-	
共 2 条								20 -	£/∏ × + 1	/1百 > ×

#### 创建路由规则


# 🔗 腾讯云

#### polaris.discover

服务安利 <b>路由规则</b> 给纸纸刷 用流规则 服务信息				
STRACT BRANN JOINT HRON 13841				
88 88				0
当以下服务੍備用本服务时,遵守下列路由规则				
#82H	8.9 G	83485	85	
- 2F	25	protocol/type	2 ± +	
如果迷水场的后起,我双重和九九场港自我以下实例分布				
20101 0122 Points 30 point.doover 20102 protocoling	1 10 100 1.1.1.0 0 2.5.1.1 <b>7.18.8</b>			
* 28	2.05	protocolgrpc	/ 音 +	
如果课术经验匹配、最权需和优先级提出到以下实例分组				
RENET OFER Pairs an polytic-decover RENET protocolgyp	6 15月 100 1551 0 15511 <b>不福田</b>			
共主张			19×余/页 × × 1 月1页)	н

### polaris.healthcheck

接旁尖利 <b>路由规则</b> 熔新规则 能流规则 服洗规则 服务信息				
N885.				
NORAZ <b>HRACH 10000</b>				
RIR HIR				•
当以下服务੍調用本服务时、遵守下列路由规则				
#82H	8 A 8	834025	85	
- 2F	25	protocol/type	Z 🗄 +	
22果康孝师馆后配, 按权重和沈光级源应到以下实例分明				
201011 0222 Points 320 polars.discover 20102 protocol.type	HE 100 UNE & SHEE FIELD			
v 28	28	protocot.grpc	/ 亩 +	
如果能够和语言是、我权能和优先成准由则以下实例分组				
RRIGET DEEX Paaris AS polaris.discover RRIEE protocol.pps	SEE 100 CAR D REAR FIRM			
共 2 条				10×⊕/Ξ × × 1 /1Ξ > ×

### polaris-java 接入相关配置文件说明

• polaris.yaml 中按照下面示例进行配置二次寻址:

global:		
serverConnector:		
addresses:		
- '{ <b>对应公网接入地址</b> }:8091'		
system:		
discoverCluster:		
namespace: Polaris		
service: polaris.discove:		
sameAsBuiltin: false		
healthCheckCluster:		
namespace: Polaris		
service: polaris.healthc	neck	
sameAsBuiltin: false		

### spring-cloud-tencent 接入相关配置文件说明

• 修改 bootstrap.yaml 或者 bootstrap.properties 文件。

spring:	
cloud:	
polaris:	
address:	' { <b>对应公网接入地址</b> } : 8091 '

- 在 resources 下创建 polaris.yml 文件。
- 按照下面示例进行配置二次寻址:

```
global:
system:
discoverCluster:
namespace: Polaris
service: polaris.discover
sameAsBuiltin: false
```



healthCheckCluster

namespace: Polaris service: polaris.healthcheck

### trpc-java 接入相关配置文件说明

```
plugins:
  registry:
    polaris:
    heartbeat_interval: 3000
    protocol: grpc
    address_list: '{对应公网接入地址}:8091'
    global:
        system:
        discoverCluster:
            namespace: Polaris
            service: polaris.discover
            sameAsBuiltin: false
        healthCheckCluster:
            namespace: Polaris
            service: polaris.healthcheck
            sameAsBuiltin: false
    selector:
    polaris:
    protocol: grpc
    address_list: '{对应公网接入地址}:8091'
    global:
        system:
        discoverCluster:
        namespace: Polaris
        service: polaris.discover
        sameAsBuiltin: false
    healthCheckCluster:
        namespace: Polaris
        service: polaris.discover
        sameAsBuiltin: false
    healthCheckCluster:
        namespace: Polaris
        service: polaris.healthcheck
        sameAsBuiltin: false
```

# Go 应用开发 Polaris-Go 接入

最近更新时间: 2025-04-17 16:00:53

# 操作场景

本文通过一个 demo 进行 Golang 应用接入 Polaris(北极星)的全流程操作演示,帮助您快速了解如何使用北极星网格。

# 前提条件

- 已创建 Polaris (北极星),请参见 创建 Polaris (北极星)。
- 下载 Github 的 demo 源码 到本地并解压。
- 本地编译构建打包机器环境已安装了 golang 环境,并且能够访问 Github。
- 根据您自身的业务,已准备好业务部署的资源,虚拟机部署和容器化部署选择其中一种方式即可。
  - 虚拟机部署已创建 CVM 虚拟机,请参见 创建 CVM 虚拟机。
  - 容器化部署已创建 TKE 容器集群,请参见 创建 TKE 集群。

# 操作步骤

- 1. 登录 TSF 控制台。
- 2. 在左侧导航栏选择Polaris(北极星),进入北极星实例列表页。
- 3. 单击目标引擎的"ID",进入基本信息页面。
- 4. 查看访问地址,Golang 应用访问使用 gRPC 端口(8091):

8090/http、8091/grpc、8092/trpc、7779	3/15、15020/xds(v2)、8761/eureka		
私有网络	子网	内网地址	
pk-gz-vpc(vpc- ) 🖸	polaris-dev(subnet-	1 .6	
	8090/http、8091/grpc、8092/trpc、7775 私有网络 pk-gz-vpc(vpc) Ľ	8090/http、8091/grpc、8092/trpc、7779/l5、15020/xds(v2)、8761/eureka 私有网络   子网 pk-gz-vpc(vpc- ) 译   polaris-dev(subnet) 译	8090/http、8091/grpc、8092/trpc、7779/l5、15020/xds(v2)、8761/eureka 私有网络

### 5. 修改 demo 中的注册中心地址

- 5.1 在下载到本地的 demo 源码 目录下,分别找到 quickstart/consumer/polaris.yaml 以及 quickstart/provider/polaris.yaml 文 件。
- 5.2 添加微服务引擎北极星网格地址到项目配置文件中(这里以 quickstart/consumer/polaris.yaml 为例)。

global:			
serverConnector:			
addresses:			

### 6. 将源码编译成可执行程序。

- 6.1 分别在 consumer 和 provider 这2个目录下,打开 cmd 命令,执行以下命令,对项目进行编译:
- 编译 consumer: CGO\_ENABLED=0 go build -ldflags "-s -w" -o consumer
- 编译 provider: CGO\_ENABLED=0 go build -ldflags "-s -w" -o provider
- 6.2 编译成功后,生成如下表所示的2个二进制包。



软件包所在目录	软件包名称	说明
\examples\quickstart\provider	provider	服务生产者
\examples\quickstart\consumer	consumer	服务消费者

6.3 分别将 consumer 以及 provider 的二进制上传到不同的 CVM 实例中,这里假定上传的路径均为/data/polaris/golang\_examples。

- 7. 部署 provider 和 consumer 微服务应用,根据您业务实际选择虚拟机部署、容器化部署中的一种部署方式即可。
  - 7.1 虚拟机部署
  - 上传二进制文件以及配置文件 (polaris.yaml) 至 CVM 实例。
  - 执行启动命令进行启动:

./[二进制文件名称]

### 7.2 容器化部署

○ 编写 dockerfile 生成镜像,参考:

```
FROM golang:alpine
WORKDIR /root
ADD . /root
ENTRYPOINT ./[二进制名称] [启动参数命令]
```

○ 通过 TKE 部署并运行镜像。

8. 确认部署结果

- 8.1 进入前面提到的微北极星网格实例页面。
- 8.2 选择服务管理 > 服务列表, 查看微服务 EchoServerGolang 的实例数量。
- 若实例数量值不为0,则表示已经成功接入微服务引擎。
- 若实例数量为0,或者找不到 EchoServerGolang 服务名,则表示微服务应用接入微服务引擎失败。

▶ EchoServerGolang default - - 1/1 2021-12-16 16:01:06 2021-12-16 16:01:06

8.3 调用 consumer 的 HTTP 接口。执行 http 调用,其中 \${app.port} 替换为 consumer 的监听端口(默认为18080), \${add.address} 则替换为 consumer 暴露的地址。

curl -L -X GET 'http://\${add.address}:\${app.port}/echo' 预期返回值: Hello, I'm EchoServerGolang Provider



# gRPC-Go 接入

最近更新时间: 2025-04-17 16:00:53

# 操作场景

本文通过一个 demo 进行 gRPC-Go 应用接入Polaris(北极星)的全流程操作演示,帮助您快速了解如何使用北极星网格。

## 前提条件

- 已创建 Polaris (北极星),请参见 创建 Polaris (北极星)。
- 下载 Github 的 demo 源码 到本地并解压。
- 本地编译构建打包机器环境已安装了 Go,并且能够使用 Go mod 拉取依赖。
- 根据您自身的业务,已准备好业务部署的资源,虚拟机部署、容器化部署和 TEM 部署选择其中一种方式即可。
  - 虚拟机部署已创建 CVM 虚拟机,请参见 创建 CVM 虚拟机。
  - 容器化部署已创建 TKE 容器集群,请参见 创建 TKE 集群。
  - TEM部署已创建 TEM 环境,请参见 创建 TEM 环境。

# 操作步骤

- 1. 登录 TSF 控制台。
- 2. 在北极星网格下的 polarismesh 页面,单击页面左上方下拉列表,选择目标地域。
- 3. 单击目标引擎的"ID",进入基本信息页面。
- 4. 查看访问地址,gRPC-Go应用访问使用 gRPC 端口(8091):

访问地址			
端口	8090/http、8091/grpc、80	)92/trpc、7779/15、15020/xds(v2)、8761/eureka	
内网地址	私有网络	子网	内网地址
	pk-gz-vpc(vpc-	) 🖸 polaris-dev(subnet- ) 🗹	1.6

### 5. 修改 demo 中的注册中心地址。

5.1 在下载到本地的 demo 源码目录 下,分别找到

5.2 添加微服务引擎北极星网格地址到项目配置文件中(以 \examples\quickstart\provider\polaris.yaml 为例)。

global:			
serverConnector:			
addresses:			
- '10.0.4.6:8091'			

