



应用性能监控 APM

用户操作指南

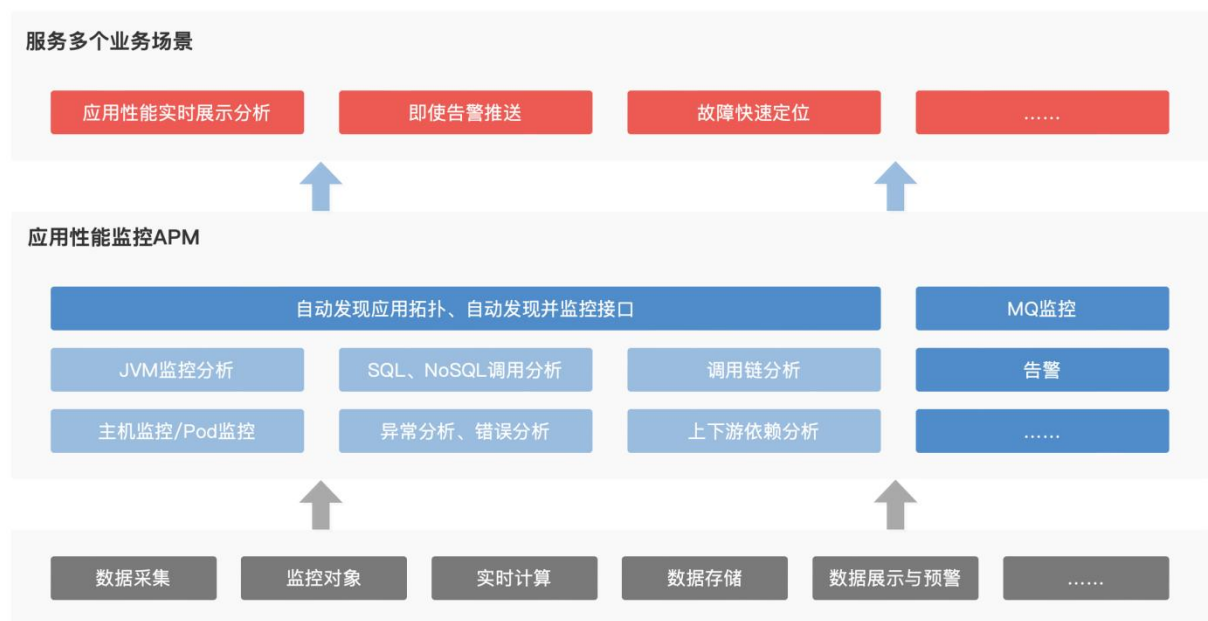
天翼云科技有限公司

一、产品概述

1.1、产品定义

应用性能监控（Application Performance Management, APM）是天翼云可观测体系中的一款子产品，帮助您进行应用性能管理。

通过无侵入式探针接入的方式，无需您修改代码，就能够对应用进行全方位监控，帮助您快速定位出错接口和慢接口，重现调用参数，发现系统瓶颈，从而大幅提升线上问题诊断的效率。



（1）数据采集计算存储

通过应用探针采集监控对象的数据，实时计算并存储。

（2）能力及场景：性能指标分析展示

- SQL、NoSQL 调用分析;异常分析、错误分析:捕获应用实例接口的慢 SQL、NoSQL 调用数据,展示分析报表或者异常分类报表,对“错”、“慢”等常见问题进行更细致的分析。
- 调用链分析;上下游依赖分析:通过应用探针能够自动发现应用的上下游依赖关系。捕获计算并立体展示不同应用之间组成的调用链。
- 更多监控分析能力:提供 JVM 监控、Pod 监控、主机监控、SQL 调用分析等能力,全面观测应用运行情况。

(3) 能力及场景:下游告警推送

在一定周期内监控某些特定指标,并根据给定的阈值,每隔若干个时间段通过多种方式(短信、邮件等)发送告警通知。

1.2、产品优势

应用性能监控 APM 的核心产品优势如下:

(1) 即开即用

支持无侵入式探针,无需修改应用逻辑代码,即可开始监控。

(2) 一站式监控

一站式提供应用性能黄金指标、全链路调用追踪能力。

(3) 兼容开放

深度融合 OpenTracing 标准，拥抱开源生态。

(4) 低成本

全托管式监控服务，免运维，按真实用量付费。

1.3、功能特性

应用性能监控 APM 的主要功能特性如下：

分类	功能	说明
应用接入	为 Java 应用手动安装 Agent	为 Java 应用安装 Agent 后，APM 即可开始监控 Java 应用，您可以查看应用拓扑、调用链路、异常事务、慢事务和 SQL 分析等一系列监控数据。
	为容器服务 Kubernetes 版 Java 应用安装探针	借助 APM 应用性能监控，您可以对容器环境的应用进行应用拓扑、接口调用、异常事务和慢事务监控、SQL 分析等监控。本文将帮助您将容器环境中的应用接入 APM 应用性能监控。
应用	应用总览	以应用为维度，统计整个应用的关键指标，帮助您快速掌握应用的整体状况。

详情	概览	应用的概览可以提供核心监控数据,帮助您快速掌握某个应用的具体情况。
	JVM 监控	JVM 监控即 Java 虚拟机监控,用于监控重要的 JVM 指标。
	主机监控	主机监控是指对单个主机(服务器)的资源使用情况进行监控和分析,包括对主机的 CPU、内存、磁盘、网络等资源使用情况的监控和分析。
	Pod 监控	Pod 监控包含对 CPU、物理内存、流量的监控,是保障 K8s 应用程序稳定性和可靠性的重要任务之一。通过定期监控和分析 Pod 的资源使用情况,您可以及时发现和解决问题,优化应用程序的运行效率和性能。
	SQL 调用分析	SQL 调用分析通常是指对一段时间内对 SQL 语句的调用次数、耗时的统计分析。
	NoSQL 调用分析	NoSQL 调用分析是指对不同操作命令的调用次数、耗时的统计分析。
	异常分析	异常分析可以帮助您方便了解应用的异常情况。
	错误分析	错误分析可以帮助您更快了解应用的错误信息。
	上游应用	上游应用是当前应用/SQL 的调用者。通过上游应用分析可

		以查看上游应用/SQL 的请求量、平均延时、错误数信息。
下游应用		下游应用指当前应用的被调用方。通过下游应用分析可以查看下游应用的请求量、平均延时、错误数信息。
调用链查询		展示该应用实例/SQL/NoSQL 涉及的调用链信息，包括 TraceID、产生时间、接口名称、耗时、状态信息。
接口调用		提供应用接口级别的监控详情。通过丰富的监控分析报表，展示包括 SQL、NoSQL 调用分析；异常、错误分析；链路上下游及调用链分析。
SQL 调用		以关系型数据库为维度，展示当前应用下，各个 SQL 的监控详情。
NoSQL 调用		以非关系型数据库为维度，展示当前应用下，各个 NoSQL 的监控详情。
外部调用		指当前应用向外部应用发起的调用，以外部应用的 IP 端口为维度进行展示，可用于定位应用外部调用缓慢或出错的问题。
MQ 监控		MQ 监控是指对消息队列系统进行监控和分析，以保障消息队列的稳定性和性能。

	自定义配置	自定义配置可以让您对具体某个应用进行单独管理，包括 agent 开关、阈值、调用链采样率等等。
其他	agent 管理	Agent 列表用于管理您安装的所有探针，包括“正常上报”和“未上报”的探针，显示各个探针的关键指标信息，提供查看应用详情的快捷入口和升级探针的功能入口。
	调用链查询	展示当前租户下所有调用链路信息。您可以根据多个筛选条件租户查询您想看的调用链，可以点击「TraceID」查看具体的调用链详情。
	用量统计	展示当前租户下的探针用量情况，包括实例数、探针时、Span 存储量。
	告警	展示当前租户下所有告警信息，支持配置告警规则，查看告警事件和告警发送历史。

1.4、应用场景

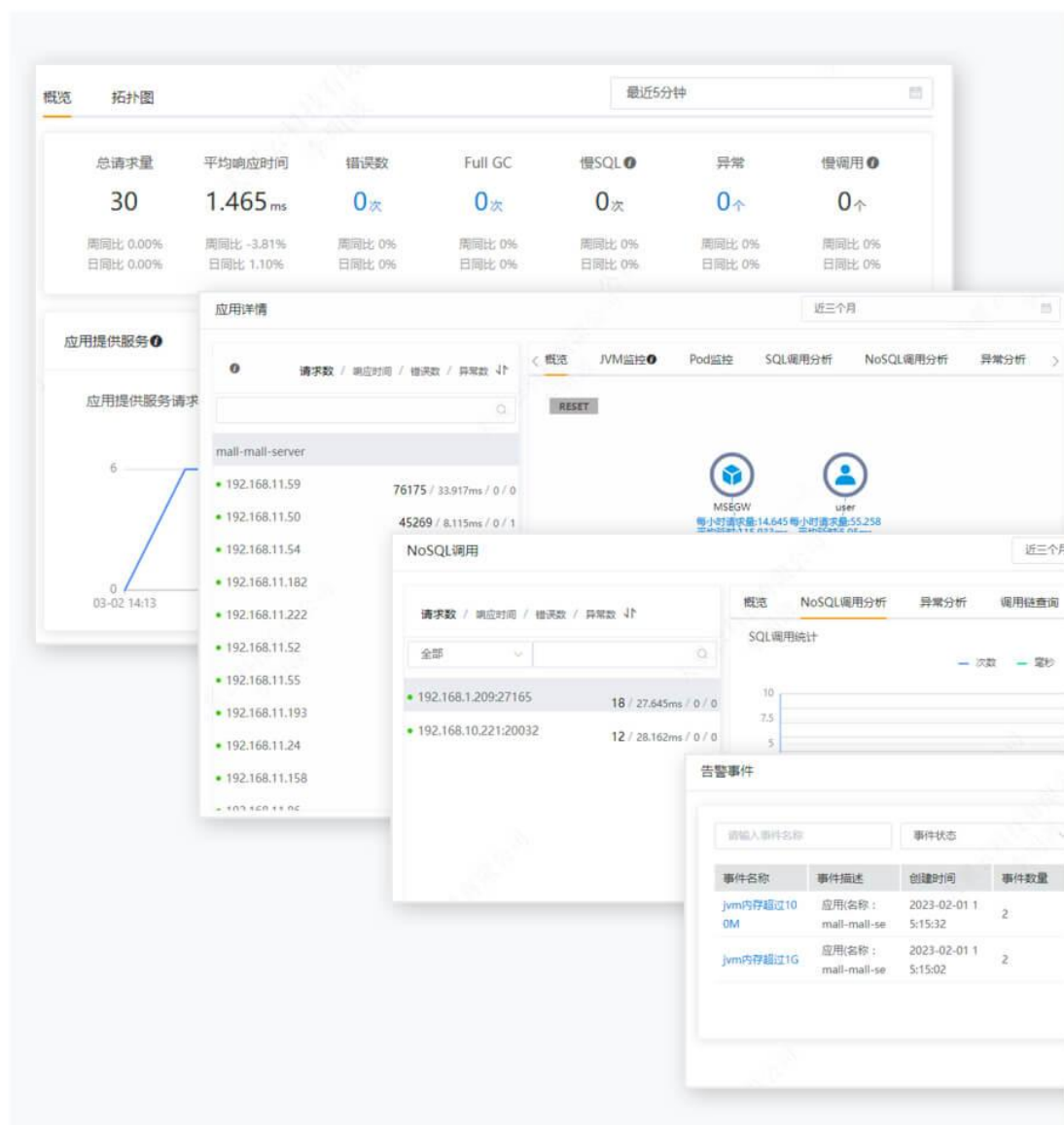
应用场景如下：

微服务架构应用性能监控

(1) 场景说明

使用分布式微服务架构的应用，虽然研发交付效率高，但处理架构梳理、性能分析、异常定位等问题的难度却陡增。应用性能监控提供了大型分布式应用的异常诊断能力，当应用出现请求失败或性能下降时，通过应用拓扑、调用链、性能指标监控等能力组合，可以帮助用户快速定位问题。

(2) 产品优势



● 异常定位

当出现异常调用，通过调用链能力查看完整调用链路详情，可将问题快速定位到代码级

- 慢 SQL 分析

通过自定义的慢查询阈值、结合 SQL 的调用频次，获取导致数据库性能下降的不规范的 SQL 语句

- 告警

针对接口响应时间、异常调用、数据库、JVM 等性能指标做一定阈值的告警，先于客户之前发现并解决问题

1.5、关键指标统计说明

1.5.1、实例

实例是指被监控的应用所部署的机器。例如在下图中，service 是一个应用，下方的每一行都是该应用所部署的一台机器，即一个实例。



1.5.2、相关统计页面

(1) 应用总览页面

在已发布应用页面单击应用总览，即可进入应用总览页面，可以看到基本信息、部署规格、访问方式配置。在页面顶部选择概览或拓扑图菜单，可以查看相应的统计信息。

概览分析页签

- 应用提供的服务：请求量和平均响应时间。
- 应用依赖的服务：请求量、平均响应时间、实例数和 HTTP-状态码。
- 慢调用：慢接口调用分析。
- 统计分析：平均响应时间、异常类型和出现次数。

拓扑图页签

应用拓扑图：应用每分钟的请求数、响应时间和错误数。

- 实例健康： 绿色表示正常，黄色表示警告，红色表示严重。
- 调用类型：

调用类型	描述	备注
HTTP 入口	客户端使用 HTTP 协议调用该应用的入口	服务入口调用
调用 HTTP	该调用为该应用对其他服务发起的 HTTP 调用	服务间调用
调用 MySQL	对 MySQL 进行操作的调用	数据库调用
调用 Redis	对 Redis 进行操作的调用	数据库调用

(2) 应用详情页面

在已发布应用页面点击监控->实例监控，即可进入应用详情页面，此页面展示当前应用的调用详细信息,可看到各实例的请求数、错误数、响应时间、异常数、HTTP 状态码统

计和最近 5 分钟响应时间曲线。在页面顶部选择不同页签，可切换展示实例响应时间、请求数、错误数、HTTP-状态码统计，以及实例概览、JVM 监控、Pod 监控、SQL 调用分析、NoSQL 调用分析、异常分析、上游应用、下游应用、调用链查询、日志等维度的详细分析。

(3) 接口调用页面

在已发布应用页面点击监控->服务接口监控，即可进入接口调用页面，此页面展示当前应用所开放的接口的统计信息。在页面顶部选择不同页签，可切换展示实例响应时间、请求数、错误数统计，以及实例概览、SQL 调用分析、NoSQL 调用分析、异常分析、错误分析、链路上游、链路下游、调用链查询等维度的详细分析。

(4) 数据库调用页面

在已发布应用页面点击监控->数据库调用，即可进入数据库调用页面，该部分展示应用所关联的数据库调用情况。在页面顶部选择不同页签，可切换展示实例响应时间、请求数、错误数统计，以及 SQL 调用分析、异常分析、调用来源、调用链查询等维度的详细分析。

1.5.3、相关页签的关键统计指标说明

- 响应时间：应用、实例调用的平均响应时间，或数据库操作的平均执行响应时间。
- 请求数：应用、实例调用的请求调用次数，或数据库操作的执行次数。
- 错误数：应用、实例调用的错误调用次数，或数据库操作中异常执行次数。
- 概览页签：

上报字段	描述
请求数	应用、实例调用的请求调用次数，或数据库操作的执行次数。
响应时间	应用、实例调用的平均响应时间，或数据库操作的平均执行响应时间。
错误数	应用、实例调用的异常调用次数，或数据库操作的错误次数。
HTTP-状态码统计	应用、实例调用的 5xx、4xx、3xx、200、2xx 状态码的次数

- SQL 分析页签

上报字段	描述
SQL 调用统计	蓝色折线与左 Y 轴为数据库请求数统计，绿色折线与右 Y 轴为数据库响应时间。

- 异常分析页签

上报字段	描述
异常统计	柱状图为该应用、实例、数据库的异常次数。
出现次数	该异常类型的错误出现的次数。

- 调用链查询页签

上报字段	描述
耗时	应用、实例的接口的调用耗时。
状态	应用、实例的接口的调用返回状态，绿色表示正常返回，红色表示抛异常。
TraceId	应用、实例调用的索引 ID，单击可以跳转到该调用链详情。

1.6、系统及 java 类型限制

1.6.1、支持的操作系统

操作系统	版本			
SU SE	SUSE Ente rprise 11 S P4 64bit	SUSE Ente rprise 12 S P1 64bit	SUSE Ente rprise 12 S P2 64bit	SUSE Ente rprise 12 S P3 64bit
Ope nSU SE	13.2 64bit	15.0 64bit	42.2 64bit	
Eul erO S	2.2 64bit	2.3 64bit	2.5 64bit	

CentOS	6.3 64bit	6.5 64bit	6.8 64bit	6.9 64bit
	6.10 64bit	7.1 64bit	7.2 64bit	7.3 64bit
	7.4 64bit	7.5 64bit	7.6 64bit	7.7 64bit
	7.8 64bit	7.9 64bit	8.0 64bit	8.1 64bit
	8.2 64bit			
Ubuntu	14.04 server 64bit	16.04 server 64bit	18.04 server 64bit	
Fedora	24 64bit	25 64bit	29 64bit	
Debian	7.5.0 32bit	7.5.0 64bit	8.2.0 64bit	8.8.0 64bit

ian	9.0.0 64bit
-----	-------------

1.6.2、支持的 java 类型

类型	名称	版本
工具	JDK	jdk7、jdk8
通讯协议	httpClient	apache httpClient3、apache httpClient4、 jdk httpURLConnection
Java 框架	iBatis	2.3.0、2.3.4.726
	mybatis	1.0.0 ~ 1.3.1 (Mybatis-Spring) 、3.0.1 ~ 3.4.5 (Mybatis3)
	spring	3.1.x ~ 5.0.x
	springboot	1.2.x~1.5.x、2.0.4 ~ 2.0.9