### 6. 将源码编译成可执行程序。

- 6.1 分别在 consumer 和 provider 这2个目录下,打开 cmd 命令,执行以下命令,对项目进行编译:
- 编译 consumer: CGO\_ENABLED=0 go build -ldflags "-s -w" -o consumer
- 编译 provider: CGO\_ENABLED=0 go build -ldflags "-s -w" -o provider
- 6.2 编译成功后,生成如下表所示的2个二进制包。



软件包所在目录	软件包名称	说明
\examples\quickstart\provider	provider	服务生产者
\examples\quickstart\consumer	consumer	服务消费者

7. 部署 provider 和 consumer 微服务应用,根据您业务实际选择虚拟机部署、容器化部署以及 TEM 部署中的一种部署方式即可。

### 7.1 虚拟机部署

- 上传 Jar 包至 CVM 实例。
- 执行启动命令进行启动:

nohup [**二进制名称**] 🌡

### 7.2 容器化部署

○ 编写 dockerfile 生成镜像,参考:

FROM golang:alpine WORKDIR /root ADD . /root ENTRYPOINT ./[**二进制名称**]

### ○ 通过 TKE 部署并运行镜像。

### 7.3 TEM 部署

○ 选择 TEM 环境,注意所选择的环境,其依赖的 VPC,必须和上面已经创建的北极星网格实例所依赖的 VPC 一致:

基本信息	资源管理 访问管理 配置管理
基本信息	
环境名称	andr-test-1 🧪
ID	env-
所在地域	广州
集群网络	VPC( 2) 子网( 2)
创建时间	2021-12-21 20:08:21
描述	1



### ○ 在已选择的环境中,新建TEM应用,相关参数填写参考:

新建应用		×
名称 *	polaris-grpcgo-provider	
	最长为40个字符,只能包含小写字母、数字及分隔符("-"),且不能以分隔符开头或结尾	
描述 (选填)	请输入描述	
开发语言	◯ JAVA <b>○</b> 其他语言	
程序包上传方式		
镜像仓库	自动创建 选择已有仓库	
	将为您自动创建一个与应用名称相同的镜像仓库 若您是第一次使用 <b>趋</b> 像仓库,请先开通镜像仓库。 个人版 区 企业版 区	
	提交关闭	

部署应用,	相关参数填写请参考(	端口号映射,	consumer 默认端口号为16011,	provider默认端口号为16010)	:
				provider www.viiim 373100107	-

应用名	polaris-grpcgo-provider
发布环境	andr-test-1 / env-   中
镜像来源	容器镜像服务 个人版
	使用Demo镜像
镜像版本	svc-polaris-grpcgo-provider-cedzerno:v2 v
版本号	v2
▼ 资源配置	
▲ 访问配置	
访问方式	○环境内访问 VPC内网访问(四层转发) 公网访问(四层转发) 提供一个可以被环境内其他应用访问的入口,支持 TCP/UDP 协议。 如您需要配置公网的 HTTP/HTTPS 协议转发规则,可前往应用路由页面 Ⅰ 转发规则配置。
端口映射	协议 ①     容器端口 ③     应用监听端口 ③
	TCP - 16010 16010 3
	添加端口映射
▼ 启动命令设置	设置容器启动和运行时需要的命令
▼ 应用启停管理	在环境启动后和应用结束前设置处理任务,例如:环境初始化、应用优雅退出等
▼ 配置设置	
▼ 环境变量	
▼ 健康检查	
<b>▼ 持久化存储</b> 为	容器提供存储,目前支持腾讯云文件存储CFS,还需挂载到容器的指定路径中。使用前,请先前往环境关联CFS存储资源。
▼ 安全组	
▼日志配置	
部署	<b>奴消</b>



○ 查看访问路径,consumer 应用部署完后,可以在基本信息 > 访问配置中查看访问地址,如需公网访问,可以编辑并更新开启公网访问:

基本信白	(古山石)型 (615年)
<u>∞</u> +)□丞	
应用名称 polaris-grpcgo-consumer	访问方式 公网访问 (四层转发)
应用描述 -	访问地址 <b>公网IP</b>
私有网络 vpc- 🖸	转发配置 前往设置 <b>2</b>
子网 subnet- 区 (可用IP数202个)	
运行/目标实例数 <b>2/2</b>	配置信息
可用区广州三区	启动命令 <b>智未设置</b>
	启动参数 <b>智未设置</b>
弹性伸缩	环境变量 智术设置
规格 1核20	持久化存储 <b>暂未设置</b>
初始化实例数 2个 编辑并更新	日志信息 日志主题: 暫未设置   采集路径: 暫未设置
弹性伸缩 未启用 编辑并更新	配置信息 暂未设置

8. 确认部署结果。

8.1 进入前面提到的微北极星网格实例页面。

- 选择**服务管理 > 服务列表**, 查看微服务 EchoServerGRPC (provider) 的实例数量:
- 若实例数量值不为0,则表示已经成功接入微服务引擎。
- 若实例数量为0,或者找不到 EchoServerGRPC 服务名,则表示微服务应用接入微服务引擎失败。

← EchoServerGRPC (defa	ult)								
服务实例									
新建 服務 非他操作									健康状态 <b>健康</b>
	9811	协议	版本	权重	健康状态 T	隔离状态 〒	创建时间	修改时间	課作
▶ 1 21		top		100	建築	不晒南	2021-12-15 21:11:16	2021-12-15 21:11:16	编辑素种
共1条								20 🔻	∰/∏ H <

8.2 调用 consumer 的 HTTP 接口。执行 http 调用,其中 \${app.port} 替换为 consumer 的监听端口(默认为16011), \${add.address} 则 替换为 consumer 暴露的地址。





# 二次寻址使用指南

最近更新时间: 2025-04-17 16:00:53

# 操作场景

本文针对用户接入 Polaris(北极星)希望使用北极星客户端的二次寻址能力。

### △ 警告:

- 如果您使用的是腾讯自研上云(OA 版)北极星 SDK,由于 OA 版 SDK 默认开启了二次寻址机制(且无法关闭),请参见本文档接入。
- 当前仅针对腾讯自研上云(OA 版)北极星 SDK 的注册发现能力做了兼容性验证,其他治理能力无兼容性保证。

# 接入机制说明

# 北极星 SDK 二次寻址

北极星 SDK 的二次寻址机制依赖 polaris.discover 以及 polaris.healthcheck 这2个服务。因此需要在北极星引擎实例创建出来后,需在控制台手动创 建 polaris.discover 以及 polaris.healthcheck 这两个服务。

## 为何需要用户手动创建

用户接入方式存在多 VPC、公网接入两类场景。不同的接入侧用户需要看到不同的服务端IP,无法自动创建相关北极星系统服务。

# 前提条件

已创建 Polaris (北极星),请参见 创建 Polaris (北极星)。

# 操作步骤

- 1. 登录 TSF 控制台。
- 2. 在左侧导航栏选择Polaris(北极星),进入北极星实例列表页。
- 3. 单击目标引擎的"ID",进入基本信息页面。
- 4. 根据实例信息的控制台

### 引擎实例未开启客户端公网访问

### 创建 polaris.discover 以及 polaris.healthcheck 服务

1. 在实例的基本信息页面中,记住访问地址中的每个 VPC 内网 IP 地址以及访问端口。

访问地址				
端口	8090/http、8091/grpc、8092/tr	rpc、7779/I5、15020/xds(v2)、8761/eureka		
内网地址				添加接入方式
	私有网络	子网	内网地址	操作
	pk-gz-v	polaris-de	10.0.4.12	删除



2. 针对每一个内网 VPC 的 IP 地址,在命名空间 Polaris 下创建服务 polaris.discover-{vpc 名称}以及 polaris.healthcheck-{vpc 名称}。

服务列表 (1)11 (1)11				
□ 服务:				
polaris.monitor			polaris	
polaris.checker	Polaris		polaris	
	新建服务	i		×
	命名空间	Polaris	•	
	服务名•	polaris.discover-pk-gz	服务名推荐格式:polar	is.discover-{vpc 名称}
	部门			
	业务			
	服务标签(	<b>〕</b> 标签键	标签值	操作
			无标签	
		添加标签		
	描述	长度不超过1024个字符		
			提交关闭	

3. 单击服务 polaris.discover-{vpc 名称},进入到服务的实例页面,根据对应上述第一步所记录的 VPC 内网 IP 地址以及端口协议 grpc、trpc 分别创 建实例。

← polaris.discover-pk-gz (Polaris)					
服务实例					
ana Xesan					
				健康状态 下	
	新建服务实例				×
	实例IP •	10.0 2	填写对应VPC的	内网IP地址	
	端口•	8091	填写对应协议端口		
	权重・	- 100	+		
	协议	grpc	填写端口协议,grp	oc 或者 trpc	
	版本				
	实例标签①	标签键	标签值	操作	
				无标签	
		添加标签			
	健康状态				
	开启健康检查①				
	是否隔离③				
			提交	关闭	

4. 单击服务 polaris.healthcheck-{vpc 名称},进入到服务的实例页面,根据对应上述第一步所记录的 VPC 内网 IP 地址以及端口协议 grpc、trpc 分 别创建实例。



← polaris.healthcheck-pk-gz (Polaris)					
服务实例					
□ 实例IP 第□ 协议				健康状态 ▼	
# 0 #	新建服务实例				×
	实例IP •	10. 2 填写对应 VPC	2的内网 IP 地址		
			W		
	端口•	8091 項与刘应协议	而凵		
	权重・	- 100 +			
	协议	grpc 填写端口协议	,grpc 或者 trpc		
	版本				
	实例标签③	1	1- nor re-	410 Am	
		你立腿	标盘道	1991.1	
			无标签		
		添加标签			
	健康状态				
	开启健康检查①				
	是否隔离③				
			提交美術		

5. 按照上述步骤的操作,最终服务 polaris.discover-{vpc 名称} 的实例列表如下:

polaris.discover-pk-gz (Polaris)										
服务实例										
新建 删除 其他操作									健康状态: <b>健康</b>	Q \$ \$
_ 実例IP	端口	协议	版本	权重	健康状态 下	隔离状态 ▼	创罐时间	修改时间	39(1)	
» 10 .12	8092	trpc	-	100	健康	不隔离	2022-03-18 12:30:33	2022-03-18 12:30:33	编辑 删除	
» 🔲 1012	8091	grpc	-	100	健康	不隔离	2022-03-18 12:30:25	2022-03-18 12:30:25	编辑 删除	
共 2 条								20 v	奈/页 H 4 1	/1页

### 6. 按照上述步骤的操作,最终服务 polaris.healthcheck-{vpc 名称}的实例列表如下:

← polaris.healthcheck-pk-gz (Polaris)										
服务实例										
新設展線									健康状态 <b>:健康</b>	Q \$ \$
英例IP	端口	协议	版本	权重	健康状态 下	隔离状态 ▼	创建时间	修改时间	操作	
► 11.	8092	trpc	-	100	92.0R	不隔离	2022-03-18 13:03:14	2022-03-18 13:03:14	sasa max	
» 🗌 10. 2	8091	grpc	-	100	经承	不隔离	2022-03-18 13:03:02	2022-03-18 13:03:02	编辑 删除	
共 2 泰								20 * 条/3	H < 1 /13	Σ ► H

### ▲ 注意:

如果只是单 VPC 接入的场景,则无需参考 polaris.discover-{vpc 名称} 这种命名规则,直接使用 polaris.discover 即可, polaris.healthcheck-{vpc 名称} 也直接使用 polaris.healthcheck 即可。

### 创建路由规则

### 创建路由规则示例

polaris.discover-{vpc 名称}



服务实例 路由規則 熔断规则 限流规则 服务信息			
<b>NUMAT</b> (MANA) NUMAT (NUMAT)			٥
当以下服务调用本服务时,遵守下列路由规则			
命名空间	服务名	请求标签	操作
▼ 全部	全部	protocol:trpc	/ 亩 +
如果请求标签匹配,按权重和优先级路由到以下实例分组			
实例分组1 命名空间 Polaris 服务 polaris.discover 实例标签 protocol.trpc	权量 100 优先级 0 是否隔离 <b>不强离</b>		
▼ 全部	全部	protocol:grpc	/ 亩 +
如果请求标签匹配,按权重和优先级路由到以下实例分组			
实例分组1 命名空间 Polaris 服务 polaris.discover 实例标签 protocolgrpu	5 校重 100 优先级 0 是否隔离 不强高		
共 2 条			10 - 余/页 🛛 🖌 🤘 1 🔢 7 1 英 🔺 😕

### polaris.healthcheck-{vpc 名称}

服务实例 路由规则 熔断规则 限流规则 服务信息			
NANKST <b>BRANN</b> NANKST <b>KRADI İYA</b> R			
新建 建交			¢
当以下服务调用本服务时,遵守下列路由规则			
命名空间	服务名	请求标签	38:17
▼ 全部	全部	protocol:trpc	/ 亩 +
如果请求标签匹配,按权重和优先级路由到以下实例分组			
突到分组! 命名空间 Polaris 服务 polaris.discover 实例标签 protocol.trp	5 权重 100 优先级 0 是否隔离 <b>不强离</b>		
▼ 全部	全部	protocol:grpc	/ 由 +
如果请求标签匹配,按权重和优先级路由到以下实例分组			
実例分振! 命名空间 Polaris 服务 polaris.discover 実例标签 protocot.grp	c 权重 100 优先级 0 是否隔离 <b>不隔离</b>		
共 2 条			10 - 余/页 🛛 🖌 1 71页 > 米

### polaris-go 接入相关配置文件说明

### • polaris.yaml中按照下面示例进行配置二次寻址:

```
global:
serverConnector:
addresses:
   - '{对应 vpc 的内网接入地址}:8091'
system:
discoverCluster:
   namespace: Polaris
   service: 'polaris.discover-{vpc 名称}'
healthCheckCluster:
   namespace: Polaris
   service: 'polaris.healthcheck-{vpc 名称}'
```

### trpc-go 接入相关配置文件说明

```
plugins:
    registry:
    polaris:
        heartbeat_interval: 3000
        protocol: grpc
        address_list: '{对应 vpc 的内网接入地址}:8091 '
        cluster_service:
        discover: 'polaris.discover-{vpc 名称}'
        health_check: 'polaris.healthcheck-{vpc 名称}
selector:
        polaris:
        address_list: '{对应 vpc 的内网接入地址}:8091'
```



```
protocol: grpc
pluster_service:
______discover: 'polaris.discover-{vpc 名称}'
_______health_check: 'polaris.healthcheck-{vpc 名称}'
```

# 引擎实例开启客户端公网访问

### 创建 polaris.discover 以及 polaris.healthcheck 服务

1. 在实例的基本信息页面中,记住访问地址中的公网 IP 地址以及访问端口。

<b>访问地址</b> <sup>端口</sup>	8090/http 8091/grpc、8092	/trpc、7779/15、15020/xds(v2)、8761/eurek	Ka la
内网地址			
	私有网络	子网	内网地址
公网地址 🛈	14 7.12 美闭		

2. 在命名空间 Polaris下创建服务 polaris.discover 以及 polaris.healthcheck。

							_
服务列表							
□ 服务							
	Polaris						
polaris.monitor	Polaris						
	Pola						
		新建服务				×	
		命名空间・	Polaris	Ŧ			
		服务名•	polaris.discover				
		部门					
		业务					
		服务标签(	标签键	标签值	操作		
				无标签			
			添加标签				
		描述	长度不超过1024个字符				
				提交关闭			



3. 单击服务 polaris.discover,进入到服务的实例页面,根据客户端公网访问 IP 地址以及端口协议 grpc、trpc 分别创建实例。

← polaris.discover (Polaris)					
服务实例					
				健康状态 ▼	
			智无数期	5	
	新建服务实	列			×
	实例IP•	it**	写客户端访问的公网	IP地址	
	• L1 m/		が収端口		
	权重•	- 100 +			
	协议	grpc 填写端口	办议,grpc 或者 trpo	5	
	版本				
	实例标签①	标签键	标签值	操作	
			无标签		
		源加标签			
	健康状态				
	开启健康检查	I I I I I I I I I I I I I I I I I I I			
	是否隔离()				
			提交关闭		

4. 单击服务 polaris.healthcheck,进入到服务的实例页面,根据客户端公网访问IP地址以及端口协议 grpc、trpc 分别创建实例。

← polaris.healthcheck (Polaris)		
服务实例		
其例 P		本 权量 <b>健康状态 T</b> 隔高状态
		智无数据
共日条	新建服务实例	×
	实例IP・	1 人名 填写客户端访问公网 IP 地址
	F	8091 填写对应协议端口
	权重•	- 100 +
	协议	grpc 填写端口协议
	版本	
	实例标签()	标签键 标签值 操作
		无标签
		透加柳签
	健康状态	
	开启健康检查	
	是否隔离()	
		提交 关闭



### 5. 按照上述步骤的操作,最终服务 polaris.discover 的实例列表如下:

polaris.discover (Polaris)										
服务实例										
新線 劉除 其他操作									註康状态: <b>健康</b>	Q \$ \$
实例IP	踏口	协议	版本	权重	健康状态 ▼	隔离状态 ▼	创建时间	像改时间	操作	
>	8092	trpc		100	建原	不隔离	2022-03-18 12:52:02	2022-03-18 12:52:02	10140 HUR	
► 106	8091	grpc		100	健康	不隔离	2022-03-18 12:51:55	2022-03-18 12:51:55	编辑 删除	
共 2 条								20 * 亲/页	H - 1 /1	<u></u>

### 6. 按照上述步骤的操作,最终服务 polaris.healthcheck 的实例列表如下:

← polaris.	healthcheck (Polari	s)									
服务实例											
新建	制牌 其他操作									健康状态: <b>健康</b>	ο φ φ
实例IP		端口	协议	版本	权重	健康状态 下	隔离状态 下	创建时间	修改时间	損作	
→ 🗌 1	13.	8092	trpc		100	建康	不陽离	2022-03-18 13:10:16	2022-03-18 13:10:16	16111 B109	
•	3.1(	8091	grpc		100	\$ <b>2</b> /0.	不隔离	2022-03-18 13:10:09	2022-03-18 13:10:09	1015R (B108	
共 2 条									20 +	奈/页 H 4 1	/1页 → H

### 创建路由规则

### polaris.discover

服务实例 路由规则 熔断规则 限流规则 服务信息			
NINKIT <b>BANN</b> NINKIT <b>BANN</b> NINKIT <b>BANKON ENKER</b>			
新建			¢
当以下服务调用本服务时,遵守下列路由规则			
命名空间	服务名	请求标签	操作
▼ 全部	全部	protocol:trpc	/ 亩 +
如果请求标签匹配,按权重和优先级路由到以下实例分组			
实例分册1 命名型间 Polaris 服务 polaris.discover 实例标签 protocol.trp	5 校重 100 优先级 0 是否隔离 不强离		
▼ 全部	全部	protocol:grpc	/ 亩 +
如果请求标签匹配,按权重和优先级路由到以下实例分组			
実例分冊1 命名空间 Polaris 服务 polaris.discover 支例标签 protocot.grp	c 权重 100 优先级 0 是否隔离 不确直		
共 2 条			10 - 余/页 🛛 🖌 1   /1页 > 米

### polaris.healthcheck

服务实例 路由规则 熔断规则 限流规则 服务信息			
19.19.72 ER 19.19 19.19.72 <b>ER 19.19</b> 19.19.72 <b>ER 19.19</b>			
新建 提交			φ
当以下服务调用本服务时,遵守下列路由规则			
命名空间	服务名	请求顺签	操作
▼ 全部	全部	protocol.trpc	/ 亩 +
如果请求标签匹配,按权重和优先级路由到以下实例分组			
实例分组1 命名空间 Polaris 服务 polaris.discover 实例标签 protocoltrpc	权重 100 优先级 0 是否隔离 <b>不隔离</b>		
- 全部	全部	protocol:grpc	/ 亩 +
如果请求标签匹配,按权重和优先级路由到以下实例分组			
実例分冊1 命名空间 Polaris 服务 polaris.discover 実例短望 protocol.gppc	校重 100 优先级 0 是否隔离 <b>不隔离</b>		
共 2 条			10 * 条/页   × ( 4 1   /1页