	Dubbo	2.5.3~2.6.2 (Dubbo RPC、Dubbo REST) 、 2.8.4 (Dubbo RPC、Dubbo REST)
	gRPC	1.11.x~1.14.x
数据库	MySQL	mysql-connector-java 5.1.X
	Oracle	ojdbc5、ojdbc6、ojdbc14
	Sybase	2.6.0~3.2.1
	mariadb	1.3.x
	voltodb	6.x~7.x
	PostgreSQL	9.0.x, 9.1.x, 9.2.x, 9.3.x, 9.4.x, 42.0.x, 42.1.x
web 服务	Tomcat	6.x、7.x、8.x

器	Jetty	7.6.x~8.0.0, 8.1.x~9.x.x
	JBoss	7.0.0~12.0.0
	undertow	1.4.x
消息队列	ActiveMQ	5.6.x~5.15.x
	RocketMQ	4.1.x~4.2.x
	RabbitMQ	1.3.3+ (spring-rabbit) 、2.7.x (amqp-client) 、 2.6.0、 3.6.5
	Kafka	0.9.0.1~0.10.0.2
NoSQL	Redis	jedis 2.0.0~2.9.0
	Memcache	2.9.0~2.12.3(arcus)

	Mongodb	3.0.x~3.6.x
	Cassandra	2.1.x~3.2.x
	zookeeper	1.0.x (com.github.adyliu.zkclient) 、 0.1.x (com.github.sgroshupf.zkclient)
	ElasticSearch	2.4.x、 5.1.x
Rest Client	Common HTTP	2.x、 3.x、 4.x (httpClient) 、 ALL (HttpURLConnection)

1.7、隐私与敏感信息保护声明

应用性能监控会将采集到的指标信息在控制台页面进行展示，如果您有隐私敏感数据请

注意加密保护，必要时不建议上传。

1.8、基本概念

实例

1 个应用可以部署在多台机器上，实例指的是被监控的应用所部署的机器。

如下图，应用 workflow-activiti-service 部署了 3 台机器，每行对应一个机器，即代表一个实例。

请求数 / 响应时间 / 错误数 / 异常数 ↓↑

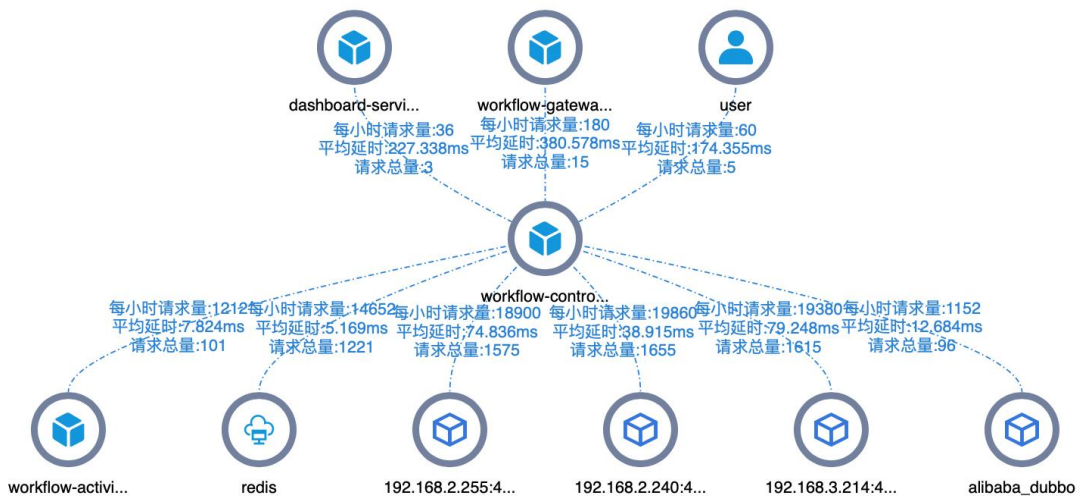
实例 ID	请求数	响应时间	错误数	异常数
...	796	265.266ms	0	0
...	784	350.788ms	0	0
...	782	346.494ms	0	0

探针时 (agent*hour)

1 个探针可以监控 1 个应用实例，1 探针时 (agent*hour) 表示一个探针使用了 1 小时。3600 探针时可支持 1 个探针使用 150 天，也可以是 5 个探针使用 30 天。

拓扑图

拓扑图是一种图形工具，可以用于描述和图形化各个应用之间的连接和依赖关系。如下图，圆圈代表一个服务，可通过 icon 和名称识别具体服务；虚线代表服务之间存在请求，显示每小时请求量、平均延时和请求总量。



二、产品计费

公测期间（2023.10.31~2024.3.31），支持免费使用应用性能监控 APM

公测期间，应用性能监控的使用限制如下：

限制类别	说明
探针数	5 个。同一租户在公测期间可使用的最大探针数为 5。
存储时长	15 天。同一租户在公测期间默认指标和链路的存储时长均是 15 天。

三、快速入门

3.1、为 Java 应用手动安装 Agent

为 Java 应用安装 Agent 后，APM 即可开始监控 Java 应用，您可以查看应用拓扑、调用链路、异常事务、慢事务和 SQL 分析等一系列监控数据。您可以选择以手动方式或脚本方式安装 Agent，本文介绍如何为 Java 应用手动安装 Agent。

下载 Agent

各个资源池的下载地址都不一样，在**接入应用面板**的 **STEP1** 区域下载对应的 Agent。

安装 Agent

1. 进入 Agent 压缩包所在目录并将其解压至任意工作目录下。

```
unzip ctyunArmsAgent.zip -d /{user.workspace}/
```

“{user.workspace}” 是示例路径，此处及以下均请根据具体环境替换为正确的路径。

2.复制以下 JVM 参数，添加到应用服务的启动脚本中。

```
-javaagent:{user.workspace}/ctyunArmsAgent/ctyunArmsAgent.jar
```

```
-Dotel.resource.attributes=service.name=xxx
```

```
-Darms.licenseKey='Jnyb84Wi@857657ULhP2G6B'
```

```
-Dotel.exporter.otlp.endpoint=http://100.64.2.29:27149
```

license.key 和 endpoint 是我们为您自动生成的。

service.name 是您的应用名称，请将 my-service 替换成您自己的应用名。

如需在同一台服务器为一个应用部署多个进程实例，则应在-Dotel.resource.attributes 内容中增加配置：

instance.id=逻辑实例名，使用逗号与其他配置隔开，如若无此配置，则多个进程实例数据将合并为同一

个实例。

启动应用

请在重新启动您的应用，大约 5 分钟后 Agent 将会安装完毕。

结果验证

约 5 分钟后，若 Java 应用出现在**应用列表**页面中且有数据上报，则说明接入成功。

应用名称	应用状态	部署环境	部署方式	发布时间	发布人	操作
workflow-job-service	运行中	prod	微服务应用容器部署	2023-09-15 19:17:29	ctyun104ba80f67c945021675305988	详情
workflow-control-service	运行中	prod	微服务应用容器部署	2023-09-15 19:15:49	ctyun104ba80f67c945021675305988	详情
workflow-activiti-service	运行中	prod	微服务应用容器部署	2023-09-15 19:09:35	ctyun104ba80f67c945021675305988	详情
workflow-center-service	运行中	灰度环境	微服务应用容器部署	2023-09-15 19:06:53	ctyun104ba80f67c945021675305988	详情
workflow-center-service	运行中	prod	微服务应用容器部署	2023-09-15 19:04:34	ctyun104ba80f67c945021675305988	详情
workflow-control-service	运行中	灰度环境	微服务应用容器部署	2023-09-15 18:57:25	ctyun104ba80f67c945021675305988	详情
workflow-activiti-service	运行中	灰度环境	微服务应用容器部署	2023-09-15 18:54:25	ctyun104ba80f67c945021675305988	详情
workflow-job-service	运行中	灰度环境	微服务应用容器部署	2023-09-14 16:58:45	ctyun104ba80f67c945021675305988	详情
workflow-uipc-merge	运行中	prod	通用应用容器部署	2023-09-13 21:29:33	ctyun104ba80f67c945021675305988	详情
lowcode-platform	运行中	prod	通用应用容器部署	2023-09-13 19:05:01	ctyun104ba80f67c945021675305988	详情

10条/页 共 26 条 < 1 2 3 >

3.2、为部署在容器环境的 Java 应用安装探针

借助应用性能监控 APM，您可以对容器环境的应用进行应用拓扑、接口调用、异常事务和慢事务监控、SQL 分析等监控。本文将帮助您将容器环境中的应用接入应用性能监控 APM。

前提

- 1、该账号有可用的 ccse 容器集群。
- 2、准备好 java 应用的镜像包。

安装 cubems

安装应用性能监控 APM 插件 cubems

1.登录[天翼云 CCSE 控制中心](#)

2.在左侧菜单点击**集群**，在右侧页面进入集群详情。

3.在左侧导航栏选择**插件 > 插件市场**，在右侧页面找到 **cubems** 插件进行安装。

开启 APM

如需在创建新应用的同时开启应用性能监控 APM，请按以下步骤操作：

- 1.在[天翼云 CCSE 控制中心](#)左侧导航栏选择**集群**，在右侧页面进入集群详情。
- 2.在左侧导航栏选择**工作负载 > 有状态/无状态**，点击**新增**。
- 3.在新增页面，填写 Deployment 相关信息，点击> **“显示 高级设置” > Pod 标签**，新增 Pod 标签，设置标签名 **armsCubeMsAutoEnable**，标签值 **on**，点击确定。

添加工作负载后，可以在 yaml 文件中查看对应的标签

labels:

armsCubeMsAutoEnable: "on" //接入 AMS 的标签

如下图：

```
1  apiVersion: "apps/v1"
2  kind: "Deployment"
3  metadata:
4    labels:
5      name: "5954aaf6-46cc-4220-9fa0-f9fa36c93571"
6      name: "test-test-9f8j6b-3"
7      namespace: "test1102"
8  spec:
9    replicas: 1
10   selector:
11     matchLabels:
12       name: "5954aaf6-46cc-4220-9fa0-f9fa36c93571"
13   template:
14     metadata:
15       labels:
16         name: "5954aaf6-46cc-4220-9fa0-f9fa36c93571"
17         source: "CCSE"
18         armsCubeMsAutoEnable: "on"
```


结果验证

约 5 分钟后，若 Java 应用出现在**应用列表**页面中且有数据上报，则说明接入成功。

应用名称	应用状态	部署环境	部署方式	发布时间	发布人	操作
workflow-job-service	运行中	prod	微服务应用容器部署	2023-09-15 19:17:29	ctyun104ba80f67c945021675305988	详情
workflow-control-service	运行中	prod	微服务应用容器部署	2023-09-15 19:15:49	ctyun104ba80f67c945021675305988	详情
workflow-activiti-service	运行中	prod	微服务应用容器部署	2023-09-15 19:09:35	ctyun104ba80f67c945021675305988	详情
workflow-center-service	运行中	灰度环境	微服务应用容器部署	2023-09-15 19:06:53	ctyun104ba80f67c945021675305988	详情
workflow-center-service	运行中	prod	微服务应用容器部署	2023-09-15 19:04:34	ctyun104ba80f67c945021675305988	详情
workflow-control-service	运行中	灰度环境	微服务应用容器部署	2023-09-15 18:57:25	ctyun104ba80f67c945021675305988	详情
workflow-activiti-service	运行中	灰度环境	微服务应用容器部署	2023-09-15 18:54:25	ctyun104ba80f67c945021675305988	详情
workflow-job-service	运行中	灰度环境	微服务应用容器部署	2023-09-14 16:58:45	ctyun104ba80f67c945021675305988	详情
workflow-uipc-merge	运行中	prod	通用应用容器部署	2023-09-13 21:29:33	ctyun104ba80f67c945021675305988	详情
lowcode-platform	运行中	prod	通用应用容器部署	2023-09-13 19:05:01	ctyun104ba80f67c945021675305988	详情

10条/页 共 26 条 < 1 2 3 >

四、用户指南

4.1、应用管理

4.1.1、应用总览

以应用为维度，统计整个应用的关键指标，帮助您快速掌握应用的整体状况。

4.1.1.1. 功能入口

(1) 选择目标资源池，并登录 APM 组件控制台

(2) 在左侧导航栏中选择「应用列表」

(3) 在应用列表中选择您想查看的应用，点击「应用名称」打开新的应用详情链接

(4) 在左侧导航栏中选择「应用总览」，您可以在应用总览页面顶部选择「概览」或「拓扑图」页签查看相应信息。

4.1.1.2. 功能说明

概览

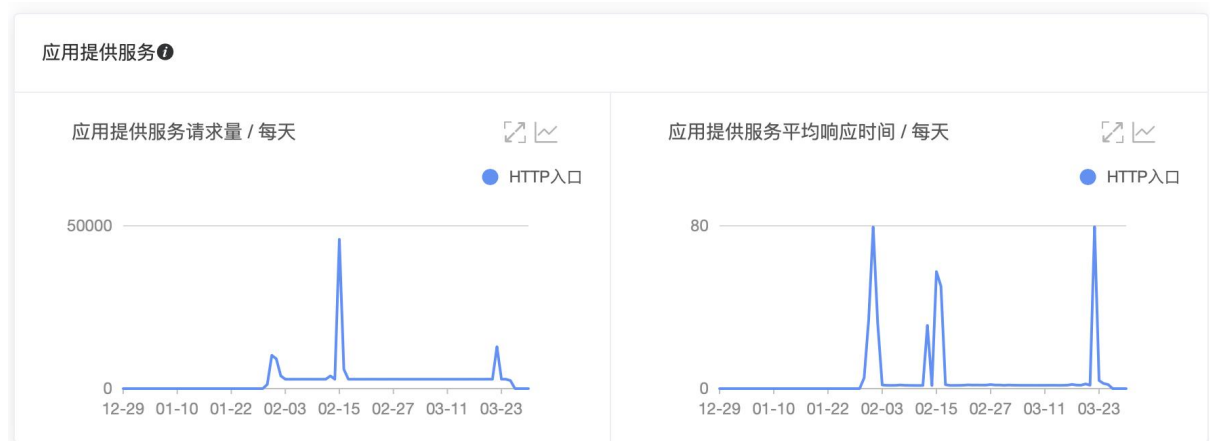
(1) 总览指标

概览	拓扑图	近三个月				
总请求量	平均响应时间	错误数	Full GC	慢SQL ⓘ	异常	慢调用 ⓘ
228093	24.196ms	0次	0次	0次	6个	18个
周同比 - 日同比 -	周同比 - 日同比 -	周同比 - 日同比 -	周同比 - 日同比 -	周同比 - 日同比 -	周同比 - 日同比 -	周同比 - 日同比 -

- **总请求量**：筛选时间段内，应用提供服务请求量+应用依赖服务请求量。
- **平均响应时间**：筛选时间段内，（所有应用提供服务响应时间+所有应用依赖服务响应时间）/总请求量。
- **错误数**：error，筛选时间段内，请求出错的数量，通常指 http 状态码为 4xx、5xx 的请求。
- **FullGC**：筛选时间段内，整堆垃圾回收的次数，回收的区域包括年轻代、老年代以及方法区。
- **慢 SQL**：筛选时间段内，执行时间大于等于慢 SQL 阈值的 SQL 数量，默认 500ms，您可以根据实际情况在「应用设置」中修改。
- **异常**：exception，筛选时间段内，该应用报的异常数。

- **慢调用**：筛选时间段内，响应时间大于等于慢调用阈值的调用数量，默认 500ms，您可以根据实际情况在「应用设置」中修改响应时间阈值。

(2) 应用提供服务



因用户访问该应用而产生的数据，例如用户在浏览器中访问该应用

- **应用提供服务请求量**：筛选时间段内，用户向该应用发起的请求数量
- **应用提供服务平均响应时间**：响应时间是指从用户发起请求到服务端给予反馈的时长，平均响应时间是筛选时间段内，所有请求的响应时间的平均值。

(3) 应用依赖服务



因该应用访问其他服务而产生的数据，例如该应用访问数据库

- **应用依赖服务请求量**：筛选时间段内，该应用向其他服务发起的请求数量
- **应用依赖服务平均响应时间**：响应时间是指从该应用发起请求到其他服务给予反馈的时长，平均响应时间是筛选时间段内，所有请求的响应时间的平均值。
- **应用实例数**：筛选时间段内，有调用行为的应用实例数量。
- **HTTP-状态码统计**
 - 5xx：服务器异常，服务器在处理请求的过程中发生错误
 - 4xx：客户端异常，请求包含语法错误或无法完成请求
 - 3xx：重定向问题，需要进一步操作
 - 2xx：成功，服务器成功接收请求并执行
 - 200：请求成功






(4) 慢调用



该应用访问其他服务时，其他服务响应时间大于等于 500ms（默认 500ms，可在应用设置中修改阈值）的调用，定义为慢调用。显示饼图和详情表，表头显示如下

- **时间**：判定为慢调用的时间点
- **服务名**：被调用的服务名称
- **IP**：被调用的服务的 IP 地址
- **耗时 (ms)**：具体响应时间
- **响应码**：200 表示请求成功，03 表示调用时长超过最大监听时长 15 秒
- **TraceID**:Trace 表示一个完整的请求链路,一个 Trace 包含了多个调用过程 span, TraceID 是该请求链路的唯一标识。