### polaris-go 接入相关配置文件说明

polaris.yaml 中按照下面示例进行配置二次寻址:

```
global:
serverConnector:
addresses:
- '{对应公网接入地址}:8091'
system:
discoverCluster:
```



namespace: Polaris

service: polaris.discover

nealthcheckCluster:

corrigo, polaria boalthebook

### trpc-go 接入相关配置文件说明

```
plugins:
    registry:
    polaris:
        heartbeat_interval: 3000
        protocol: grpc
        address_list: '{对应公网接入地址}:8091'
        cluster_service:
        discover: polaris.discover
        health_check: polaris.healthcheck
selector:
    polaris:
        address_list: '{对应公网接入地址}:8091'
        protocol: grpc
        cluster_service:
        discover: polaris.discover
        health_check: polaris.healthcheck
```



# 就近访问使用指南

最近更新时间: 2025-04-17 16:00:53

# 操作场景

本文通过一个 demo 进行 Golang 应用接入Polaris(北极星)的全流程操作演示,帮助您快速了解如何使用北极星网格的就近路由能力。

# 前提条件

- 已创建 Polaris (北极星)实例,请参见 创建 Polaris (北极星)。
- 下载 Github 的 demo 源码 到本地并解压。
- 本地编译构建打包机器环境已安装了 golang 环境,并且能够访问 Github。
- 根据您自身的业务,已准备好业务部署的资源,虚拟机部署和容器化部署选择其中一种方式即可。
  - 虚拟机部署已创建 CVM 虚拟机,请参见 创建 CVM 虚拟机。
  - 容器化部署已创建 TKE 容器集群,请参见 创建 TKE 集群。

# 操作步骤

- 1. 登录 TSF 控制台。
- 2. 在左侧导航栏选择Polaris(北极星),进入北极星实例列表页。
- 3. 单击目标引擎的"ID",进入基本信息页面。
- 4. 查看访问地址,Golang 应用访问使用 gRPC 端口(8091):

访问地址			
端口	8090/http、8091/grpc、8092/trpc、7779/l5、150	)20/xds(v2)、8761/eureka	
内网地址	私有网络	子网	内网地址
	pk-gz-vpc(vpc- )	polaris-dev(subnet-	1.6

### 5. 修改 demo 中的注册中心地址:

5.1 在下载到本地的 demo 源码 目录下,分别找到 nearby/consumer/polaris.yaml 以及 nearby/provider/polaris.yaml 文件。

5.2 添加微服务引擎北极星网格地址到项目配置文件中(这里以 nearby/consumer/polaris.yaml 为例)。

global: serverConnector: addresses:
serverConnector: addresses:
addresses:
- 10.0.4.6:8091
location:
# 设置 polaris-go 进程地理信息的提供插件为 腾讯云 插件
# 如果当前进程在CVM或者TKE中,则能够自动获取当前进程所在的位置信息
#
# 如果自动获取失败,则可以设置插件为 env,然后在 linux中注入以下环境变量
# POLARIS_INSTANCE_ZONE <b>: 设置</b> zone <b>信息, 例如</b> ap-guangzhou
# POLARIS_INSTANCE_CAMPUS <b>: 设置</b> IDC <b>信息,例如</b> ap-guangzhou-3
provider: qcloud
consumer:
serviceRouter:
# 服务路由链



chain:
# 基于主调和被调服务规则的路由策略(默认的路由策略)
- ruleBasedRouter
# 就近路由策略
- nearbyBasedRouter
#描述:服务路由插件的配置
plugin:
nearbyBasedRouter:
#描述:就近路由的最小匹配级别
#范围:region(大区)、zone(区域)、campus(园区)
matchLevel: campus
6. 部署 provider 和 consumer 微服务应用。详情请参见 Polaris-Golang Demo 部署 。
7. 进入前面提到的微服务引擎北极星网格实例页面。

- •选择服务管理 > 服务列表,查看微服务 RouteNearbyEchoServer 的实例数量。
  - 若实例数量值不为0,则表示已经成功接入微服务引擎。
  - 若实例数量为0,或者找不到 RouteNearbyEchoServer 服务名,则表示微服务应用接入微服务引擎失败。

 ▶
 EchoServerGolang
 default
 1/1
 2021-12-16 16:01:06
 2021-12-16 16:01:06

• 开启 RouteNearbyEchoServer 的就近路由。

 EchoServerGolang
 default
 1/1
 2021-12-16 16:01:06
 2021-12-16 16:01:06

- 确认 consumer 所在 CVM (容器POD)的腾讯云位置信息。
- 调用 consumer 的 HTTP 接口,执行 http 调用,其中 \${app.port} 替换为 consumer 的监听端口(默认为18080), \${add.address} 则替 换为 consumer 暴露的地址。

curl -L -X GET 'http://\${add.address}:\${app.port}/echo' 预期返回值: Hello, I'm RouteNearbyEchoServer Provider, MyLocInfo's : xxx, host : xxx:xxx

# 迁移指南 TSF Consul 迁移方案

最近更新时间: 2025-01-25 16:52:42

# 操作场景

当您在生产环境上已经使用了腾讯云的 TSF Consul 集群作为服务治理中心,并希望将其上已运行的服务迁移至腾讯云的北极星网格时,可以根据您的业务 需求,在本指引中选择适合您的迁移方案。

本文主要介绍 TSF Consul 迁移的方案和操作步骤,帮助您的微服务应用的服务治理将 TSF Consul 集群迁移到北极星网格集群。

# 前提条件

- 已创建 PolarisMesh 北极星网格,请参见 创建 PolarisMesh 治理中心。
- 本地编译构建打包机器环境已安装了Java JDK、Maven,并且能够访问 Maven 中央库。

# 迁移改造

### 应用侧改造

### 依赖更新

您需要将 TSF Consul 对应的 SDK Spring Cloud TSF 升级为 Spring Cloud Tencent(后称 SCT )。因为 Spring Cloud TSF 在依赖管理中推荐 以其自身为parent,而 SCT 的定位是治理组件,给用户的服务调用组件提供服务治理功能,但不包含服务调用实际的组件,因此需要用户额外引入服务调用 等其他组件。

以 Spring Cloud TSF 2021版本为例。业务代码项目**父依赖**更新为如下所示,该父 pom 指定了:

- spring-boot-starter-parent,开源 Spring Boot 的版本号为2.7.18。
- spring-cloud-tencent-dependencies,开源 SCT 的版本号为2.0.0.0-2021.0.9。
- spring-cloud-dependencies, 开源 Spring Cloud 的版本号为2021.0.9。



```
project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding;</pre>
```

业务代码项目**子依赖**更新可以参考下方 pom.xml ,该 pom 引入了如下依赖:

- spring-cloud-starter-tencent-all, SCT 服务治理组件总包。
- spring-boot-starter-web,web应用组件,提供 HTTP 服务给外部调用。
- spring-cloud-starter-openfeign,支持应用使用 Feign 客户端调用其他服务。如果不使用 Feign 客户端,也可以不引入。
- spring-cloud-starter-bootstrap,支持应用使用 bootstrap.xml 或者 bootstrap.yml 或者 bootstrap.yml 作为配置文件。Spring Cloud 2021 以前的版本默认引入改依赖,Spring Cloud 2021 开始不默认引入,因此需要手动引入。或改为 application.xml 或者 application.yaml 或者 application.yml。

<groupid>com.tencent.tsf</groupid>
<artifactid>tsf-demo</artifactid>
<artifactid>consumer-demo</artifactid>
<packaging>jar</packaging>



Spring Cloud Tencent <b 服务治理组件>
<proupid>com.tencent.cloud</proupid>
<artifactid>spring-cloud-starter-tencent-all</artifactid>
web<b 应用组件,提供 HTTP <b>服务给外部调用</b> >
<proupid>org.springframework.boot</proupid>
<artifactid>spring-boot-starter-web</artifactid>
<proupid>org.springframework.cloud</proupid>
<artifactid>spring-cloud-starter-openfeign</artifactid>
开启 Bootstrap 阶段
<proupid>org.springframework.cloud</proupid>
<artifactid>spring-cloud-starter-bootstrap</artifactid>

### 配置更新

SCT 使用了全新的配置,完整配置及其说明如下所示。

```
server:
    port: 48082 # 端口号

port: 48082 # 端口号

spring:
    application:
    name: QuickstartCallerService # 服务名
    config:
        import: optional:polaris # 开启北极星的配置中心
    cloud:
        polaris:
        address: grpc://(北极星的:p):8091 # 北极星的治理中心地址
        namespace: default # 服务的命名空间,默认default
        enabled: true # 开启北极星服务治理功能
        discovery:
        enabled: true # 开启北极星服务治理功能
        discovery:
        enabled: true # 开启北极星服务注册,默认true
        heartbeat:
        enabled: true # 开启北极星服务注册,默认true
        heartbeat:
        enabled: true # 开启北极星服务注册,默认true
        heartbeat:
        enabled: true # 开启北极星服务发现零实例保护,默认为false
        is-need-test-connectivity: true # 开启北极星服务发现零实例保护连通性测试,默认为false
        is-need-test-connectivity: true # 开启北极星服务发现零读量;
        weight: 100 # 指定该服务按调明的权度,默认为100
        version: 2.0.0.0 # 指定该服务按照册的故属导,默认为100
```



```
address: grpc://{北极星的ip}:8093 # 北极星的配置中心地址
           "config/caller.properties" ] # 指定北极星配置文件名列表
internal-enabled: true # 开启内部配置文件拉取,默认为tr
strategy: roundRobin # 负载均衡策略,默认为polarisWeightedRoundRobin,可选roundRobin、random、
exclude-path: /echo # 排除以下路径的服务契约扫描, 逗号分隔
name: Polaris # 服务契约名称,默认为服务名
reject-request-tips: ratelimited # 服务限流时的返回字段
enabled: true # 开启服务路由功能,默认为true
 fail-over: all # 规则路由降级配置,默认为all,可选none
```



```
address: {北极星的ip}:9091 # 指定pushgateway地址
 health-check-interval: 1000 # 健康检查间隔,默认为1000(毫秒)
local-ip-address: 127.0.0.1 # 指定注册ip
   enabled: true # 开启服务调用上报
    - polariscircuitbreaker # 开启服务熔断端点
```