(5) 统计分析

统计分析				
接口名称	统计信息	平均响应时间 (ms)	异常类型	出现次数
/mall//DemoController/test	最大值: 496.388ms 平均值: 1.868ms		java.net.SocketTimeoutException: Read timed out at java.net.SocketInputStream.socketRead0(Native Method) at java.net.SocketInputStream.socketRead(SocketInput	1
/mall/getProductList	最大值: 15653.703ms 平均值: 112.469ms		java.net.SocketTimeoutException: Read timed out at java.net.SocketInputStream.socketRead0(Native Method) at java.net.SocketInputStream.socketRead(SocketInput	1
/mall/getProductsInfo	最大值: 1666.659ms 平均值: 20.786ms		java.net.SocketTimeoutException: Read timed out at java.net.SocketInputStream.socketRead0(Native Method) at java.net.SocketInputStream.socketRead(SocketInput	1
/mall//subtractInventory	最大值: 25.068ms 平均值: 4.601ms		java.net.SocketTimeoutException: Read timed out at java.net.SocketInputStream.socketRead0(Native Method) at java.net.SocketInputStream.socketRead(SocketInput	
/**	最大值: 118.936ms 平均值: 42.633ms		java.lang.RuntimeException: com.netflix.client.ClientException: Load balancer	

以接口维度来统计调用的情况

- **接口名称**: 被调用的接口的名称
- **最大值**: 筛选时间段内, 该接口被调用的响应时间的最大值
- **平均值**: 筛选时间段内, 该接口被调用的平均响应时间
- **平均响应时间**: 筛选时间段内, 每天的平均响应时间的趋势图
- **异常情况**
- **异常类型**: 显示异常明细, 与点击详情按钮看到的内容一致
- **出现次数**: 筛选时间段内, 此类异常出现的次数

拓扑图

(1) 拓扑图



拓扑图是一种以图形化方式展示应用之间关系的图表，帮助开发人员或运维人员了解应用程序的整体结构和运行状况。

拓扑图通常包括以下信息：

- **应用或服务的组成部分：**例如数据库、缓存、消息队列、Web 服务器等。
- **组件之间的依赖关系：**例如一个组件调用另一个组件的接口等。
- **基础指标：**例如请求量、时延等，帮助开发/运维人员了解组件的运行状况和性能状况。

通过分析拓扑图，开发/运维人员可以快速定位应用中的问题，并进行及时的排查和修复。拓扑图还可以帮助开发/运维人员进行容量规划和性能优化，以提高应用程序或服务的性能和可靠性。

(2) 实例健康

通过“实例平均响应时间”判断该应用所有实例的健康程度

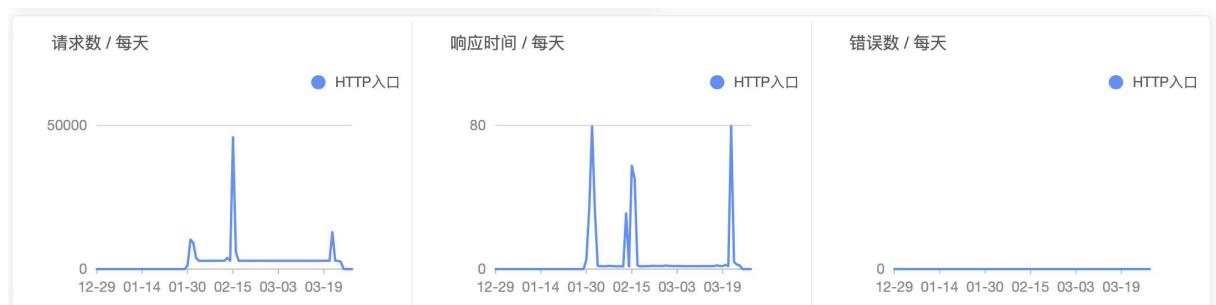
- 健康程度

- 正常：小于 500ms
- 警告：大于等于 500ms，小于 1s
- 严重：大于等于 1s

- 实例的调用详情

- **调用类型**：根据该应用调用和被调用对象的不同来区分类型，包括 http 入口、调用 redis、调用 http、调用 mysql、未知调用等等。
- **调用次数**：筛选时间段内，调用的次数
- **平均响应时间**：筛选时间段内，调用等待响应的平均时长
- **错误率**：筛选时间段内，调用错误次数/调用次数

(3) 请求数、响应时间、错误数



将应用的调用详情按天展示，包括请求数、响应时间、错误数。

4.1.2、应用详情

4.1.2.1、概览

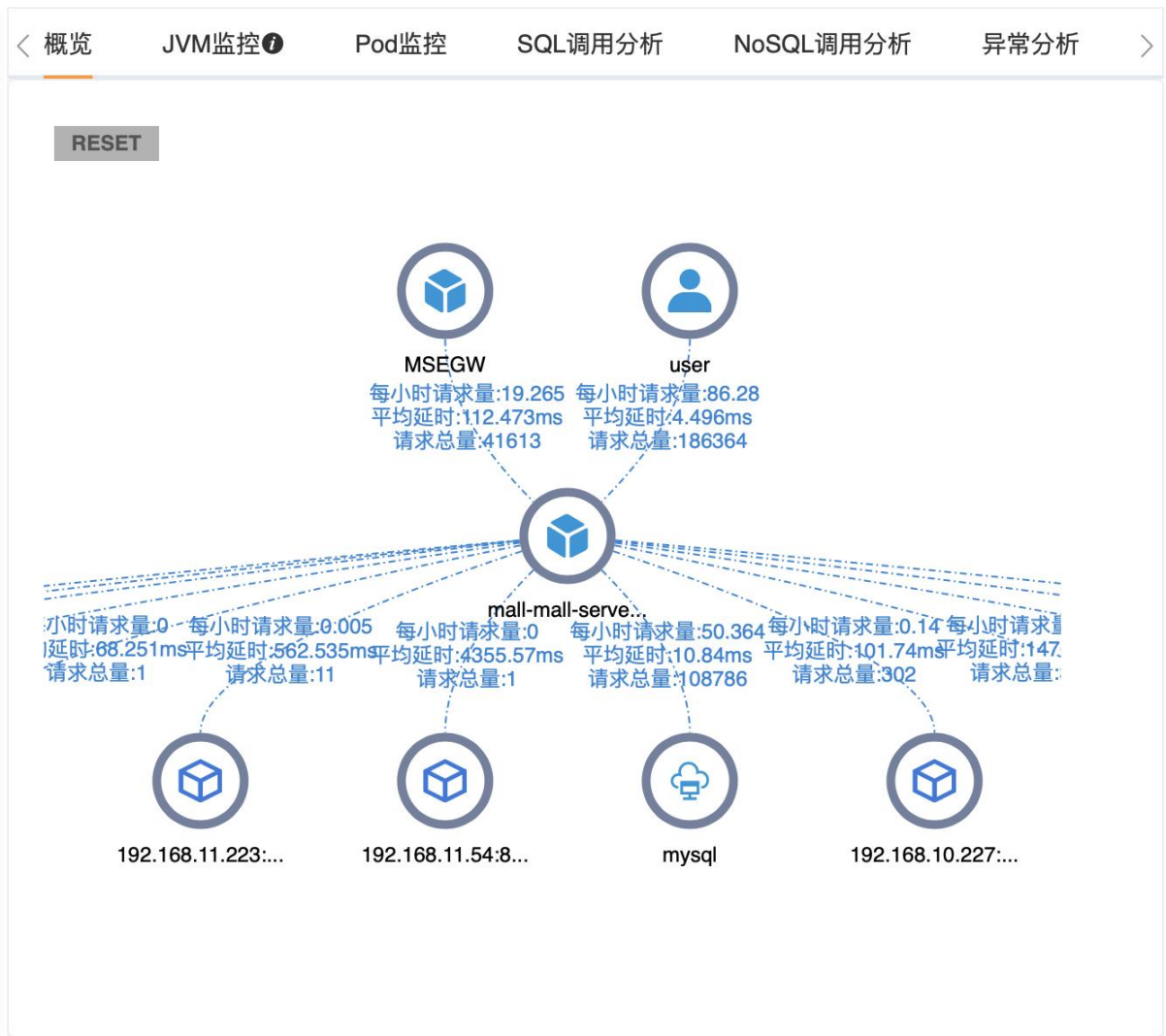
应用的概览可以提供核心监控数据，帮助您快速掌握某个应用的具体情况。

功能入口

- (1) 选择目标资源池，并登录 APM 组件控制台
- (2) 在左侧导航栏中选择「应用列表」
- (3) 在应用列表中选择您想查看的应用，点击「应用名称」打开新的应用详情链接
- (4) 在左侧导航栏中选择「应用详情」或「接口调用」，您可以在应用详情页面切换至「概览」页签，在左侧关键指标中选择不同的应用实例/接口，可查看该应用实例/接口相应的概览信息。

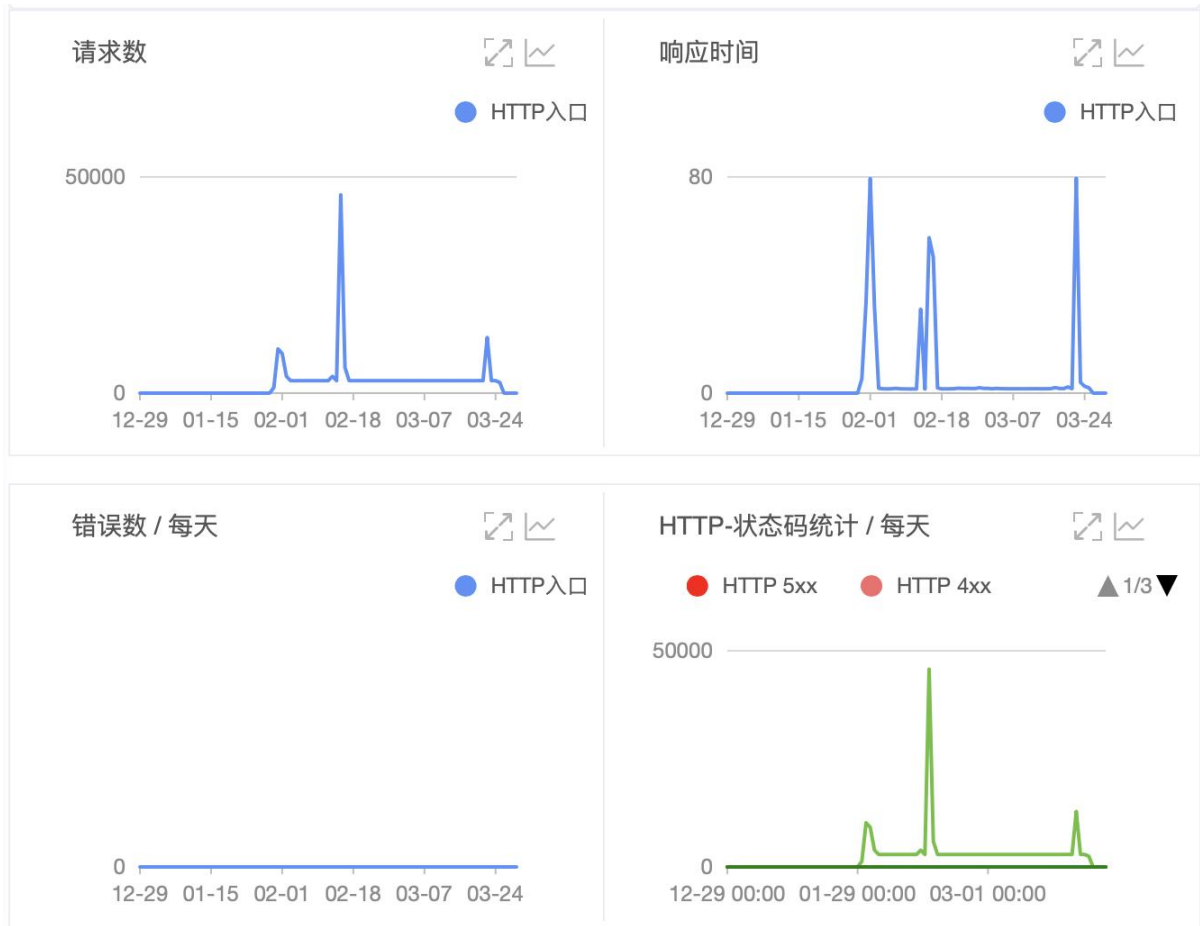
功能说明

- (1) 展示拓扑图





拓扑图是一种以图形化方式展示应用之间关系的图表，帮助开发人员或运维人员了解应用程序的整体结构和运行状况。详情参见“[应用总览-2.1、拓扑图-（1）拓扑图](#)”

(2) 展示请求数、响应时间、错误数、HTTP-状态码统计



- **请求数:** 该应用实例/接口在筛选时间段内的 HTTP 入口的总请求数
- **响应时间:** 该应用实例/接口在筛选时间段内的 HTTP 入口请求的日均响应时间
- **错误数:** 该应用实例/接口在筛选时间段内的 HTTP 入口请求的错误数
- **HTTP-状态码统计:** 该应用实例/接口在筛选时间段内的 HTTP 入口请求的状态码
 - 5xx: 服务器异常, 服务器在处理请求的过程中发生错误
 - 4xx: 客户端异常, 请求包含语法错误或无法完成请求
 - 3xx: 重定向问题, 需要进一步操作
 - 2xx: 成功, 服务器成功接收请求并执行
 - 200: 请求成功

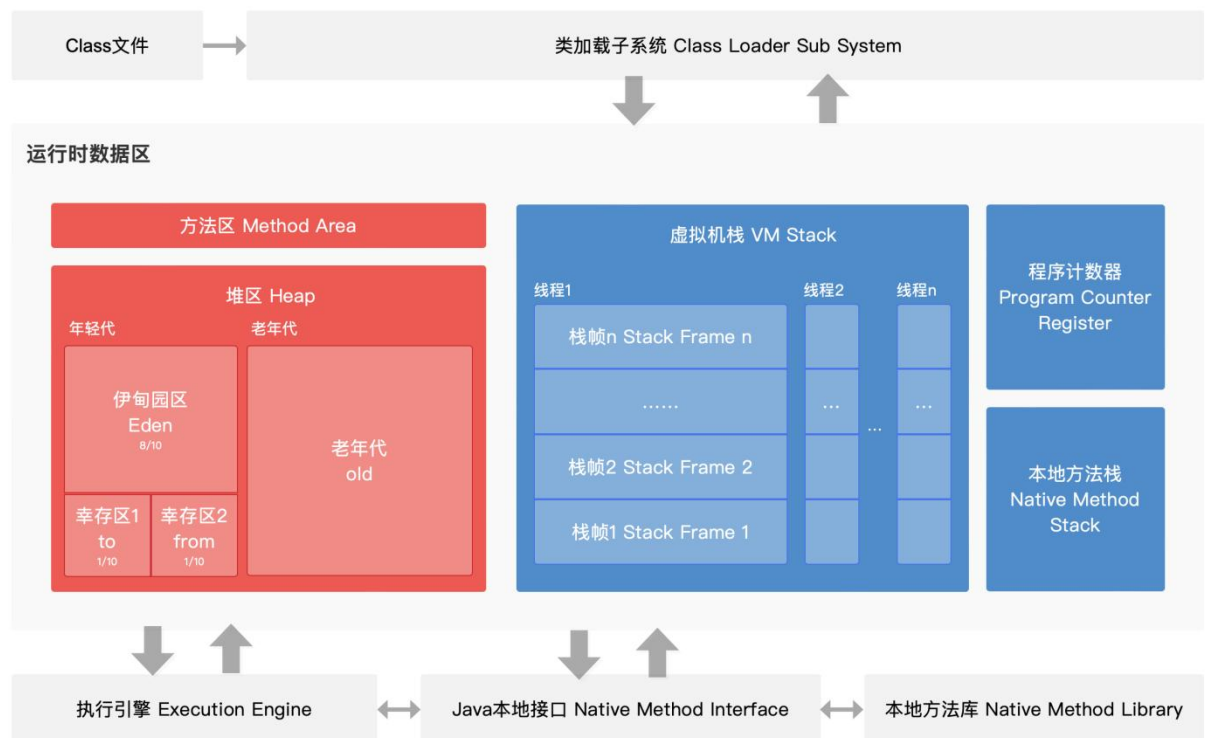
统一交互操作说明:

- 将光标移到统计图上，可以查看光标所至时间点的数据详情。
- 单击图标，可以将当前图表放大显示。
- 单击图标，查看该指标在某个时间段的统计情况或对比不同日期同一时间段的统计情况。

4.1.2.2、JVM 监控

Java 虚拟机 (Java Virtual Machine, JVM) 是一种可以执行 Java 字节码的虚拟机，它是 Java 平台的核心组成部分之一。JVM 负责将 Java 字节码翻译成特定平台上的机器指令，使得 Java 程序可以在各种不同的平台上运行。

JVM 监控即 Java 虚拟机监控，用于监控重要的 JVM 指标。



- JVM 内存包括堆 (heap) 内存、非堆 (non-heap) 内存。堆内存用于存储对象实例，而非堆内存用于存储 Java 虚拟机自身使用的数据和代码等。

- 非堆内存包括直接内存中的元空间（sdk1.8 用的元空间，sdk1.7 用的是线程共享的方法区）、线程私有的虚拟机栈、程序计数器、本地方法栈。
- 堆的释放受到 GC 垃圾回收器的管理，GC 会去找那些很久没有引用地址指向的内存块，把它们清理掉。