### 代码改造

删除 Spring Cloud TSF 注解(可选)

以下注解为 Spring Cloud TSF 定义的功能注解,SCT 做了兼容化处理,无需引入注解也可开启对应功能,或通过配置进行关闭,引入注解不会报错。

```
@EnableTsfUnit // 新单元化单独开启
@EnableTsfAuth // 鉴权
@EnableTsfRoute // 路由
@EnableTsfRateLimit // 限流
```



@EnableISISleutn // 洞用键
@EnableTsfMonitor // 监控
@EnableTsfCircuitBreaker // 熔断
@EnableTsfFaultTolerance // <b>服务容错</b>
@EnableTsfLane // 泳道
@TsfUnitCall // <b>单元化相关</b>
@TsfUnitCustomerIdentifier // 单元化相关
@TsfUnitLocalCall // <b>单元化相关</b>
@EnableTsfZoneFilter

### 服务容错代码定义改造

- @TsfFaultTolerance 注解升级为 @FaultTolerance注解。TsfFaultToleranceProperty注解属性去除。
- TsfFaultToleranceStragety 注解属性升级为 FaultToleranceStrategy 注解属性。

### 服务监听触发回调代码改造

Spring Cloud TSF 中服务监听触发回调的接口为 ConsulServiceChangeListener, SCT 中升级为 ServiceInstanceChangeCallback。代码实 现可以参考下方示例代码:

```
@Component
@ServiceInstanceChangeListener(serviceName = "provider-demo")
public class ProviderServiceChangeCallback implements ServiceInstanceChangeCallback {
    private static final Logger LOG = LoggerFactory.getLogger(ProviderServiceChangeCallback.class);
    @Override
    public void callback(List<Instance> currentServiceInstances, List<Instance> addServiceInstances,
List<Instance> deleteServiceInstances) {
        String du = generateNodeList(currentServiceInstances);
        String delete = generateNodeList(deleteServiceInstances);
        String delete = generateNodeList(deleteServiceInstances);
        LOG.info("current: (), add: (), delete: {}", current, add, delete);
        }
        private String generateNodeList(List<Instance> deleteServiceInstances) {
        StringBuilder nodeListStr = new StringBuilder("(");
        for (Instance instance : deleteServiceInstances) {
            if (nodeListStr.length() > 1) {
                nodeListStr.append(instance.getRost()).append(":").append(instance.getPort());
        }
        nodeListStr.append(")");
        return nodeListStr.toString();
        }
    }
}
```

### API Doc 改造

Spring Cloud TSF 使用 swagger 3.0.0 实现 API Doc 的管理,SCT 使用 springdoc 2.2.0 实现。目前开源 swagger 已于2020年停止维护,相关 的很多 bugfix 和漏洞不会修复,目前持续维护的是 springdoc, springdoc官方迁移文档。无法同时使用 swagger 和 springdoc,且为了您的项目的 安全考虑,需要按照 springdoc 官方文档修改代码。 springdoc样例:

```
@RestController("/swagger")
@Tag(description = "swagger 测试", name = "swaggerValue1")
public class SwaggerApiController {
```



<pre>@RequestMapping(value = "/swagger/findMessages", produces = MediaType.APPLICATION_JSON_UTF8_VALUE)</pre>	
<pre>@Operation(method = "POST",</pre>	
summary = "根据任务ID <b>查询任务列表"</b> ,	
description = "根据任务ID <b>查询任务列表</b> Note")	
<pre>public List<messagemodel> findMessages(@RequestBody</messagemodel></pre>	
@Parameter(name = "msgIds", description = "消息ID <b>列表</b> ")    List <string> msgIds) {</string>	
List <messagemodel> messageModels = new ArrayList&lt;&gt;();</messagemodel>	
MessageModel <b>messageModel =</b> new MessageModel();	
<pre>messageModel.setMsgContent("test1");</pre>	
<pre>messageModel.setMsgId("1");</pre>	
<pre>messageModel.setSendTime(System.currentTimeMillis());</pre>	
MessageUser messageSender = new MessageUser();	
<pre>messageSender.setEmail("abc@xxxx.com");</pre>	
messageSender.setName(" <b>张三</b> ");	
messageSender.setOfficeAddress("腾讯大厦");	
<pre>messageSender.setPhoneNum("123123123");</pre>	
<pre>messageModel.setSendUser(messageSender);</pre>	
MessageUser <b>messageReceiver =</b> new MessageUser();	
<pre>messageReceiver.setEmail("abc@xxxx.com");</pre>	
<pre>messageReceiver.setName("李四");</pre>	
messageReceiver.setOfficeAddress(" <b>滨海大厦</b> ");	
<pre>messageReceiver.setPhoneNum("456456456");</pre>	
<pre>messageModel.setReceiveUsers(Lists.newArrayList(messageReceiver));</pre>	
<pre>messageModels.add(messageModel);</pre>	
return messageModels;	

## 应用管理改造

引入 Spring Cloud TSF 应用,注册到 TSF Consul 的应用详情如下图所示。

腾讯微服务平台	FREE 邀您免费试用DNSPod,实现在	E外也可访问群晖NAS								×
应用管理中心	← 应用:skye-test									
◆ 应用管理	・ 应用详情	应用部署								
制品管理	<ul> <li>应用部署</li> </ul>									
② 资源管理	• 变更记录	基本信息					部署组			
ら 部署组	<ul> <li>应用配置</li> </ul>	ID	application-y44z87py	名称	skye-test		总数	运行中	异常	
l吕 发布计划		应用类型	业务应用	开发语言	JAVA		0	0	0	
注册配置治理中心		部署方式	虚拟机部署	备注名称	- 0					
◇ Polaris(北极星)		创建时间	2024-12-04 19:25:50	标签	1					
් Nacos		42+			-		注册配置治理			编辑
S TSF Consul		窗)土	0				实例类别 <b>共享实</b>	例(TSF-Consul)		
() Zookeeper							服务注册配置治理 TSF C	onsul 🗹		
😌 Eureka		制品					接入方式 <b>SDK/#</b>	<b>፤架接</b> 入		
Consul		上传程序包	删除		支持按照程序包/版本搜索	Q C ±				
组件中心		程序名	版本号	包类型	备注 MD5	操作				
□ 微服务网关 >				新五物道						
可观测				III 7638335	-					•••
② 观测中心		共0条		20	▶ 条/页 🖂 🖣 1	/1页 ▶ ₩				C
⑤ 依赖分析 🛛 🗸										63
□。 事件中心 ✓										

如果需要迁移到北极星网格,除了需要应用更新到 Spring Cloud Tencent,还需要编辑注册配置治理,开发框架选择 Spring Cloud,实例类别选择独享 实例(北极星),然后选择可以正常连通的关联实例后,点击确认按钮。



编辑注册配置洸	台理	×
注册配置治理		
开发框架	O SpringCloud ── Dubbo ── 其他框架	
实例类别	● 独享实例(北极星)           共享实例(TSF-Consul)	
关联实例	注册中心 请选择实例 ~ C 若现有的实例不合适,您可以新建实例 ? (2)	
	配置中心 请先选择注册中心	
	治理中心 接入北极星即可使用服务治理能力,参考接入指南 C	
	<ul> <li>(3)</li> <li>→ 确定 取消</li> </ul>	

如果当前应用存在对应的部署组,则不能直接修改注册配置治理,而是需要先删除应用下的全部部署组,或者另外新建一个应用进行迁移。

<ul> <li>● 当前应用下存在部署组,暂不支持修改注册配置治理,请删除应用下全部部署组后重试。</li> <li>注册配置治理</li> <li>开发框架</li> <li>● SpringCloud</li> <li>● Dubbo</li> <li>● 其他框架</li> <li>实例类别</li> <li>● 独享实例(北极星)</li> <li>● 共享实例(TSF-Consul)</li> <li>接入方式</li> <li>BDK/框架接入</li> <li>Mesh接入</li> <li>无侵入接入</li> <li>基于 SDK/Spring Cloud 等开发框架微服务模式,提供注册注册中心、配置中心等组件</li> </ul>	编辑注册配置	治理	×
注册配置治理	() 当前应用	用下存在部署组,暂不支持修改注册配置治理,请删除应用下全部部署组后重试。	
开发框架     SpringCloud     Dubbo     其他框架       实例类别     独享实例(北极星)     共享实例(TSF-Consul)       接入方式     接入方式     SDK/框架接入     Mesh接入       无侵入接入     基于 SDK/Spring Cloud 等开发框架微服务模式,提供注册注册中心、配置中心等组件	注册配置治理		
<ul> <li>实例类别</li> <li>独享实例(北极星) ○ 共享实例(TSF-Consul)</li> <li>接入方式</li> <li>接入方式</li> <li>SDK/框架接入 Mesh接入 无侵入接入 基于 SDK/Spring Cloud 等开发框架微服务模式,提供注册注册中心、配置中心等组件</li> </ul>	开发框架	O SpringCloud ● Dubbo ● 其他框架	
接入方式 接入方式 SDK/框架接入 Mesh接入 无侵入接入 基于 SDK/Spring Cloud 等开发框架微服务模式,提供注册注册中心、配置中心等组件	实例类别	🦳 独享实例(北极星) 🔹 💿 共享实例(TSF-Consul)	
基于 SDK/Spring Cloud 等开发框架微服务模式,提供注册注册中心、配置中心等组件	接入方式	接入方式 SDK/框架接入 Mesh接入 无侵入接入	
		基于 SDK/Spring Cloud 等开发框架微服务模式,提供注册注册中心、配置中心等组件	
如一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一		确定取消	