JVM 相关数据采集间隔为 15s 一次，采集的是瞬时值中的最大值，总数由多个数据加和而成。

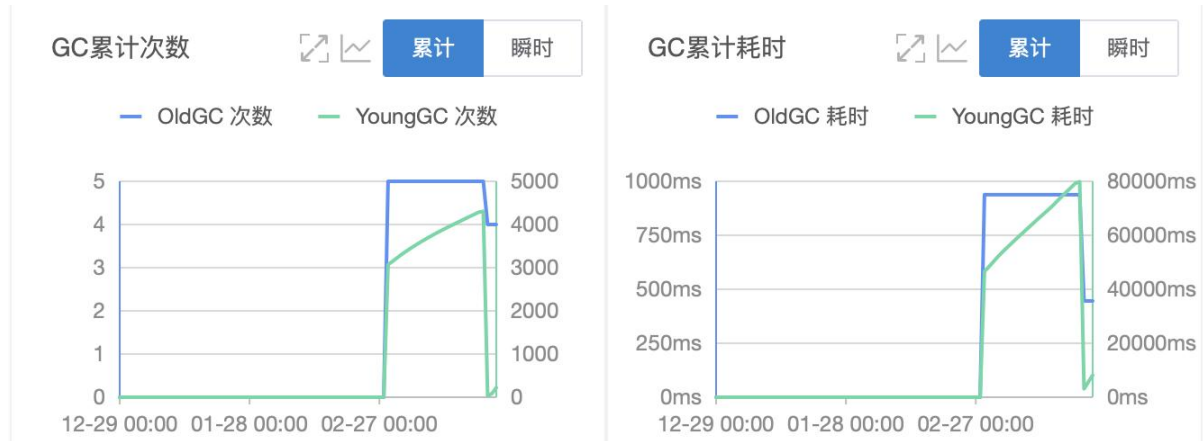
以堆内存图表为例，1 分钟内我们分别采集了老年代、年轻代各 4 次，取其中的最大值展示，总和为最大值之和（因此总和可能大于用户配置的 jvm 内存大小）。

功能入口

- (1) 选择目标资源池，并登录 APM 组件控制台
- (2) 在左侧导航栏中选择「应用列表」
- (3) 在应用列表中选择您想查看的应用，点击「应用名称」打开新的应用详情链接
- (4) 在左侧导航栏中选择「应用详情」，您可以在应用详情页面切换至「JVM 监控」页签，在左侧关键指标中选择不同的应用实例，可查看该应用实例相应的概览信息。

功能说明

- (1) GC



支持切换累计/瞬时两种视角查看

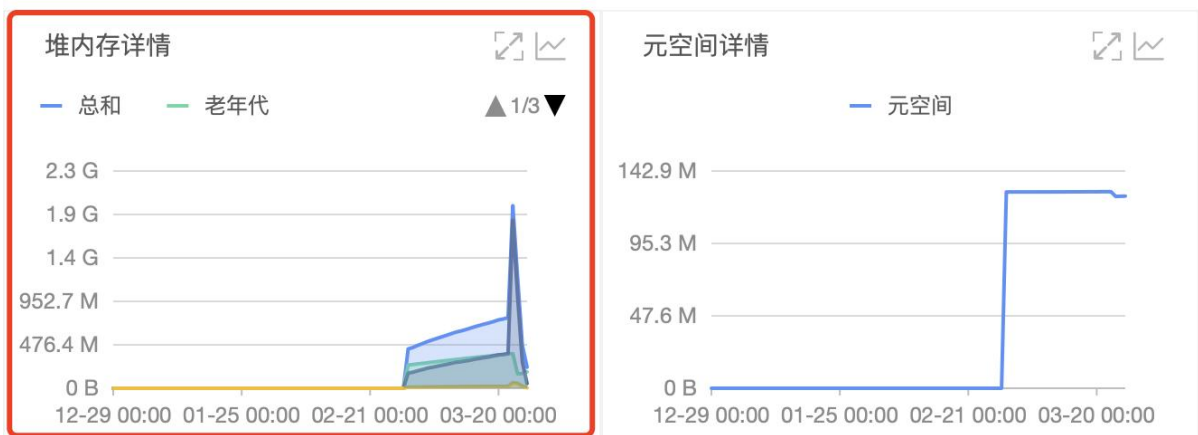
● GC 次数

- OldGC 次数：指的是 Java 虚拟机中的老年代垃圾回收次数。它主要处理老年代中不再使用的对象。
- YoungGC 次数：指的是 Java 虚拟机中的新生代垃圾回收次数。它主要处理新生代中不再使用的对象。

● GC 耗时

- OldGC 耗时：指的是 Java 虚拟机中的老年代垃圾回收累计花费的时间。
- YoungGC 耗时：指的是 Java 虚拟机中的新生代垃圾回收累计花费的时间。

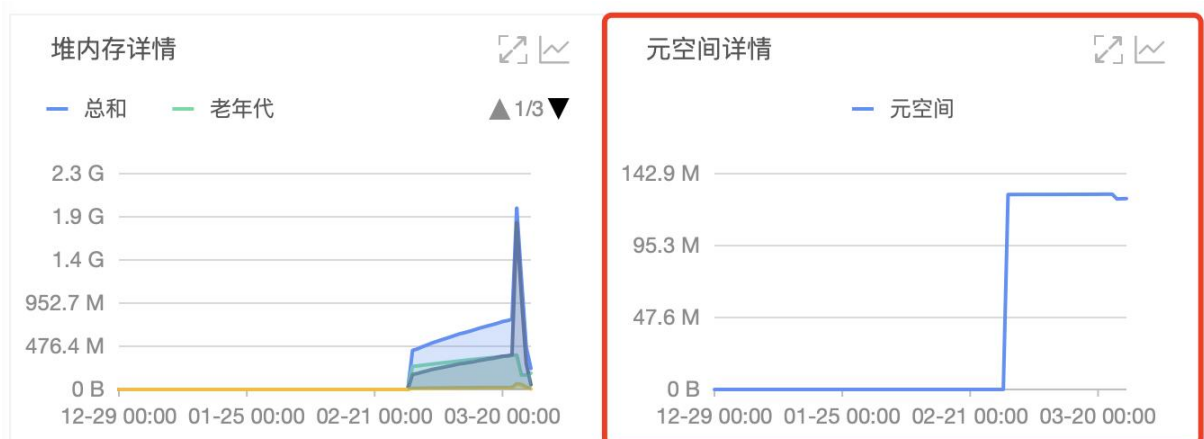
(2) 堆内存详情



堆内存用于存储对象实例，包含 1/3 的新生代和 2/3 的老年代。

- **新生代**：是用来存放新生的对象。分为 Eden 区、SurvivorFrom、SurvivorTo 三个区。Minor GC 进行垃圾回收。JVM 每次只会使用 Eden 和其中的一块 Survivor 区域来为对象服务，新生代实际可用的内存空间为 9/10 (即 90%) 的新生代空间。
 - **Eden 区**：Java 新对象的出生地（如果新创建的对象占用内存很大，则直接分配到老年代）。当 Eden 区内存不够的时候就会触发 Minor GC，对新生代区进行一次垃圾回收。Survivor 区不会触发 Minor GC。
 - **SurvivorFrom 区**：上一次 GC 的幸存者，作为这一次 GC 的被扫描者
 - **SurvivorTo 区**：保留了一次 MinorGC 过程中的幸存者。
- **老年代**：主要存放应用程序中生命周期长的内存对象。Major GC 进行垃圾回收。可能会抛出 OOM (Out Of Memory) 异常

(3) 元空间详情

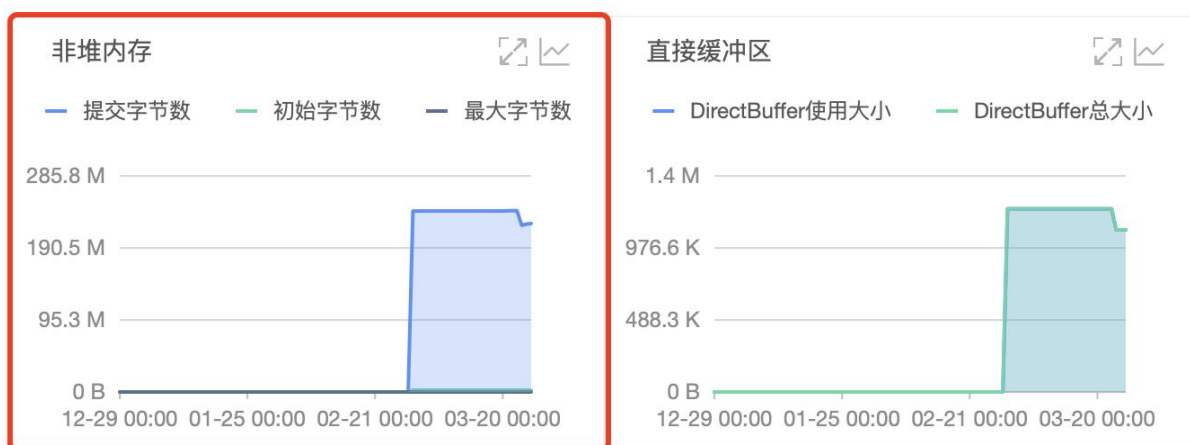


- **元空间 (Metaspace)**：是 JDK1.8 及以上版本引入的概念，用来替代了永久代 (PermGen) 的概念。元空间主要用来存储 class 的元数据信息，包括类的名称、

父类、接口、字段、方法等信息。元空间不再像永久代那样有固定的大小，而是使用本地内存来存储元数据。通过设置元空间大小参数，可以控制元空间的大小。当元空间不足时，JVM 会自动扩容。元空间的大小仅受本地内存限制。

永久代（PermGen）是 JDK1.7 及以下版本中的概念，用来存储类信息和常量池。永久代的大小是固定的，由 JVM 启动时指定。当永久代空间不足时，会导致 OutOfMemoryError 异常。

(4) 非堆内存

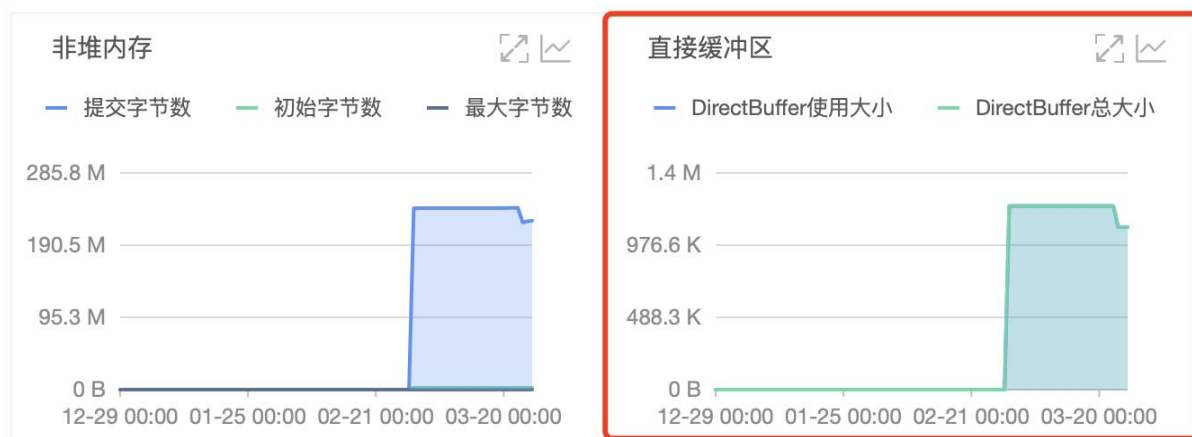


非堆内存用于存储 Java 虚拟机自身使用的数据和代码等等。提交字节数、初始字节数和最大字节数是指非堆内存的三个重要参数，具体含义如下

- **提交字节数 (Committed Bytes)**：指已经被操作系统分配给 Java 虚拟机的非堆内存大小，包括已经使用的和未使用的部分，是在 Java 虚拟机运行时动态确定的。
- **初始字节数 (Initial Bytes)**：指 Java 虚拟机启动时申请的非堆内存大小，也就是 Java 虚拟机最初分配的非堆内存大小。
- **最大字节数 (Max Bytes)**：指 Java 虚拟机能够申请的非堆内存的最大值。当 Java 虚拟机需要更多的非堆内存时，可以向操作系统请求更多的内存，直到达到最大字节数为止。

这些指标可以用来观察非堆内存的大小变化，以便我们根据具体的应用程序调整非堆内存的参数设置（例如使用“-XX:MaxDirectMemorySize”参数来设置非堆内存的最大字节数等等），以保证应用程序能够正常运行，并且不会造成不必要的内存浪费。

(5) 直接缓冲区

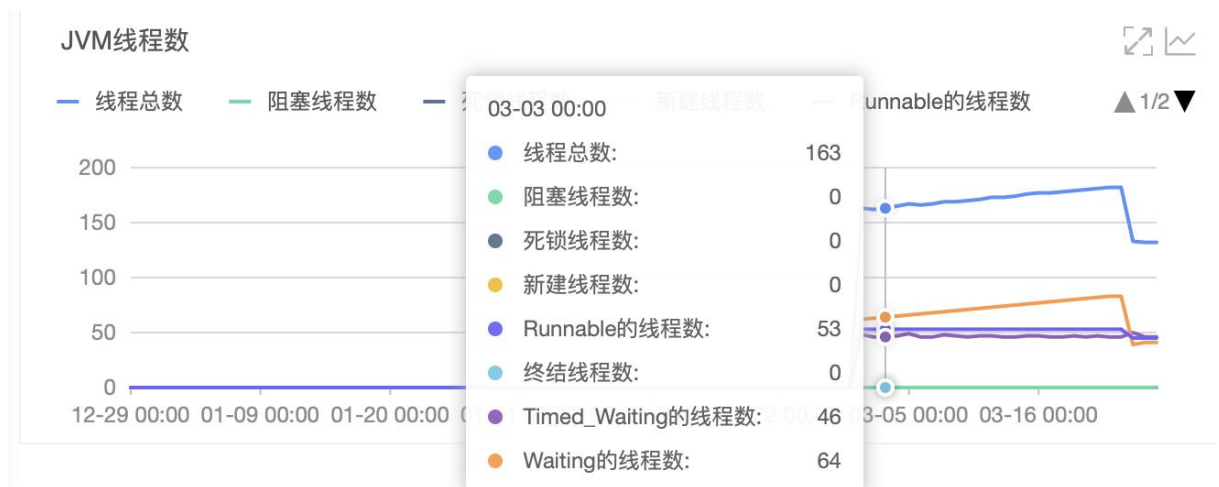


直接缓冲区（Direct Buffer）通常也称为直接内存（Direct Memory），它是一种可以直接访问操作系统内存空间的缓冲区。与普通的 Java NIO 缓冲区不同，直接缓冲区不需要将数据从 Java 堆内存复制到直接内存，而是直接将数据存储存储在直接内存中，从而避免了数据拷贝的开销，提高了 IO 操作的效率。

- **DirectBuffer 总大小**：指已经被 Java 虚拟机分配的所有 DirectBuffer 缓冲区的总大小，包括已经使用的和未使用的部分。
- **DirectBuffer 使用大小**：指当前正在被使用的 DirectBuffer 缓冲区的总大小。

DirectBuffer 总大小和使用大小都是在运行时动态确定的。这些指标可以用来判断直接缓冲区的使用情况。由于直接内存不受 Java 虚拟机的垃圾回收机制控制，所以如果不及及时释放直接内存，可能会导致内存泄漏或者 OutOfMemoryError 等问题。因此，我们需要关注直接内存的使用情况，以便及时释放不再需要的内存空间，可以使用 ByteBuffer 的 cleaner() 方法或者手动调用 System.gc() 方法来释放直接内存。

(6) JVM 线程数



线程（Thread）是操作系统能够进行运算调度的最小单位，是程序执行的基本单元。

在 Java 中，每个线程都是一个独立的执行路径，它可以独立地执行代码、访问变量和资源，与其他线程并发执行。



JVM 线程数是指在 Java 虚拟机中活跃的线程数，也就是当前正在执行的 Java 线程数量，包括用户线程（比如计算、IO 操作等等）和守护线程（比如垃圾回收、内存监控等等）。

- **线程总数**：指当前 Java 虚拟机中的线程总数，包括用户线程和守护线程。
- **阻塞线程数**：指当前因为等待某些条件而被阻塞的线程数量，例如等待 IO 操作完成、等待锁释放等。
- **死锁线程数**：指当前处于死锁状态的线程数量，即两个或多个线程相互等待对方释放资源，从而导致所有线程都无法继续执行。
- **新建线程数**：指当前正在创建的线程数量，这些线程尚未开始执行任何任务。
- **Runnable 线程数（可运行线程数）**：指当前处于可运行状态的线程数量，这些线程已经准备好被调度执行，但是可能还没有得到 CPU 的时间片。

- **终结线程数**：指已经被终止但是还没有被垃圾回收的线程数量，这些线程的 run() 方法已经执行完毕，但是线程对象还没有被回收。
- **Timed_Waiting 的线程数（限时等待线程数）**：指当前正在等待一段时间后才能继续执行的线程数量，例如使用 Thread.sleep() 方法或 Object.wait(long) 方法进行限时等待的线程数量。
- **Waiting 的线程数（等待中线程数）**：指当前正在等待某些条件而被挂起的线程数量，例如使用 Object.wait() 方法进行无限等待的线程数量。

这些指标可以用来监控 Java 虚拟机中线程状态的变化，以及分析线程运行情况和性能瓶颈。

统一交互操作说明：

- 将光标移到统计图上，可以查看光标所至时间点的数据详情。
- 单击  图标，可以将当前图表放大显示。
- 单击  图标，查看该指标在某个时间段的统计情况或对比不同日期同一时间段的统计情况。

4.1.2.3、主机监控

如果您的应用是**主机部署**，则可进行主机监控。

主机监控是指对单个主机（服务器）的资源使用情况进行监控和分析，包括对主机的 CPU、内存、磁盘、网络等资源使用情况的监控和分析。

功能入口

- (1) 选择目标资源池，并登录 APM 组件控制台。

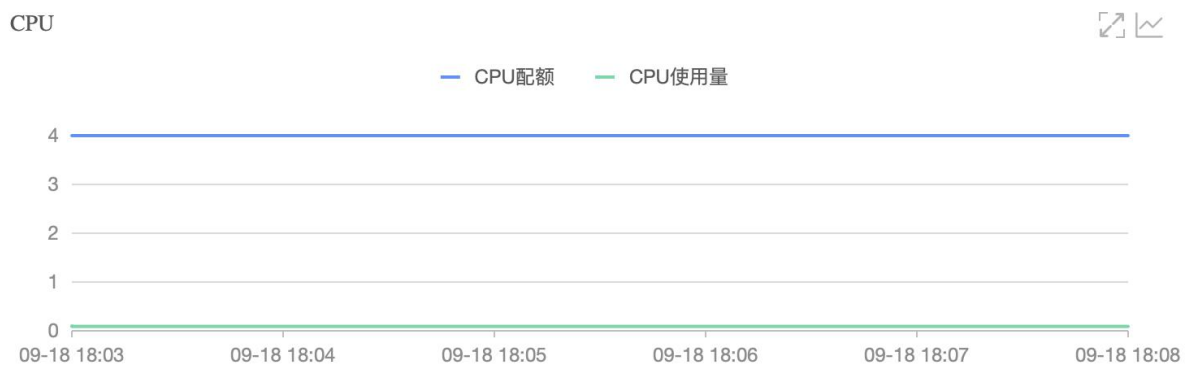
(2) 在左侧导航栏中选择「应用列表」。

(3) 在应用列表中选择您想查看的应用，点击「应用名称」打开新的应用详情链接。

(4) 在左侧导航栏中选择「应用详情」，您可以在应用详情页面切换至「Pod 监控」页签（如果实例是虚机部署，则表示主机监控），在左侧关键指标中选择不同的应用实例，可查看该应用实例相应的概览信息。

功能说明

(1) CPU



指主机中的 CPU 资源使用情况

- CPU 配额 (CPU Quota)：指分配的 CPU 资源上限，单位是 CPU 核数。
- CPU 使用量 (CPU Usage)：指实际使用的 CPU 资源量，单位是 CPU 核数。

(2) 内存



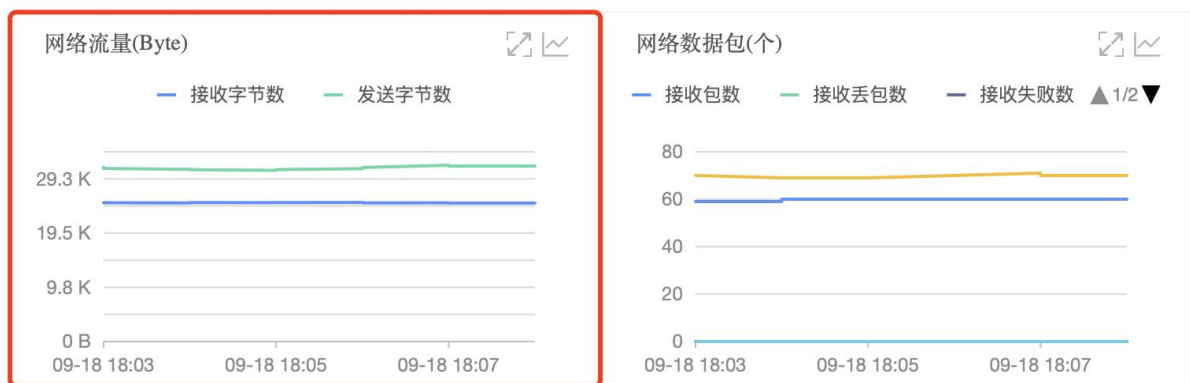
指主机中的内存使用情况

- **内存配额 (Memory Quota)**：指分配的内存资源上限，通常以字节或者二进制字节 (Byte、KB、MB、GB 等) 的形式表示。
- **内存使用量 (Memory Usage)**：指实际使用的内存资源量，通常以字节或者二进制字节 (Byte、KB、MB、GB 等) 的形式表示。

(3) 磁盘

指磁盘的使用情况。

(4) 网络流量

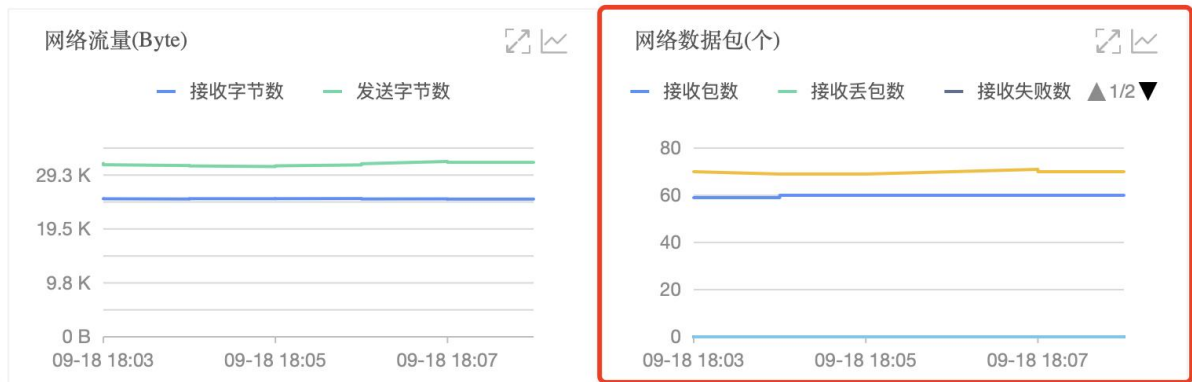


网络流量：网络流量是指通过计算机网络传输的数据量。

- **接收字节数**：也称作入站流量，是指进入主机的数据量。
- **发送字节数**：也称作出站流量，是从主机发送出去的数据量。

监控网络流量可以帮助识别网络拥塞、带宽使用情况和异常流量。

(5) 网络数据包





网络数据包：是在计算机网络中传输的数据单元。

- **接收包数**：接收包数是指网络接口（网卡）成功接收到的数据包的总数量。它表示从网络中接收到的数据包的计数。
- **接收丢包数**：接收丢包数指的是在接收过程中丢失的数据包数量。这表示在数据包传输过程中，一些数据包未能成功到达目的地或未能被接收，导致丢包。
- **接收失败数**：接收失败数是指在接收过程中发生的错误数量。这些错误可能与数据包传输、网络协议或硬件设备有关，例如校验错误、帧错误等。
- **发送包数**：发送包数是指通过网络接口成功发送的数据包的总数量。它表示从主机发送到网络的数据包的计数。
- **发送丢包数**：发送丢包数指的是在发送过程中丢失的数据包数量。这表示在数据包传输过程中，一些数据包未能成功到达目的地或未能被接收，导致丢包。
- **发送失败数**：发送失败数是指在发送过程中发生的错误数量。这些错误可能与数据包传输、网络协议或硬件设备有关，例如校验错误、帧错误等。

监控网络数据包可以提供有关网络通信活动和传输负载的信息。

统一交互操作说明：

- 将光标移到统计图上，可以查看光标所至时间点的数据详情。

- 单击图标，可以将当前图表放大显示。
- 单击图标，查看该指标在某个时间段的统计情况或对比不同日期同一时间段的统计情况。

4.1.2.4、Pod 监控

如果您的应用采用**容器化部署**，则可进行 Pod 监控。

Pod 是 Kubernetes 中最小的可部署单元，是一组容器的集合，它们共享存储和网络资源，通常运行在同一个工作节点上。每个 Pod 都有一个唯一的 IP 地址和一个 DNS 名称，可以通过它们进行访问和通信。

Pod 监控包含对 **CPU、物理内存、流量**的监控，是保障 K8s 应用程序稳定性和可靠性的重要任务之一。通过定期监控和分析 Pod 的资源使用情况，您可以及时发现和解决问题，优化应用程序的运行效率和性能。

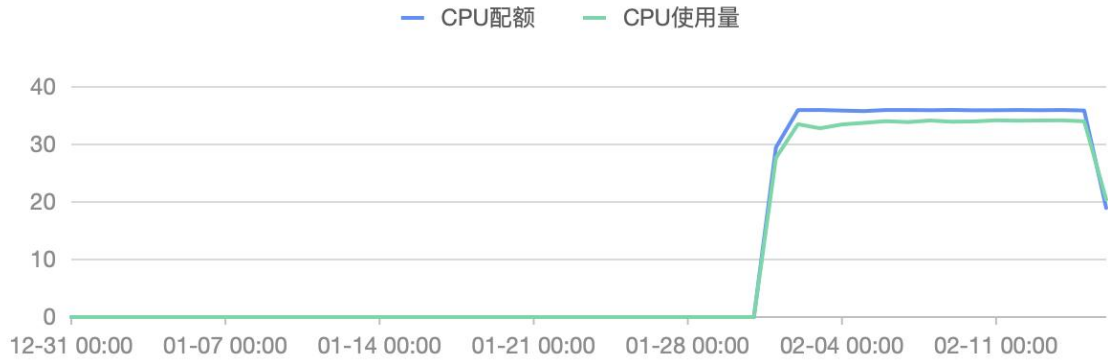
功能入口

- (1) 选择目标资源池，并登录 APM 组件控制台。
- (2) 在左侧导航栏中选择「应用列表」。
- (3) 在应用列表中选择您想查看的应用，点击「应用名称」打开新的应用详情链接。
- (4) 在左侧导航栏中选择「应用详情」，您可以在应用详情页面切换至「Pod 监控」页签，在左侧关键指标中**选择不同的应用实例**，可查看该应用实例相应的概览信息。

功能说明

- (1) CPU

CPU



指 Pod 中运行的容器所占用的 CPU 资源，CPU 资源的使用情况可以用来评估容器的负载情况，以及调整容器的资源分配。

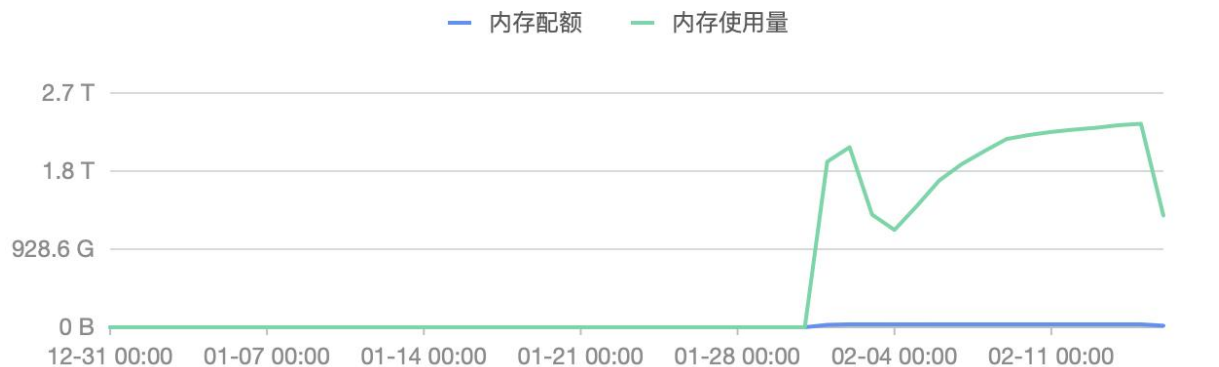
- CPU 配额 (CPU Quota)：指为容器分配的 CPU 资源上限，单位是 CPU 核数。
- CPU 使用量 (CPU Usage)：指容器实际使用的 CPU 资源量，单位是 CPU 核数。

需注意，使用量不能超过配额。例如，一个 Pod 的 CPU 配额为 1 核，表示该 Pod 中所有容器的 CPU 使用量总和不能超过 1 核。如果容器的 CPU 使用量超过了配额限制，容器将会被限制在配额范围内运行，超出部分的 CPU 使用量将会被抛弃。

通过观察 CPU 配额和 CPU 使用量可以合理分析 CPU 资源使用率，避免资源浪费和不足。

(2) 物理内存

物理内存



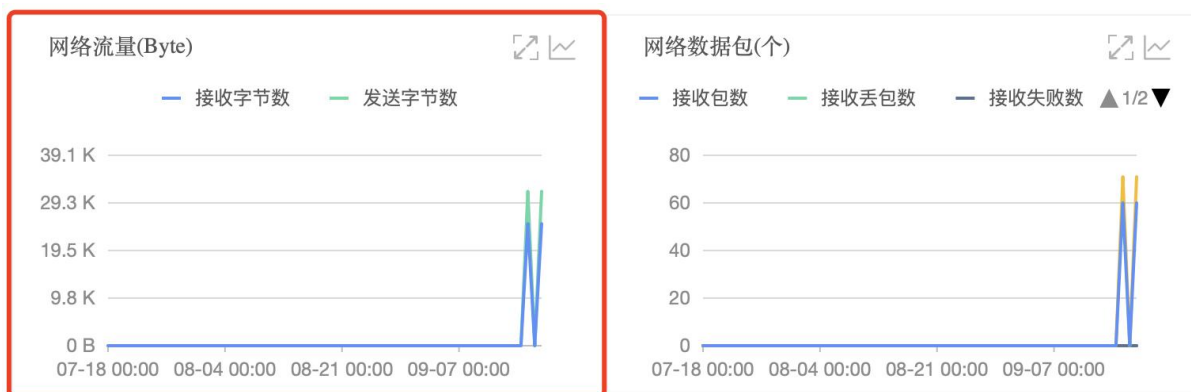
Pod 中运行的容器所占用的物理内存资源，内存资源的使用情况可以用来评估容器的内存使用情况，以及调整容器的资源分配。

- **内存配额 (Memory Quota)**：指为容器分配的内存资源上限，通常以字节或者二进制字节 (Byte、KB、MB、GB 等) 的形式表示。
- **内存使用量 (Memory Usage)**：指容器实际使用的内存资源量，通常以字节或者二进制字节 (Byte、KB、MB、GB 等) 的形式表示。

需注意，使用量不能超过配额。例如，一个 Pod 的内存配额为 1GB，表示该 Pod 中所有容器的内存使用量总和不能超过 1GB。如果容器的内存使用量超过了配额限制，容器将会被限制在配额范围内运行，超出部分的内存使用量将会被抛弃。

通过观察物理内存配额及内存使用量，可以帮助用户根据应用程序的实际内存使用情况，合理设置内存配额，以避免内存资源的浪费和不足。

(3) 网络流量

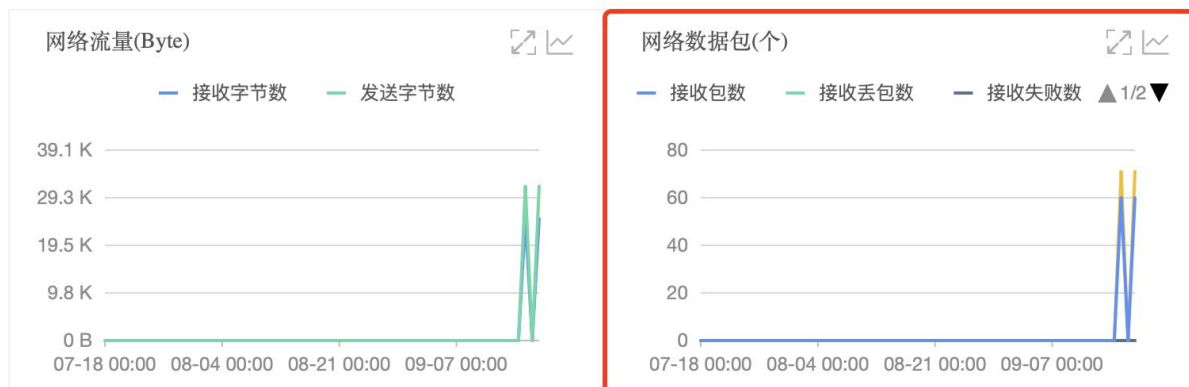


网络流量：Pod 中容器的网络数据量，单位是 Byte。

- **接收字节数**：也称作入站流量，是指进入容器 pod 的数据量。
- **发送字节数**：也称作出站流量，是从容器 pod 发送出去的数据量。

监控网络数据量可以诊断网络性能问题、预测网络容量需求，了解网络的使用情况和资源瓶颈，及时进行网络流量控制和优化，以避免网络拥塞和数据传输失败。

(4) 网络数据包





网络数据包: Pod 中容器之间或者容器与外部网络之间进行通信的基本单位。它是在网络中传输的数据的载体，包含了一定的控制信息和有效载荷数据。

- **接收包数:** 接收包数是指网络接口（网卡）成功接收到的数据包的总数量。它表示从网络中接收到的数据包的计数。
- **接收丢包数:** 接收丢包数指的是在接收过程中丢失的数据包数量。这表示在数据包传输过程中，一些数据包未能成功到达目的地或未能被接收，导致丢包。
- **接收失败数:** 接收失败数是指在接收过程中发生的错误数量。这些错误可能与数据包传输、网络协议或硬件设备有关，例如校验错误、帧错误等。
- **发送包数:** 发送包数是指通过网络接口成功发送的数据包的总数量。它表示从容器 pod 发送到网络的数据包的计数。
- **发送丢包数:** 发送丢包数指的是在发送过程中丢失的数据包数量。这表示在数据包传输过程中，一些数据包未能成功到达目的地或未能被接收，导致丢包。

- **发送失败数**：发送失败数是指在发送过程中发生的错误数量。这些错误可能与数据包传输、网络协议或硬件设备有关，例如校验错误、帧错误等。

监控网络数据包的数量可以帮助管理员诊断网络性能问题和瓶颈，以及进行容器资源分配和负载均衡的调整。例如，如果某个容器接收到的网络数据包数量过多，可能会导致容器负载过高，需要调整容器资源分配或者优化应用程序的网络使用方式。此外，对于网络数据包的监控还可以帮助管理员检测网络攻击和安全漏洞，以及进行网络流量控制和优化。

统一交互操作说明：

- 将光标移到统计图上，可以查看光标所至时间点的数据详情。
- 单击图标，可以将当前图表放大显示。
- 单击图标，查看该指标在某个时间段的统计情况或对比不同日期同一时间段的统计情况。