# Nacos 迁移方案 协议兼容接入

最近更新时间:2024-02-20 09:56:31

# 使用场景

如果您希望使用北极星代替 Nacos 作为新的注册中心,北极星提供了协议兼容的方式,您仅需要更改应用中的 nacos-client 的服务端接入地址即可。无需 修改代码或者 pom 依赖。该能力目前开白使用,如需使用,可提工单,联系腾讯云助手协助。

# 前提条件

已创建 PolarisMesh 北极星网格,请参见 创建 PolarisMesh 治理中心。

# 使用说明

本文通过服务注册和配置管理两个场景介绍北极星接入的使用说明。

## 服务注册

1. Nacos 服务名和北极星服务名映射关系

Nacos 字段	Nacos 字段值	北极星字段	北极星字段值描述
namespac e	默认命名空间/非默认 命名空间 ID	namespac e	default/命名空间名称。
group	DEFAULT_GROU P	service	北极星服务名称由 nacos group 字段值和 nacos service 字段值拼接而成。
service	DEFAULT_GROU P	service	●{group/
cluster	DEFAULT	instance. metadata	作为实例标签的一部分, 实例标签 key 为 internal-nacos-cluster。

### 2. 修改服务端接入地址。

### Spring Cloud Alibaba

1. 修改应用配置文件:北极星服务端地址和服务端访问账密:

```
spring.cloud.nacos.username="可任意值,如: username"
spring.cloud.nacos.password="北极星用户/用户组的资源访问凭据 Token"
spring.cloud.nacos.discovery.server-addr="北极星服务端 IP:8848"
spring.cloud.nacos.discovery.namespace="北极星命名空间名称"
```

# 2. 北极星用户/用户组的资源访问凭据 Token 查看方式:

- 登录 TSE 控制台。
- 在**服务治理中心**下,单击侧边栏**权限控制**,选择目标北极星引擎实例。
- 选择用户或者用户组,单击查看Token。



用户用户组	权限策略			
新建导入腾讯云频	长号			请输入关键字 Q
腾讯云用户名	用户类型	用户id	创建时间	操作
Nami Tina dia 04	C_0.00	an the P	2023-11-28 12:24:45	授权 查看Token 更多 ▼
and a set	2-1-48-4	No.0000.00	2023-12-06 11:51:24	授权 查看Token 更多 ▼
Prosent N	10000(0.5)	1993 (11993	2023-12-13 13:07:36	授权 查看Token 更多▼
出日本			20 - 条/页	H 4 1 /1页 ►

- 登录 TSE 控制台。
- 在**服务治理中心**下,单击侧边栏**实例列表**,单击目标北极星引擎实例。
- 在引擎详情页的**实例信息**页面,可以查看到引擎的访问地址。

客户端接入端口	注册发现	8091 (service-grpc)	8092 (service-trpc)			
	配置管理	8093 (config-grpc)				
	监控上报	9091 (pushgateway)				
OpenAPI 端口	8090 (http)					
协议兼容端口	7779 (IS)	8761 (eureka) 15	5010 (xdsv3) 15020 (xdsv2)			
内网负载均衡 🚯	私有网络		子网	内网地址	访问控制	操作
内网负载均衡 ①	私有网络 New York I They	er var B	子用 1000 (1000 1000 1100 1100 1100 1100 110	内网地址 10.0.20.246	访问控制	操作
内网负载均衡 🛈	私有网络 ************************************	er nam 12	子用 Engle making englishing 2	內网地址 10.0.20.246	访问控制	操作 删除
内网负载均衡 ①	私有网络 添加内网负载均衡 公网IP			内网地址 10.0.20.246 访问策略	访问控制	操作 <b>删除</b> 操作

### 原生 Nacos-Client

1. 原生Nacos Client修改:北极星服务端地址和服务端访问账密:

<pre>Properties properties = new Properties(); properties.put(PropertyKeyConst.SERVER_ADDR, "北极星服务端IP:8848"); properties.put(PropertyKeyConst.NAMESPACE, "北极星命名空间名称"); properties.put(PropertyKeyConst.USERNAME, "可任意值"); properties.put(PropertyKeyConst.PASSWORD, "北极星用户/用户组的资源访问凭据 Token");</pre>	
// <b>创建注册发现客户端</b> NamingService <b>namingService =</b> NacosFactory.createNamingService(properties);	

- 2. 北极星用户/用户组的资源访问凭据 Token 查看方式:
  - 登录 TSE 控制台。
  - 在 **服务治理中心**下,点击侧边栏 权限控制,选择目标北极星引擎实例。
  - 选择 用户 或者 用户组,点击 查看Token。



0.000			/ BUPPOEL, KAN	
用户用户组权	限策略			
新雄 导入腾讯云账号	8			请输入关键字 Q
腾讯云用户名	用户类型	用户id	创建时间	操作
Navel 111 and the Office	C_0.00	and the P	2023-11-28 12:24:45	授权 查看Token 更多 ▼
na an a	2-7-4844	No.0999.000	2023-12-06 11:51:24	授权 查看Token 更多 ▼
er son en s	10000754	2010/1201	2023-12-13 13:07:36	授权 查看Token 更多▼
共3条			20 - 条/页	स <b>∢</b> 1 /1页 ▶

- 登录 TSE 控制台。
- 在 **服务治理中心**下,点击侧边栏 **实例列表**,点击目标北极星引擎实例。
- 在引擎详情页的**实例信息**页面,可以查看到引擎的访问地址。

	问地址				
客户端接入端口	注册发现 8091 (:	service-grpc) 8092 (service-trpc)			
	配置管理 8093(	config-grpc)			
	监控上报 9091 (	pushgateway)			
OpenAPI 端口	8090 (http)				
协议兼容端口	7779 (I5) 8761	eureka) 15010 (xdsv3) 15020 (xdsv2)			
國负载均衡 🚯	私有网络	子网	内网地址	访问控制	操作
前负载均衡 🚺	私有网络 Sponse appoint of a	. К. Поболисти и и 15 3-14 3-14	内网地址 10.0.20.246	访问控制	操作
月负载均衡 ①	私有网络 1400-111-11-11-11-11-11-11-11-11-11-11-11-	78 - Fig. (1999) (19977) (19977) (1997) (19977) (1997) (1997) (1997) (1997) (19	內阿地址 10.0.20.246	访问控制	操作
间负载均衡 ①	私有网络 1990-111-11-11-11-12 添加内网负载均衡 公网IP	子利 子利 公用等意	内网地址 10.0.20.246 访问策略	访问控制	操作 <b>删除</b> 操作

### Dubbo

1. 修改应用配置文件:北极星服务端地址和服务端访问账密:

dubbo	
registry	
address: nacos:// <b>北极星服务端</b> IP:8848?username= <b>可任意值</b> ≬password= <b>北极星用户/用户组的资源访问凭据</b> Tok	ken
parameters.namespace: 北极星命名空间名称	
metadata-report	
address: nacos:// <b>北极星服务端</b> IP:8848	

- 2. 北极星用户/用户组的资源访问凭据 Token 查看方式:
  - 登录 TSE 控制台。
  - 在**服务治理中心**下,单击侧边栏**权限控制**,选择目标北极星引擎实例。
  - 选择**用户**或者**用户组**,单击**查看Token**。



用户 用户组 权同	<b>根策略</b>			
新建导入腾讯云账号				训输入关键字
腾讯云用户名	用户类型	用户id	创建时间	操作
Navel 111 and the OPE	Con	and the M	2023-11-28 12:24:45	授权 查看Token 更多▼
na na san	2-7-4-6	NG 8999 (1990)	2023-12-06 11:51:24	授权 查看Token 更多 ▼
er nossenno	1041 (105)	1993.011993	2023-12-13 13:07:36	授权 查看Token 更多 ▼

- 登录 TSE 控制台。
- 在**服务治理中心**下,单击侧边栏**实例列表**,单击目标北极星引擎实例。
- 在引擎详情页的**实例信息**页面,可以查看到引擎的访问地址。

	Johen				
客户端接入端口	注册发现 8091	(service-grpc) 8092 (service-trpc)			
	配置管理 8093	(config-grpc)			
	监控上报 9091	(pushgateway)			
OpenAPI 端口	8090 (http)				
协议兼容端口	7779 (15) 8761	(eureka) 15010 (xdsv3) 15020 (xdsv2)			
的网负载均衡 🚯	私有网络	子网	内网地址	访问控制	操作
9网负载均衡 🛈	私有网络 <b>1</b> 10-11-11-11-11-11-11-11-11-11-11-11-11-1	za Marina analazio esta analia a	内网地址 10.0.20.246	访问控制	操作 删除
的网负载均衡 🛈	私有网络 ************************************	子月 王 M M M M M M M M M M M M M M M M M M M	内阔地址 10.0.20.246	访问控制	操作 删除
的网负载均衡 ③	私有网络	公開単変 子用 子用	内隔地址 10.0.20.246 访问策略	访问控制	操作 删除 操作

## 配置管理

#### 1. Nacos 配置信息和北极星配置信息映射关系

Nacos 字段	Nacos 字段值	北极星字段	北极星字段值描述
namespace	默认命名空间/非默认命名空间 ID	namespace	default/命名空间名称
group	DEFAULT_GROUP	group	北极星配置分组名称
datald	application.yaml	file_name	北极星配置文件名称

## 2. 修改服务端接入地址

### Spring Cloud Alibaba

1. 修改应用配置文件:北极星服务端地址和服务端访问账密:

```
spring.cloud.nacos.username="可任意值"
spring.cloud.nacos.password="北极星用户/用户组的资源访问凭据 Token"
spring.cloud.nacos.config.namespace="北极星命名空间名称"
spring.cloud.nacos.config.server-addr="北极星服务端IP:8848"
spring.cloud.nacos.config.group="北极星配置分组名称"
```