4.1.2.5、SQL 调用分析

SQL 调用分析通常是指对一段时间内对 SQL 语句的**调用次数**、**耗时的**统计分析。

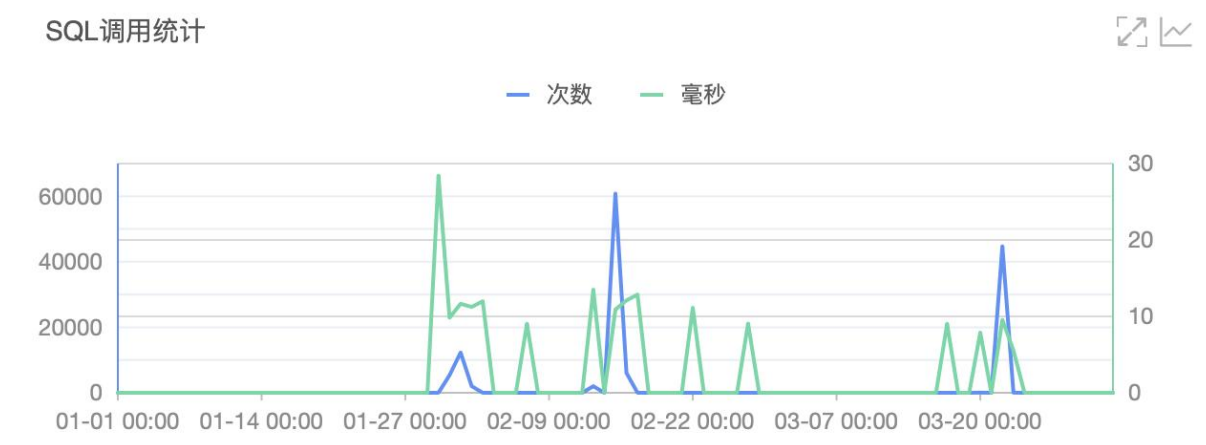
功能入口

- (1) 选择目标资源池，并登录 APM 组件控制台
- (2) 在左侧导航栏中选择「**应用列表**」
- (3) 在应用列表中选择您想查看的应用，点击「**应用名称**」打开新的应用详情链接

(4) 在左侧导航栏中选择「应用详情」或「接口调用」或「SQL 调用」，您可以在应用详情页面切换至「SQL 调用分析」页签，在左侧关键指标中选择不同的应用实例/接口/SQL，可查看该应用实例/接口/SQL 相应的概览信息。

功能说明

(1) SQL 调用统计图



显示该应用在筛选时间段内的所有 SQL 语句的调用次数和平均耗时

(2) SQL 调用统计表



所属应用	数据库类型	SQL 语句	平均耗时(ms) ⚙	调用次数 ⚙
mall-mall-server	mysql	CREATE TABLE IF N...	23.698	18
mall-mall-server	mysql	CREATE TABLE IF N...	19.765	18
mall-mall-server	mysql	SELECT id,name,des...	11.663	25276
mall-mall-server	mysql	SELECT COUNT(?) F...	10.88	53978
mall-mall-server	mysql	SELECT id,name,des...	9.745	54146

以 SQL 语句为维度，详细显示不同 SQL 语句各自的调用次数和平均耗时。

- **所属应用**：显示 SQL 语句所属的应用名称

- **数据库类型**：显示当前应用使用的数据库类型
- **SQL 语句**：显示具体执行的 SQL 语句，不显示 SQL 语句中的具体参数
- **平均耗时**：包括执行时间和等待时间两个方面。
 - **执行时间**：SQL 语句的执行时间，包括 CPU 时间和 IO 时间，可以反映 SQL 语句的性能和效率。
 - **等待时间**：SQL 语句等待的时间，包括等待锁的时间、等待 IO 的时间等，可以反映 SQL 语句的并发性和资源争用情况。
- **调用次数**：SQL 语句被执行的次数，可以反映 SQL 语句的重要性的使用频率。

统一交互操作说明：

- 将光标移到统计图上，可以查看光标所至时间点的数据详情。
- 单击  图标，可以将当前图表放大显示。
- 单击  图标，查看该指标在某个时间段的统计情况或对比不同日期同一时间段的统计情况。

4.1.2.6、NoSQL 调用分析

NoSQL 调用分析是指对不同操作命令的**调用次数**、**耗时**的统计分析。

功能入口

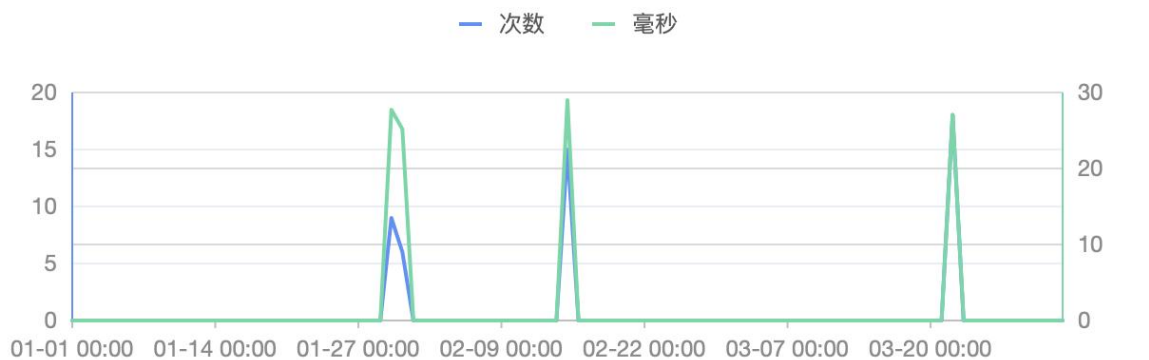
- (1) 选择目标资源池，并登录 APM 组件控制台
- (2) 在左侧导航栏中选择「应用列表」
- (3) 在应用列表中选择您想查看的应用，点击「应用名称」打开新的应用详情链接

(4) 在左侧导航栏中选择「应用详情」或「接口调用」或「NoSQL 调用」，您可以在应用详情页面切换至「NoSQL 调用分析」页签，在左侧关键指标中选择不同的应用实例/接口/NoSQL，可查看该应用实例/接口/NoSQL 相应的概览信息。

功能说明

(1) NoSQL 调用统计图

SQL调用统计



显示该应用在筛选时间段内的所有操作命令的调用次数和平均耗时。



(2) NoSQL 调用统计表

所属应用	NoSQL 类型	操作命令	平均耗时(ms) ▾	调用次数 ▾
mall-mall-server	redis	AUTH ?	27.567	48

- **所属应用**：显示 NoSQL 操作命令所属的应用名称
- **NoSQL 类型与操作命令**：显示 NoSQL 的数据库类型和具体操作命令
- **平均耗时**：（Average Latency）是指执行一组查询请求所需的平均时间。通常用来衡量数据库的响应速度和性能。通常包括执行时间和等待时间两个方面。

- **执行时间**：指处理一个查询请求所需的实际时间，包括进行查询计算、读取数据、返回结果等过程。执行时间受到多种因素的影响，包括数据量、查询复杂度、硬件配置等。
- **等待时间**：指查询请求在队列中等待处理的时间。当 NoSQL 数据库的并发请求数量过多时，会出现查询请求被排队等待的情况，这就会增加查询的等待时间。等待时间通常受到并发请求数量、数据库连接池大小、网络延迟等因素的影响。
- **调用次数**：指在一定时间范围内，应用程序向 NoSQL 数据库发出的该执行命令的请求次数。

统一交互操作说明：

- 将光标移到统计图上，可以查看光标所至时间点的数据详情。
- 单击  图标，可以将当前图表放大显示。
- 单击  图标，查看该指标在某个时间段的统计情况或对比不同日期同一时间段的统计情况。

4.1.2.7、异常分析

异常分析可以帮助您方便了解应用的异常情况。

功能入口

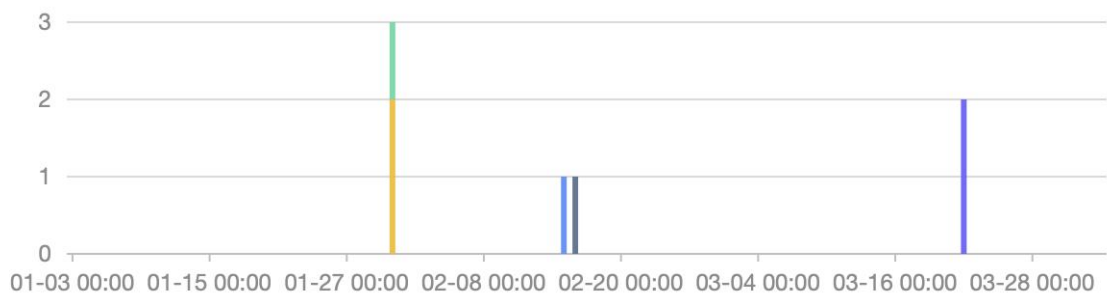
- (1) 选择目标资源池，并登录 APM 组件控制台。
- (2) 在左侧导航栏中选择「应用列表」。
- (3) 在应用列表中选择您想查看的应用，点击「应用名称」打开新的应用详情链接。

(4) 在左侧导航栏中选择「应用详情」或「接口调用」或「SQL 调用」或「NoSQL 调用」，您可以在应用详情页面切换至「异常分析」页签，在左侧关键指标中选择不同的应用实例/接口/SQL/NoSQL，可查看该应用实例/接口/SQL/NoSQL 相应的概览信息。

功能说明

(1) 异常分析图

异常统计



以具体的异常堆栈为维度，展示各个时间段异常出现的次数。

(2) 异常明细

出现次数	异常堆栈 (提示: 如果要忽略异常类型, 请到应用设置>自定义配置>高级设置下的异常过滤字段进行设置。)	操作
1	<pre>method timeout. method: clearZkDataJob, provider: dubbo://192.168.128.87:53605/com.ctg.workflow.api.sys.intf.WorkflowJobService? __micro.service.app.id__=88371681e90873b036cdec53eab94c5a_b31a820b2b88c5f&__micro.service.env__= [{"desc":"edas-canary","priority":-99,"tag":"1694775969887","type"</pre>	详情 调用链查询 调用统计
1	<pre>com.alibaba.dubbo.rpc.RpcException: Invoke remote method timeout. method: clearZkDataJob, provider: dubbo://192.168.128.52:53619/com.ctg.workflow.api.sys.intf.WorkflowJobService? __micro.service.app.id__=88371681e90873b036cdec53eab94c5a_b31a820b2b88c5f&__micro.service.env__=</pre>	详情 调用链查询 调用统计
1	<pre>com.alibaba.dubbo.rpc.RpcException: Invoke remote method timeout. method: clearZkDataJob, provider: dubbo://192.168.128.113:53667/com.ctg.workflow.api.sys.intf.WorkflowJobService? __micro.service.app.id__=88371681e90873b036cdec53eab94c5a_b31a820b2b88c5f&__micro.service.env__=</pre>	详情 调用链查询 调用统计

- **出现次数:** 显示该异常堆栈在筛选时间段内出现的总次数。
- **异常堆栈:** 显示异常明细, 与点击详情按钮看到的内容一致。
- **操作**
 - **详情:** 点击打开异常详情弹窗, 显示具体的异常信息, 且支持复制。

异常详情 ×

```

com.alibaba.dubbo.rpc.RpcException: Invoke remote method timeout. method: clearZkDataJob, provider:
dubbo://192.168.128.113:53667/com.ctg.workflow.api.sys.intf.WorkflowJobService?
__micro.service.app.id__=88371681e90873b036cdec53eab94c5a_b31a820b2b88c5f&__micro.service.env__=[{"desc":"edas-canary","priority":-99,"tag":"1694775969887","type":"canary"}]&anyhost=true&application=activiti-job-consumer&bean.name=com.alibaba.dubbo.config.spring.ServiceBean#24&category=providers&check=false&default.check=false&default.loadbalance=roundrobin&default.reference.filter=regerConsumerFilter&default.retries=0&default.service.filter=providerTraceFilter,dubboExceptionFilter,-exception,regerProviderFilter&default.timeout=30000&dubbo=2.0.2&generic=false&interface=com.ctg.workflow.api.sys.intf.WorkflowJobService&methods=companyStat,listEnableJobMonitorConfig,retryRequestApiJob,ruWfTimerFirstTimeJob,statRunningD
atas,companyAllInfoJob,autoFixFormJob,clearAccountJob,sendMsgExternSysJob,triggerDelayNode,ruWfShellDataJob,checkTemplateAccountDataJob,ruWfTimerExpiredTimeJob,KafkaLEOMonitorJob,syncOrganJob,clearZkDataJob,checkCompanyUseStatJob,dataArchiveJob,completeToDoJob,implicitNotifyJob,apiAccessSaveDataJob,getPollingJob,updateCompanyLimitJob,messageRetryJob,dynamicBuildIndexJob,getDictItem,clearAppJob,sendMsgJob,retryKafkaMsg,tTableDefaultColBuildIndexJob,overTimeRuWfTaskJob,clearnTableJob,noticeSendMsgJob,getAllCompanyIdsWithEnv,aScriptJob,resetCloudExperAccountDataJob,ruWfTimerJob,countExceptionnTraceLogJob,clearAppDeleteData,clearnDataDelete,monitorAppJob,ruAutoDealTaskJob&mse.appId=88371681e90873b036cdec53eab94c5a_b31a820b2b88c5f&payload=8388608&pid=1&protocol=dubbo&register.ip=192.168.128.53&remote.timestamp=1694776291071&serialization=hessian2&side=consumer&timestamp=1694776745320, cause: Waiting server-side response timeout by scan timer.
start time: 2023-09-18 09:00:00.143, end time: 2023-09-18 09:05:00.161, client elapsed: 0 ms, server elapsed: 300018
ms, timeout: 300000 ms, request: Request [id=283194, version=2.0.2, twoway=true, event=false, broken=false,
data=RpcInvocation [methodName=clearZkDataJob, parameterTypes=[], arguments=[], attachments=
{path=com.ctg.workflow.api.sys.intf.WorkflowJobService,
```


- **调用链查询**：点击切换至“调用链”tab，查看该异常堆栈对应的调用链的具体信息。




- **调用统计**：点击后，系统会将该异常堆栈作为筛选条件，重新筛选上方异常分析图的结果。



统一交互操作说明：

- 将光标移到统计图上，可以查看光标所至时间点的数据详情。
- 单击图标，可以将当前图表放大显示。

- 单击  图标，查看该指标在某个时间段的统计情况或对比不同日期同一时间段的统计情况。

4.1.2.8、错误分析

错误 (Error) 通常指在应用程序运行过程中出现的严重问题，比如内存溢出、系统崩溃、IO 异常等。错误通常是无法通过编写代码来处理的，因为它们通常是由操作系统、硬件或其他外部因素引起的。

错误分析可以帮助您更快了解应用的错误信息。

功能入口

- (1) 选择目标资源池，并登录 APM 组件控制台。
- (2) 在左侧导航栏中选择「应用列表」。
- (3) 在应用列表中选择您想查看的应用，点击「应用名称」打开新的应用详情链接。
- (4) 在左侧导航栏中选择「应用详情」或「接口调用」，您可以在应用详情页面切换至「错误分析」页签，在左侧关键指标中选择不同的应用实例/接口，可查看该应用实例/接口相应的概览信息。

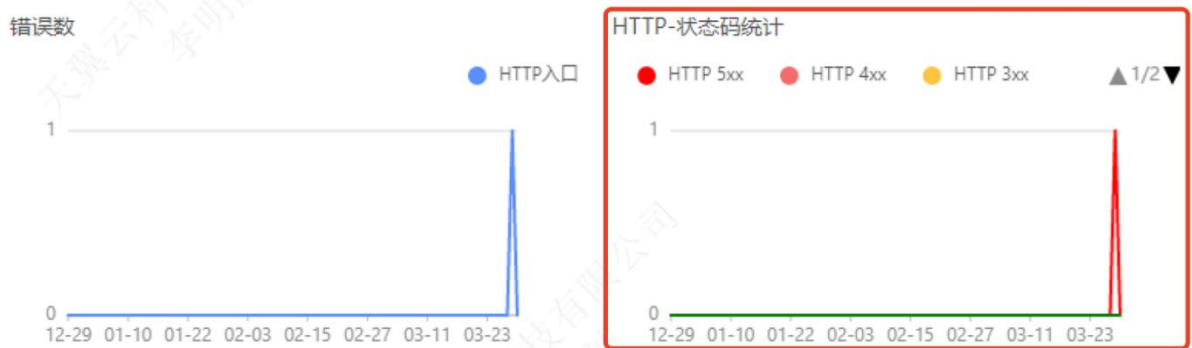
功能说明

- (1) 错误数



显示筛选时间段内，每天出现的错误次数

(2) Http-状态码统计



显示筛选时间段内 Http 不同状态码出现的次数

- 5xx: 服务器异常，服务器在处理请求的过程中发生错误
- 4xx: 客户端异常，请求包含语法错误或无法完成请求
- 3xx: 重定向问题，需要进一步操作
- 2xx: 成功，服务器成功接收请求并执行
- 200: 请求成功



(3) 错误明细表

产生时间	接口名称	所属应用	耗时(ms)	状态	traceId
2023-03-28 19:03:46.590	http://localhost:8777/exception/test	test_web_liuyd2	8.565	●	ee91565ec7163bc0ca9e2db56e109cdd
2023-03-28 19:03:44.196	http://localhost:8777/exception/test	test_web_liuyd2	17.47	●	08ae3a71b8aad2cc6f2eb9579e61b265
2023-03-28 19:02:49.224	http://localhost:8777/users/get?username=12&pageNum=1&pageSize=10	test_web_liuyd2	1989.212	●	f633b5326268a789e8ddf92ae1497483

显示错误明细

- **产生时间**：发现错误的时间点
- **接口名称**：指发生错误时，调用的 API 接口名称
- **所属应用**：指发生错误时，调用的 API 属于哪个应用
- **耗时 (ms)**：指发生错误时，调用 API 的耗时，即 API 调用开始到 API 返回结果的时间间隔。
- **状态**：和状态码对应
- **traceID**：链路 id，指用于跟踪一次 API 调用的唯一标识符，可以跨越不同的应用程序和服务，记录 API 调用的整个调用链路，用于追踪分布式系统中的问题。

统一交互操作说明：

- 将光标移到统计图上，可以查看光标所至时间点的数据详情。
- 单击  图标，可以将当前图表放大显示。
- 单击  图标，查看该指标在某个时间段的统计情况或对比不同日期同一时间段的统计情况。

4.1.2.9、上游应用

上游应用是当前应用/SQL 的调用者。上游应用向当前应用/SQL 发起请求，并等待当前应用/SQL 返回结果。通过上游应用分析可以查看上游应用/SQL 的**请求量、平均延时、错误数**信息。

功能入口

- (1) 选择目标资源池，并登录 APM 组件控制台
- (2) 在左侧导航栏中选择「**应用列表**」
- (3) 在应用列表中选择您想查看的应用，点击「**应用名称**」打开新的应用详情链接
- (4) 在左侧导航栏中选择「**应用详情**」或「**SQL 调用**」或「**接口调用**」，您可以在应用详情页面切换至「**上游应用**」/「**链路上游**」页签，在左侧关键指标中**选择不同的应用实例/SQL/接口**，可查看该应用实例/SQL/接口相应的概览信息。

功能说明



- 以卡片形式显示该应用实例在当前筛选时间段内的上游应用，默认全部展开，可以操作展开/收起，每个卡片上显示一个应用的应用名称、请求量、平均延时（即：耗时/响应时间）和错误数。
- 支持对应用名称进行搜索，支持对请求量/平均延时/错误数进行排序。

通过记录上游应用和下游应用，可以在应用性能监控系统中查看应用程序与应用/SQL/接口之间的调用链路，并追踪请求和响应的流程。

4.1.2.10、下游应用

下游应用指当前应用的被调用方。当前应用程序向下游应用发起请求，并等待下游应用返回结果。通过下游应用分析可以查看下游应用的请求量、平均延时、错误数信息。

功能入口

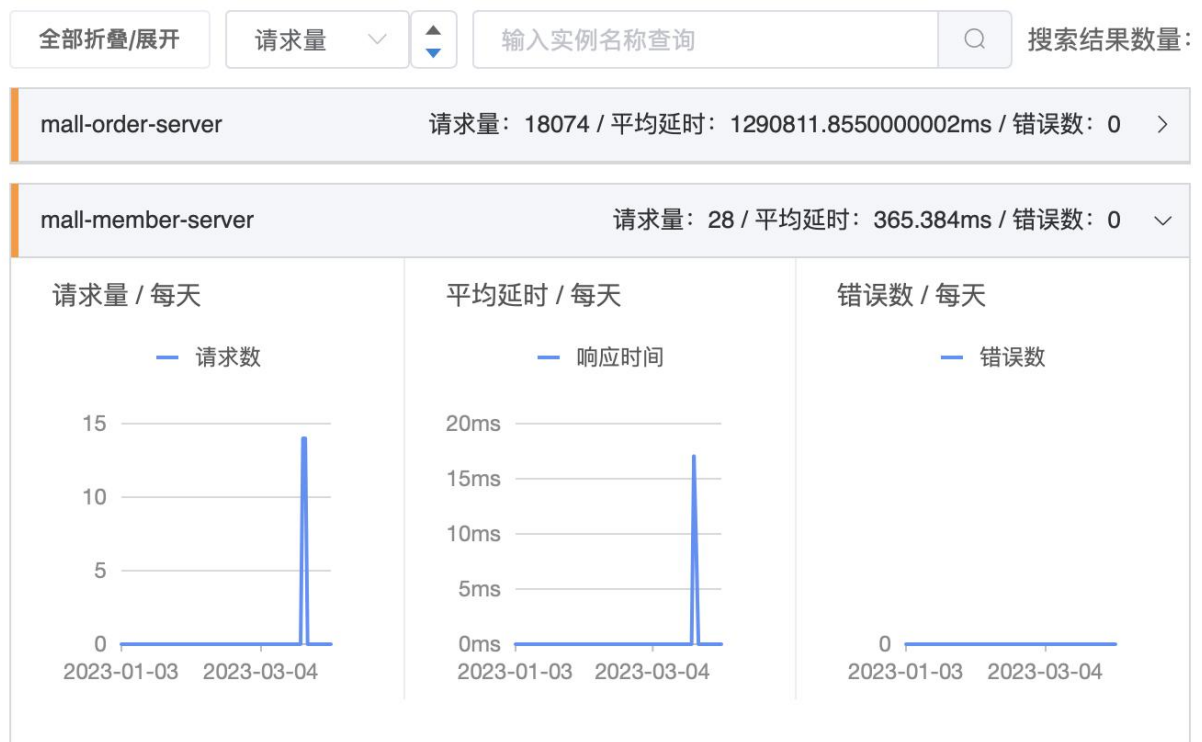
- (1) 选择目标资源池，并登录 APM 组件控制台

(2) 在左侧导航栏中选择「应用列表」

(3) 在应用列表中选择您想查看的应用，点击「应用名称」打开新的应用详情链接

(4) 在左侧导航栏中选择「应用详情」或「接口调用」，您可以在应用详情页面切换至「下游应用」/「链路下游」页签，在左侧关键指标中选择不同的应用实例/接口，可查看该应用实例/接口相应的概览信息。

功能说明



- 以卡片形式显示该应用实例在当前筛选时间段内的下游应用，默认全部展开，可以操作展开/收起，每个卡片上显示一个应用的应用名称、请求量、平均延时（即：耗时/响应时间）和错误数。
- 支持对应用名称进行搜索，支持对请求量/平均延时/错误数进行排序。

通过记录上游应用和下游应用，可以在应用性能监控系统中查看应用程序与应用/接口之间的调用链路，并追踪请求和响应的流程。

4.1.2.11、调用链查询

展示该应用实例/SQL/NoSQL 涉及的调用链信息，包括 TraceID、产生时间、接口名称、耗时、状态信息。

功能入口

- (1) 选择目标资源池，并登录 APM 组件控制台。
- (2) 在左侧导航栏中选择「应用列表」。
- (3) 在应用列表中选择您想查看的应用，点击「应用名称」打开新的应用详情链接。
- (4) 在左侧导航栏中选择「应用详情」或「SQL 调用」或「NoSQL 调用」，您可以在应用详情页面切换至「调用链查询」页签，在左侧关键指标中选择不同的应用实例/SQL/NoSQL，可查看该应用实例/SQL/NoSQL 相应的概览信息。

功能说明

产生时间	接口名称	所属应用	耗时(ms)	状态	TraceId
2023-03-22 21:05:54.773	/mall/getProductList	mall-mall-server	1108.457	●	78e9d4359bc165177feaf683cba9149
2023-03-22 21:34:32.211	/mall/getProductList	mall-mall-server	312.975	●	3b2ae9db79f3172200b51d69b9c2797f
2023-03-22 21:14:03.504	/mall/getProductList	mall-mall-server	161.181	●	93626ea0819a001fff4a8f6970e78ab4

显示筛选时间段内，该应用实例/SQL 涉及的调用链信息

- **产生时间**：该调用链产生的时间点
- **接口名称**：显示发起 API 调用时的接口名称，完整调用链中涉及的接口信息在 [trace 详情中查看](#)
- **所属应用**：显示当前应用实例所属应用的名称，该调用链涉及的其他应用信息在 [trace 详情中查看](#)
- **耗时**：一个完整的链路调用所消耗的时间
- **状态**：与 HTTP 状态码对应
- **TraceID**：调用链的唯一标识。点击显示详情弹窗如“Trace 详情”

4.1.3、其他分析视角

4.1.3.1、接口调用

提供应用接口级别的监控详情。通过丰富的监控分析报表，展示包括 SQL、NoSQL 调用分析；异常、错误分析；链路上下游及调用链分析。

接口是该应用对外提供的各个接口

功能入口

- (1) 选择目标资源池，并登录 APM 组件控制台
- (2) 在左侧导航栏中选择「应用列表」
- (3) 在应用列表中选择您想查看的应用，点击「应用名称」打开新的应用详情链接
- (4) 在左侧导航栏中选择「接口调用」，您可以在应用详情页面切换不同的页签，诸如「概览」、「SQL 调用分析」等等查看相应信息。

功能说明

(1) 关键指标



以接口为维度，展示各个接口的接口名称、请求数、响应时间、错误数和异常数。

(2) 各维度指标详情

概览

和“[应用详情-概览](#)”的指标一致，只是统计维度是接口

SQL 调用分析

和“[应用详情-SQL 调用分析](#)”的指标一致，只是统计维度是接口

NoSQL 调用分析

和“[应用详情-NoSQL 调用分析](#)”的指标一致，只是统计维度是接口

异常分析

和“[应用详情-异常分析](#)”的指标一致，只是统计维度是接口

错误分析

和“[应用详情-错误分析](#)”的指标一致，只是统计维度是接口

链路上游

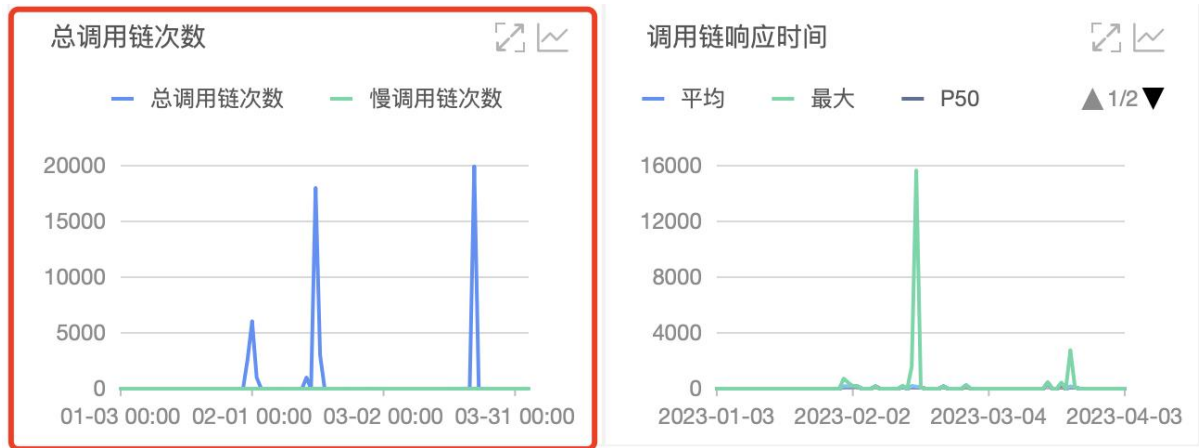
和“[应用详情-上游应用](#)”的指标一致，只是统计维度是接口

链路下游

和“[应用详情-下游应用](#)”的指标一致，只是统计维度是接口

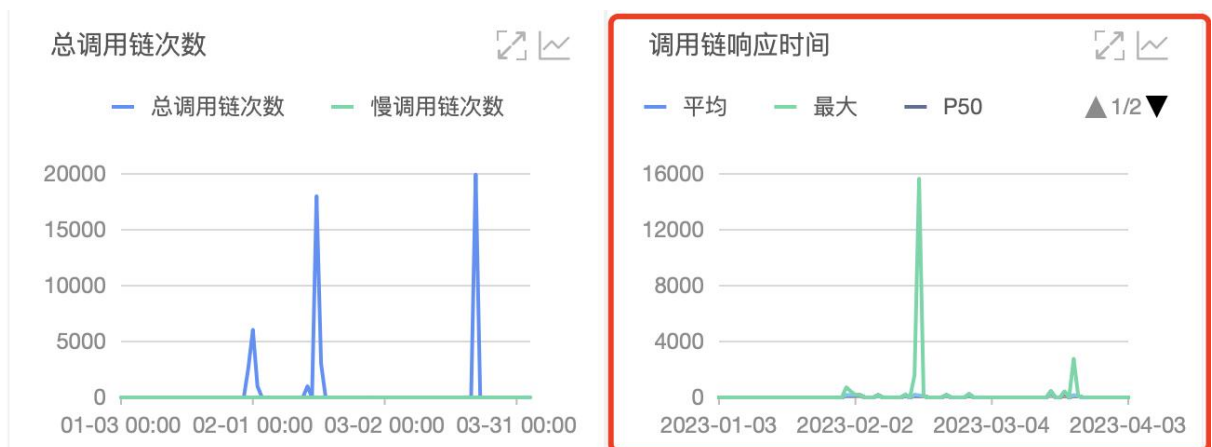
调用链查询

(1) 总调用链次数



- **总调用链次数**：指该接口发起的调用链的总次数，包括成功和失败的调用。
- **慢调用链次数**：指调用链路中响应时间超过预设阈值的调用链路次数，默认阈值为 500ms，该阈值可以根据实际业务需求在「应用设置」中修改。

(2) 调用链响应时间



在筛选时间段内，由该接口发起的调用所属的调用链的耗时

- **平均**：单日响应时间的平均值
- **最大**：单日响应时间的最大值

- **P50**: 单日响应时间的中位数 (Median), 指的是响应时间按从小到大排序后, 位于中间的响应时间, 它表示一组数据的中间值, 也就是 50% 的响应时间低于 P50 值, 50% 的响应时间高于 P50 值。
- **P75**: 表示 75% 的响应时间低于 P75 值, 25% 的响应时间高于 P75 值。
- **P90**: 表示 90% 的响应时间低于 P90 值, 10% 的响应时间高于 P90 值。
- **P99**: 表示 99% 的响应时间低于 P99 值, 1% 的响应时间高于 P99 值。

(3) 调用链明细表

产生时间	接口名称	所属应用	耗时(ms)	状态	TraceId
2023-03-22 21:34:39.108	/mall/getProductList	mall-mall-server	88.246	●	110311fb21 6659edfedd 24ffc2399d2a
2023-03-22 21:34:38.411	/mall/getProductList	mall-mall-server	76.199	●	a620f10884 2524980088 d5b5a20ed9fb
2023-03-22 21:34:32.211	/mall/getProductList	mall-mall-server	312.975	●	3b2ae9db79 f3172200b5 1d69b9c2797f

显示筛选时间段内, 该接口涉及的调用链信息

- **产生时间**: 该调用链产生的时间点
- **接口名称**: 显示发起 API 调用时的接口名称, 完整调用链中涉及的接口信息在 [trace 详情中查看](#)

- **所属应用**: 显示发起 API 调用时的应用名称, 该调用链涉及的其他应用信息在 trace 详情中查看
- **耗时**: 一个完整的链路调用所消耗的时间
- **状态**: 与 HTTP 状态码对应
- **TraceID**: 调用链的唯一标识。点击显示详情弹窗如 “[Trace 详情](#)”

4.1.3.2、SQL 调用

以关系型数据库为维度, 展示当前应用下, 各个 SQL 的监控详情

功能入口

- (1) 选择目标资源池, 并登录 APM 组件控制台
- (2) 在左侧导航栏中选择「**应用列表**」
- (3) 在应用列表中选择您想查看的应用, 点击「**应用名称**」打开新的应用详情链接
- (4) 在左侧导航栏中选择「**SQL 调用**」, 您可以在应用详情页面切换不同的页签, 诸如「**概览**」、「**SQL 调用分析**」等等查看相应信息。

功能说明

关键指标



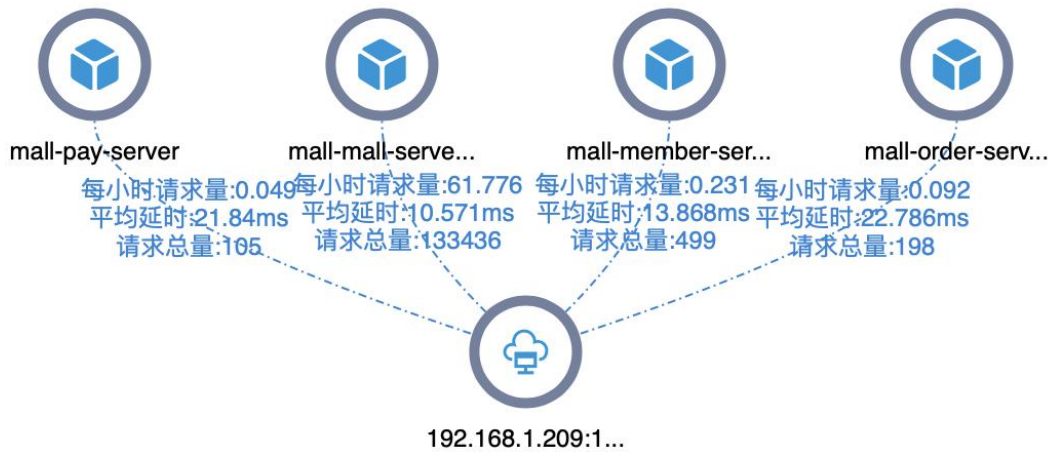
以关系型数据库为维度，展示各个 SQL 的 IP 端口、数据库名称、请求数、响应时间、错误数和异常数。