### 2. 北极星用户/用户组的资源访问凭据 Token 查看方式:

○ 登录 TSE 控制台。



○ 在**服务治理中心**下,单击侧边栏**权限控制**,选择目标北极星引擎实例。 ○ 选择用户或者用户组,单击查看Token。 . **权限控制** ⑤ 中国大陆 ▼ ■■ 产品体验, 您说了算 加入用户交流群 ① 操作指南 记 用户 用户组 权限策略 请输入关键字 Q 🗘 新建 导入腾讯云账号 用户id 脑讯云用户名 用户类型 创建时间 18-01 Nave Marka (1994) La Capital 授权 查看Token 更多 ▼  $\mathrm{Im} \mathcal{C}^{**} \subset \mathrm{Im} \mathcal{B}^*$ 2023-11-28 12:24:45 201-4810 10000-000  $\sim 1000000, 100$ 2023-12-06 11:51:24 授权 查看Token 更多 ▼ 1041(78-9) 1110044100 1203111203 2023-12-13 13:07:36 授权 查看Token 更多 ▼ 20▼条/页 H < 1 /1页 → H 共3条

#### 3. 北极星服务端 IP 地址查看方式:

- 登录 TSE 控制台。
- 在**服务治理中心**下,单击侧边栏**实例列表**,单击目标北极星引擎实例。
- 在引擎详情页的**实例信息**页面,可以查看到引擎的访问地址。

Polaris Serveri	方问地址				
客户端接入端口	注册发现 8091 (servic	e-grpc) 8092 (service-trpc)			
	配置管理 8093 (config	-grpc)			
	监控上报 9091 (pushg	ateway)			
OpenAPI 端口	8090 (http)				
协议兼容端口	7779 (I5) 8761 (eure)	a) 15010 (xdsv3) 15020 (xdsv2)			
内网负载均衡 🚯	私有网络	子网	内网地址	访问控制	操作
	de constates e a R	$\mathbf{n}$ is the state of the state of the state $\mathbf{n}$	10.0.20.246	1	删除
	添加内网负载均衡				
公网负载均衡 🛈	公网IP	公网带宽	访问策略		操作

### 原生 Nacos-Client

1. 原生Nacos Client修改:北极星服务端地址和服务端访问账密:

<pre>Properties properties = new Properties(); properties.put(PropertyKeyConst.SERVER_ADDR, "北极星服务端IP:8848"); properties.put(PropertyKeyConst.NAMESPACE, "北极星命名空间名称"); properties.put(PropertyKeyConst.USERNAME, "可任意值"); properties.put(PropertyKeyConst.PASSWORD, "北极星用户/用户组的资源访问凭据 Token");</pre>
// <b>注册配置客户端</b> ConfigService <b>configService</b> = new NacosConfigService( <b>properties</b> );

- 2. 北极星用户/用户组的资源访问凭据 Token 查看方式:
  - 登录 TSE 控制台。
  - 在**服务治理中心**下,单击侧边栏**权限控制**,选择目标北极星引擎实例。
  - 选择用户或者用户组,单击查看Token。



G TEXA			1 00144-522, 122.01	
用户 用户组	权限策略			
新建 导入腾讯云	账号			请输入关键字 Q
腾讯云用户名	用户类型	用户id	创建时间	操作
National Contraction (CP)	i_s.p	any na an	2023-11-28 12:24:45	授权 查看Token 更多 ▼
and the second	201-000	No.0000.00	2023-12-06 11:51:24	授权 查看Token 更多 ▼
er son en s	1980 (188)	2010/12/08	2023-12-13 13:07:36	授权 查看Token 更多 ▼
共3条			20 - 条/页	

- 登录 TSE 控制台。
- 在**服务治理中心**下,单击侧边栏**实例列表**,单击目标北极星引擎实例。
- 在引擎详情页的**实例信息**页面,可以查看到引擎的访问地址。

	问地址				
客户端接入谐口	注册发现 8091 (service-gr	pc) 8092 (service-trpc)			
	配置管理 8093 (config-gr	oc)			
	监控上报 9091 (pushgatev	vay)			
OpenAPI 端口	8090 (http)				
协议兼容端口	7779 (I5) 8761 (eureka)	15010 (xdsv3) 15020 (xdsv2)			
网负载均衡 🚯					
	私有网络	子网	内网地址	访问控制	操作
			10.0.20.240		mino
	de ser de la R	in observations in the second of 2	10.0.20.246	/	DES PAR
	添加内网负载均衡	noberanisered and 5	10.0.20.248		DE3 MAX
网负载均衡 🚯	添加内网负载均衡 公网IP	公网带宽	访问策略	,	操作

### Dubbo

1. 修改应用配置文件:北极星服务端地址和服务端访问账密:

```
ubbo
config-center
address: nacos://北极星服务端IP:8848
```

- 2. 北极星用户/用户组的资源访问凭据 Token 查看方式:
  - 登录 TSE 控制台。
  - 在**服务治理中心**下,单击侧边栏**权限控制**,选择目标北极星引擎实例。
  - 选择用户或者用户组,单击查看Token。

<b>限控制</b> 🔇 中国大陆 🔻		¥	产品体验,您说了	1 加入用户交流群 💬 操作指南
用户用户组,	权限策略			
新建导入腾讯云频	K <del>S</del>			请输入关键字 🛛 🗘
腾讯云用户名	用户类型	用户id	创建时间	操作
Salah Malanda Off	0.00	as to P	2023-11-28 12:24:45	授权 <b>查看Token 更多 ▼</b>
n an	2-7 - +8-14	2.00000.00	2023-12-06 11:51:24	授权 查看Token 更多 ▼
Press Pro-	$\max\{i \in Q, i\}$	1983.01988	2023-12-13 13:07:36	授权 查看Token 更多 ▼
共3条			20 - 条/页	H < 1 /1页 ► H

3. 北极星服务端 IP 地址查看方式:



- 登录 TSE 控制台。
- 在**服务治理中心**下,单击侧边栏**实例列表**,单击目标北极星引擎实例。
- 在引擎详情页的**实例信息**页面,可以查看到引擎的访问地址。

and the second second second second	14 M (1 M )				
答尸端按入端口	注册发现 8091 (se	rvice-grpc) 8092 (service-trpc)			
	配置管理 8093 (c)	onfig-grpc)			
	监控上报 9091 (pt	ishgateway)			
OpenAPI 端口	8090 (http)				
协议兼容端口	7779 (I5) 8761 (e	ureka) 15010 (xdsv3) 15020 (xdsv2)			
内网负载均衡 🚯					
-	私有网络	子网	内网地址	访问控制	操作
				<i>.</i>	B(04-
	de constate a su	R molecular and service and	10.0.20.246		Dis Max
	1940-1073-1949-0-1-19 添加内网负载均衡	<ul> <li>Reconcision and the second seco</li></ul>	10.0.20.246		003 MPR
公网负载均衡 🕄	<b>和かった日本和</b> の人数均衡 公网IP	E E E E E E E E E E E E E E E E E E E	10.0.20.246	问策略	操作

# Eureka 迁移方案 Eureka协议使用命名空间

最近更新时间: 2023-04-20 15:59:57

# 介绍

北极星对 Eureka 的 API 进行了全兼容,由于 Eureka 默认不支持命名空间,北极星拓展了一个请求头 x-namespace 来支持命名空间的隔离。当有客户 端的请求到达服务端,服务端将检查这个 header:

- 如果 header 存在,则使用这个 header 标识的命名空间进行服务注册发现的隔离。
- 如果 header 不存在或者值为空,则使用默认的命名空间 default 。

# 示例

下面通过 eureka 协议的api进行说明。向命名空间 ns1 下注册服务 EUREKA-DEMO-PROVIDER 的一个实例,实例端口为18081。






```
curi -x 'GET' 'http://127.0.0.1:8761/eureka/apps/EUREKA-DEMO-PROVIDER' \

-H "Accept: application/json" \

-H "x-namespace: ns1" \

-v

## 返回18081端口实例

查询 ns2 下 EUREKA-DEMO-PROVIDER 的实例。
```



如上所见,向不同命名空间下相同的服务名进行服务注册发现,已经通过命名空间进行隔离。

# API 支持列表

## 注册服务实例

- 请求方法: PUT
- 路由: /eureka/apps/{appId}
- 示例:



"secureVipAddress": "EUREKA-DEMO-PROVIDER",

# 健康检查

- 请求方法: PUT
- 路由: /eureka/apps/{appId}/{instanceId}
- 示例:

```
curl -X 'PUT' 'http://127.0.0.1:8761/eureka/apps/EUREKA-DEMO-PROVIDER/10.91.80.100:eureka-demo-
provider:18081' \
-H "x-namespace: ns1" \
-V
```

#### 取消注册

- 请求方法:DELETE
- 路由: /eureka/apps/{appId}/{instanceId}
- 示例:

```
curl -X 'DELETE' 'http://127.0.0.1:8761/eureka/apps/EUREKA-DEMO-PROVIDER/10.91.80.100:eureka-demo-
provider:18081' \
-H "x-namespace: ns1" \
-v
```

# 查询所有实例

- 请求方法: GET
- 路由: /eureka/apps
- 示例:

```
curl -X 'GET' 'http://127.0.0.1:8761/eureka/apps' \
-H "Accept: application/json" \
-H "x-namespace: ns1" \
-v
```

#### 查询 delta 实例

- 请求方法: GET
- 路由: /eureka/apps/delta
- 示例:

rl -X 'GET' 'http://127.0.0.1:8761/eureka/apps/delta' \



-H "Accept: application/jso

## 查询某个服务下所有实例

- 请求方法: GET
- 路由: /eureka/apps/{appId}
- 示例:

```
curl -X 'GET' 'http://127.0.0.1:8761/eureka/apps/EUREKA-DEMO-PROVIDER' \
-H "Accept: application/json" \
-H "x-namespace: ns1" \
-v
```

## 查询某个服务下某个实例

- 请求方法: GET
- 路由: /eureka/apps/{appId}/{instanceId}
- 示例:

```
curl -X 'GET' 'http://127.0.0.1:8761/eureka/apps/EUREKA-DEMO-PROVIDER/10.91.80.100:eureka-demo-
provider:18081' \
-H "Accept: application/json" \
-H "x-namespace: ns1" \
-v
```

#### 查询某个实例

- 请求方法: GET
- 路由: /eureka/instance/{instanceld}
- 示例:

```
curl -X 'GET' 'http://127.0.0.1:8761/eureka/instance/10.91.80.100:eureka-demo-provider:18081' \
-H "Accept: application/json" \
-H "x-namespace: ns1" \
-v
```

# 查询 vip 下的所有实例

- 请求方法: GET
- 路由: /eureka/vips/{vip}
- 示例:

```
curl -X 'GET' 'http://127.0.0.1:8761/eureka/vips/EUREKA-DEMO-PROVIDER' \
-H "Accept: application/json" \
-H "x-namespace: ns1" \
-v
```

#### 查询 secure vip 下的所有实例

- 请求方法:GET
- 路由: /eureka/svips/{svip}
- 示例



```
curl -X 'GET' 'http://127.0.0.1:8761/eureka/svips/EUREKA-DEMO-PROVIDER' \
-H "Accept: application/json" \
-H "x-namespace: ns1" \
-v
```

# 强制隔离实例

- 请求方法: PUT
- 路由: /eureka/apps/{appId}/{instanceId}/status
- 示例

```
curl -X 'PUT' 'http://127.0.0.1:8761/eureka/apps/EUREKA-DEMO-PROVIDER/10.91.80.100:eureka-demo-
provider:18081/status' \
    -H "x-namespace: ns1" \
    -v
```

## 关闭强制隔离

- 请求方法:DELETE
- 路由: /eureka/apps/{appId}/{instanceId}/status
- 示例

```
curl -X 'DELETE' 'http://127.0.0.1:8761/eureka/apps/EUREKA-DEMO-PROVIDER/10.91.80.100:eureka-demo-
provider:18081/status' \
-H "x-namespace: ns1" \
...
```

#### 更新元数据

- 请求方法: PUT
- 路由: /eureka/apps/{appId}/{instanceId}/metadata?{key}={value}
- 示例

```
curl -X 'PUT' 'http://127.0.0.1:8761/eureka/apps/EUREKA-DEMO-PROVIDER/10.91.80.100:eureka-demo-
provider:18081/metadata?k=v' \
-H "x-namespace: ns1" \
-v
```

#### 批量复制

- 请求方法: POST
- 路由: /eureka/peerreplication/batch
- 示例





```
}' \
-v
```

# 从 Eureka 迁移到 Polaris

最近更新时间: 2024-01-11 15:30:01

# 操作场景

本文通过一个 demo 进行应用使用 spring-cloud-eureka-client 接入微服务引擎托管的 PolarisMesh 治理中心的全流程操作演示,帮助您快速了 解如何使用北极星网格。

#### 前提条件

- 已创建 PolarisMesh 北极星网格,请参见 创建 PolarisMesh 治理中心。
- 下载 Github 的 demo 源码 到本地并解压。
- 本地编译构建打包机器环境已安装了Java JDK、Maven,并且能够访问 Maven 中央库。
- 根据您自身的业务,已准备好业务部署的资源, 虚拟机部署 、 容器化部署 和 TEM 部署 选择其中一种方式即可。
  - 虚拟机部署已创建 CVM 虚拟机,请参见 创建 CVM 虚拟机。
  - 容器化部署已创建 TKE 容器集群,请参见 创建 TKE 集群。
  - TEM部署已创建 TEM 环境,请参见 创建 TEM 环境。

#### 操作步骤

- 1. 登录 TSE 控制台。
- 2. 在北极星网格下的 polarismesh 页面,单击页面左上方下拉列表,选择目标地域。
- 3. 单击目标引擎的"ID",进入基本信息页面。
- 4. 查看访问地址, eureka-client 应用访问使用 eureka 端口(8761):

访问地址			
端口	8090/http、8091/grpc、8092/trpc、7779/l5、1	5020/xds(v2)、8761/eureka	
内网地址	私有网络	子网	内网地址
	pk-gz-vpc(vpc-)	polaris-dev(subnet) 🖬	1 .6

- 5. 修改 demo 中的注册中心地址。
- 在下载到本地的 demo 源码目录 下,分别找到:

 $\texttt{eureka/eureka-java/consumer/src/main/resources/application.yml} \ \pmb{n}$ 

eureka/eureka-java/provider/src/main/resources/application.yml 两个文件。

• 添加微服务引擎北极星网格地址到项目配置文件中(以 eureka/eureka-java/consumer/src/main/resources/application.yml 为例)。

eureka:				
client:				
serviceUrl:				
defaultZo.	ne: http://10.0.4.6	:8761/eureka/		

6. 打包 demo 源码成 jar 包。

6.1 在 eureka-java 源码根目录下,打开 cmd 命令,执行 mvn clean package 命令,对项目进行打包编译。

6.2 编译成功后,生成如下表所示的2个 Jar 包。



软件包所在目录	软件包名称	说明
\eureka-java\provider\target	eureka-provider-\${version}-SNAPSHOT.jar	服务生产者
\eureka-java\consumer\target	eureka-consumer-\${version}-SNAPSHOT.jar	服务消费者

7. 部署 provider 和 consumer 微服务应用,虚拟机部署方式、容器化部署方式以及TEM部署方式根据您业务实际的部署方式选择一种即可。

- 7.1 虚拟机部署方式部署 provider 和 consumer 微服务应用。
- 上传 Jar 包至 CVM 实例。
- 执行启动命令进行启动:

nohup java -jar [jar**包名称**] &

- 7.2 容器化部署方式部署 provider 和 consumer 微服务应用。
- 编写 dockerfile 生成镜像,参考:

```
FROM java:8
ADD ./eureka-provider-${VERSION}.jar /root/app.jar
ENTRYPOINT ["java","-jar","/root/app.jar"]
```

○ 通过 TKE 部署并运行镜像。

7.3 TEM 部署方式部署 provider 和 consumer 微服务应用。

○ 选择 TEM 环境,注意所选择的环境,其依赖的 VPC,必须和上面已经创建的北极星网格实例所依赖的 VPC 一致:

基本信息	资源管理 访问管理 配置管理
基本信息	
环境名称	andr-test-1 🎤
ID	env-
所在地域	广州
集群网络	VPC( 2)闩网( 2)
创建时间	2021-12-21 20:08:21
描述	1



○ 在已选择的环境中,新建 TEM 应用,相关参数填写参考:

<b>石称 *</b>	polaris-eureka-provider
	最长为40个字符,只能包含小写字母、数字及分隔符(*->),且不能以分隔符开头或结尾
描述 (选填)	请输入描述
开发语言 程序包上传方式	<ul> <li>● JAVA 其他语言</li> <li>〕 遺像</li> <li>● JAR包</li> <li>● WAR包</li> </ul>
启用组件	将为您上传的程序包自动制作镜像,并存储至镜像仓库

○ 部署应用,相关参数填写请参考(端口号映射,consumer 默认端口号为20002, provider 默认端口号为20001):

应用名	polaris-eureka-provider
发布环境	andr-test-1
	如无合适环境,可前往创建区
JDK版本	KonaJDK 8 👻
上传程序包	上传 Demo下载 ▼
版本号	建议填写有意义的版本号 使用时间戳作为版本号
	最长32个字符,支持英文字母a-z、A-Z、横杠"-"、下划线"_"、点"."
版本描述	请输入版本描述
启动参数	-Xms128m -Xmx512m -XX:MetaspaceSize=128m - XX:MaxMetaspaceSize=512m
▼ 资源配置	
▲ 访问配置	
访问方式	● 环境内访问 ● VPC内网访问(四层转发) ● 公网访问(四层转发) 提供一个可以被环境内其他应用访问的入口,支持TCP/UDP协议。 如您需要配置公网的HTTP/HTTPS协议转发规则,可前往应用路由页面 ■ 转发规则配置。
端口映射	协议 ()     容器端口 ()     应用监听端口 ()
	TCP 👻 20001 20001
	添加端口映射



○ 查看访问路径,consumer 应用部署完后,可以在基本信息 > 访问配置中查看访问地址,如需公网访问,可以编辑并更新开启公网访问:

基本信息 日志 监控 实例列表 发布记录	
基本信息	访问配置 编辑并更新
应用名称 polaris-eureka-consumer	访问方式 公网访问(四层转发)
应用描述 -	访问地址 公网IF
私有网络 vpc-pva1tmnj 🖸	转发配置 前往设置 🗹
子网 subnet-5118438c 🗹 (可用P数202个)	
运行/目标实际数 22	配置信息
ohz rwie	启动参数 -Xms128m -Xms512m -XX:MetaspaceSize=128m -XX:MaxMetaspaceSize=512m
	环境变量 暂未设置
弹性伸缩	持久化存储 暂未设置
规格 1核2G	日志信息 日志主题: 暫未设置   采集路径: 暫未设置
初始化实例数 2个 编辑并更新	配置信息 <b>智未设置</b>
弹性伸缩 未 <b>启用 编辑并更新</b>	

- 8. 确认部署结果。
  - 8.1 进入前面提到的微北极星网格实例页面。
  - 选择**服务管理 > 服务列表**, 查看微服务 EUREKA-CONSUMER-SERVICE 和 EUREKA-PROVIDER-SERVICE 的实例数量:
  - 若实例数量值不为0,则表示已经成功接入微服务引擎。
  - 若实例数量为0,或者找不到具体服务的服务名,则表示微服务应用接入微服务引擎失败。

服务列表							
新建制除						请选择条件进行过	× Q ¢ φ
服务名	命名空间 ▼	音阶门	业务	健康实例/总实例	创建时间	修改时间	操作
	default	-	-	2/2	2022-01-10 23:04:02	2022-01-10 23:04:02	編輯 删除
	default			2/2	2022-01-10 23:01:19	2022-01-10 23:01:19	编辑 删除

8.2 调用 consumer 的 HTTP 接口。执行 http 调用,其中 \${app.port} 替换为 consumer 的监听端口(默认为20002), \${add.address} 则替换为 consumer 暴露的地址。

curl -L -X GET 'http://\${add.address}:\${app.port}/echo?value=hello\_world'' **预期返回值:** echo: hello\_world