各维度指标详情

(1) 概览

- 拓扑图

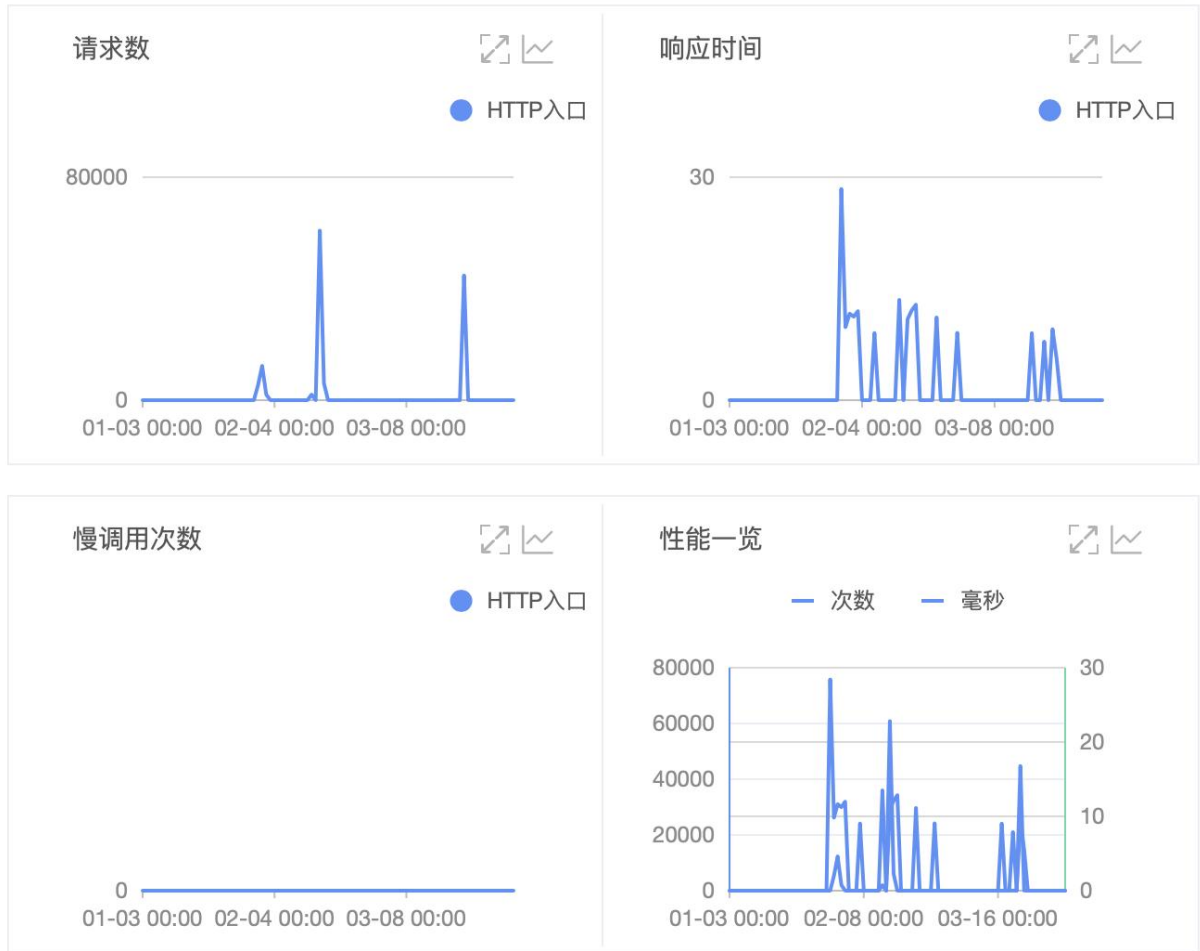
RESET



拓扑图是一种以图形化方式展示应用之间关系的图表，此处展示的是与所选关系型数据库相关的链路信息，可帮助开发人员或运维人员了解应用程序的整体结构和运行状况。

详情参见“[应用总览-2.1、拓扑图-（1）拓扑图](#)”

- 请求数、响应时间、慢调用次数、性能一览



- **请求数:** 该 SQL 在筛选时间段内的 HTTP 入口的总请求数
- **响应时间:** 该 SQL 在筛选时间段内的 HTTP 入口请求的日均响应时间
- **慢调用次数:** 该 SQL 在筛选时间段内的 HTTP 入口请求的响应时间超过阈值的次数, 阈值为 500ms
- **性能一览:** 该应用实例在筛选时间段内的 HTTP 入口请求的次数和响应时间

(2) SQL 调用分析

和“[应用详情-SQL 调用分析](#)”的指标一致，只是统计维度是 SQL

(3) 异常分析

和“[应用详情-异常分析](#)”的指标一致，只是统计维度是 SQL

SQL 查询语句在执行过程中，如果出现语法错误或其他错误，数据库会立即返回错误信息，而不会继续执行查询语句。

而 SQL 查询语句在执行过程中，如果出现性能问题或其他异常情况，例如查询语句执行时间过长、锁定等待、内存使用过高等，数据库会继续执行查询语句，并将相关的异常信息记录下来。

故 SQL 通常会有异常分析，但不会有错误分析

(4) 调用来源

和“[应用详情-上游应用](#)”的指标一致，只是筛选条件是当前 SQL，统计维度是来源接口。

(5) 调用链查询

和“[应用详情-调用链查询](#)”的指标一致，只是统计维度是 SQL

4.1.3.3、NoSQL 调用

以非关系型数据库为维度，展示当前应用下，各个 NoSQL 的监控详情

功能入口

(1) 选择目标资源池，并登录 APM 组件控制台

(2) 在左侧导航栏中选择「[应用列表](#)」

(3) 在应用列表中选择您想查看的应用，点击「[应用名称](#)」打开新的应用详情链接

(4) 在左侧导航栏中选择「[NoSQL 调用](#)」，您可以在应用详情页面切换不同的页签，诸如「[概览](#)」、「[SQL 调用分析](#)」等等查看相应信息。

功能说明

关键指标



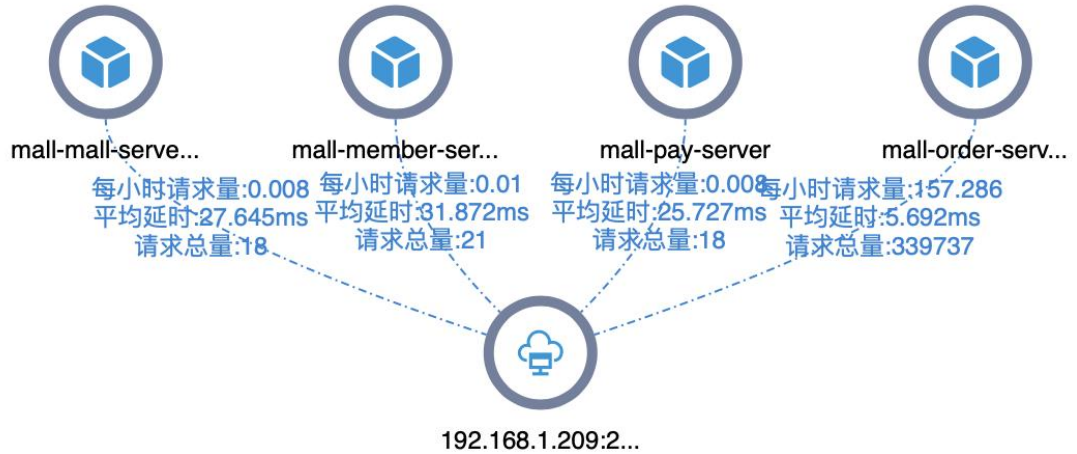
以非关系型数据库为维度，展示各个 NoSQL 的 IP 端口、请求数、响应时间、错误数和异常数。

各维度指标详情

(1) 概览

- 拓扑图

RESET



拓扑图是一种以图形化方式展示应用之间关系的图表，此处展示的是与所选非关系型数据库相关的链路信息，可帮助开发人员或运维人员了解应用程序的整体结构和运行状况。

详情参见“[应用总览-2.1、拓扑图-（1）拓扑图](#)”

- 请求数、响应时间、慢调用次数、HTTP-状态码统计



- **请求数:** 该 NoSQL 在筛选时间段内的 HTTP 入口的总请求数
- **响应时间:** 该 NoSQL 在筛选时间段内的 HTTP 入口请求的日均响应时间
- **慢调用次数:** 该 NoSQL 在筛选时间段内的 HTTP 入口请求的响应时间超过阈值的次数，阈值为 500ms
- **HTTP-状态码统计:** 该 NoSQL 在筛选时间段内的 HTTP 入口请求的状态码
 - 5xx: 服务器异常，服务器在处理请求的过程中发生错误
 - 4xx: 客户端异常，请求包含语法错误或无法完成请求
 - 3xx: 重定向问题，需要进一步操作
 - 2xx: 成功，服务器成功接收请求并执行

○ 200: 请求成功

(2) NoSQL 调用分析

和“[应用详情-NoSQL 调用分析](#)”的指标一致，只是统计维度是 NoSQL

(3) 异常分析

和“[应用详情-异常分析](#)”的指标一致，只是统计维度是 NoSQL

在 NoSQL 分析中，和 SQL 分析类似，通常只会出现异常而不是错误。

因为 NoSQL 数据库在执行查询或操作时，如果出现语法错误或其他错误，通常会立即返回异常信息，而不会继续执行查询或操作。

相反，NoSQL 数据库在执行过程中，如果出现性能问题或其他异常情况，例如查询时间过长、读写冲突等，通常会继续执行查询或操作，并将相关的异常信息记录下来。

(4) 调用链查询

和“[应用详情-调用链查询](#)”的指标一致，只是统计维度是 NoSQL

4.1.3.4、外部调用

指当前应用向外部应用发起的调用，以外部应用的 IP 端口为维度进行展示，可用于定位应用外部调用缓慢或出错的问题。

功能入口

(1) 选择目标资源池，并登录 APM 组件控制台

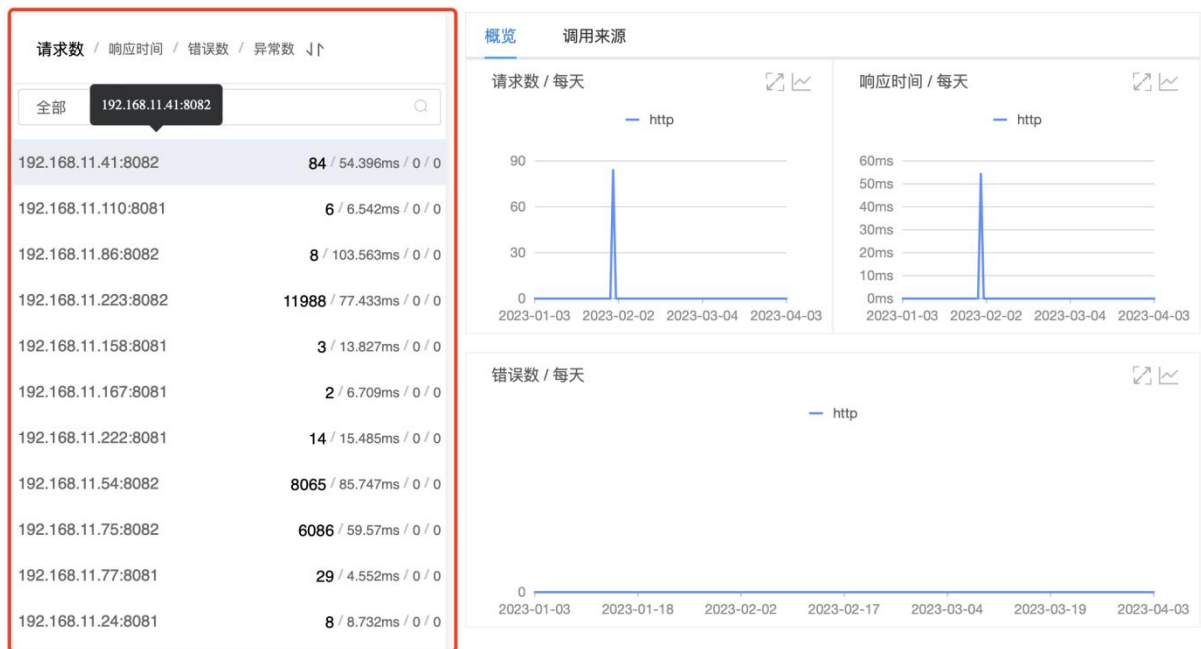
(2) 在左侧导航栏中选择「应用列表」

(3) 在应用列表中选择您想查看的应用，点击「应用名称」打开新的应用详情链接

(4) 在左侧导航栏中选择「外部调用」，您可以在应用详情页面切换不同的页签，诸如「概览」、「调用来源」等等查看相应信息。

功能说明

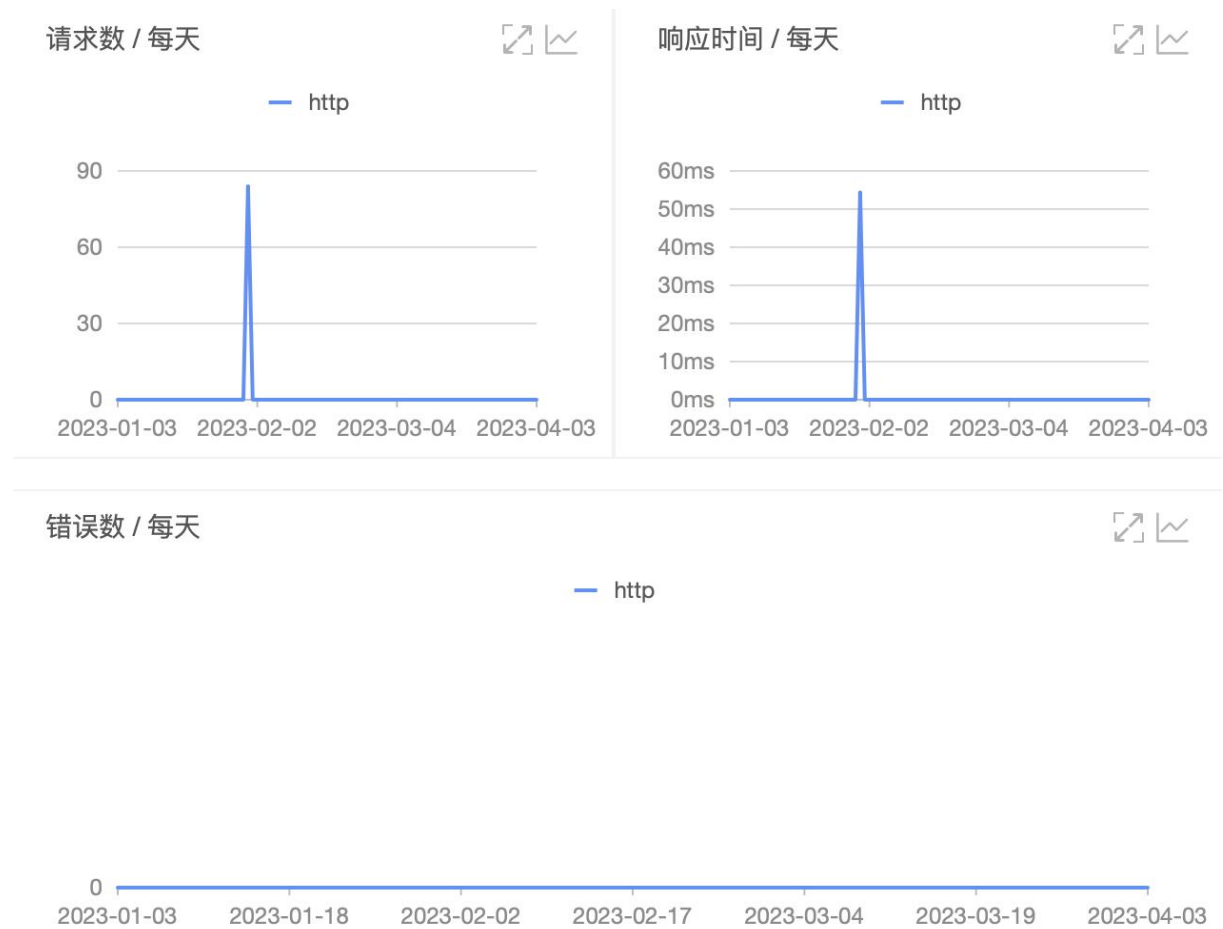
(1) 关键指标



以外部应用的 IP 端口为维度，展示各个外部应用的 IP 端口、请求数、响应时间、错误数和异常数。

支持排序和搜索。

(2) 概览



显示当前筛选时间段内对应外部应用调用的请求数、响应时间和错误数。

(3) 调用来源

和“[应用详情-上游应用](#)”的指标一致，只是统计条件是当前外部应用，统计维度是来源接口。

提供快捷入口，点击「[查看详情](#)」，会切换到“[接口调用-调用链查询](#)”，并选中该来源接口。

4.1.3.5、MQ 监控

MQ 监控是指对消息队列系统进行监控和分析，以保障消息队列的稳定性和性能。

消息队列系统是一种常用的**异步通信机制**，它可以将消息从一个应用程序传递到另一个应用程序，从而实现各种通信和协作方式。

功能入口

- (1) 选择目标资源池，并登录 APM 组件控制台。
- (2) 在左侧导航栏中选择「应用列表」。
- (3) 在应用列表中选择您想查看的应用，点击「应用名称」打开新的应用详情链接。
- (4) 在左侧导航栏中选择「MQ 监控」。

功能说明

(1) 显示通过消息队列发送消息的情况

通过消息队列发送消息的情况

结果数：4



以消息队列为维度显示有多少消息队列发送消息给到当前应用。展示请求数、响应时间和错误数，点击消息队列名称可查看该应用的 MQ 监控详情。

(2) 显示通过消息队列接收消息的情况

通过消息队列接收消息的情况

结果数: 3



以消息队列为维度显示有多少消息队列从当前应用接收消息。展示请求数、响应时间和错误数，点击消息队列名称可查看该应用的 MQ 监控详情。

(3) MQ 监控详情

显示该应用的消息队列发送及接收消息的情况，包括

- 概览：显示拓扑图和关键指标信息。



- 发布端统计：显示请求数、响应时间、错误数的趋势图和明细表。



● 订阅端统计：显示请求数、响应时间、错误数的趋势图和明细表。



● 调用链查询：显示调用链明细。

4.1.4、应用设置

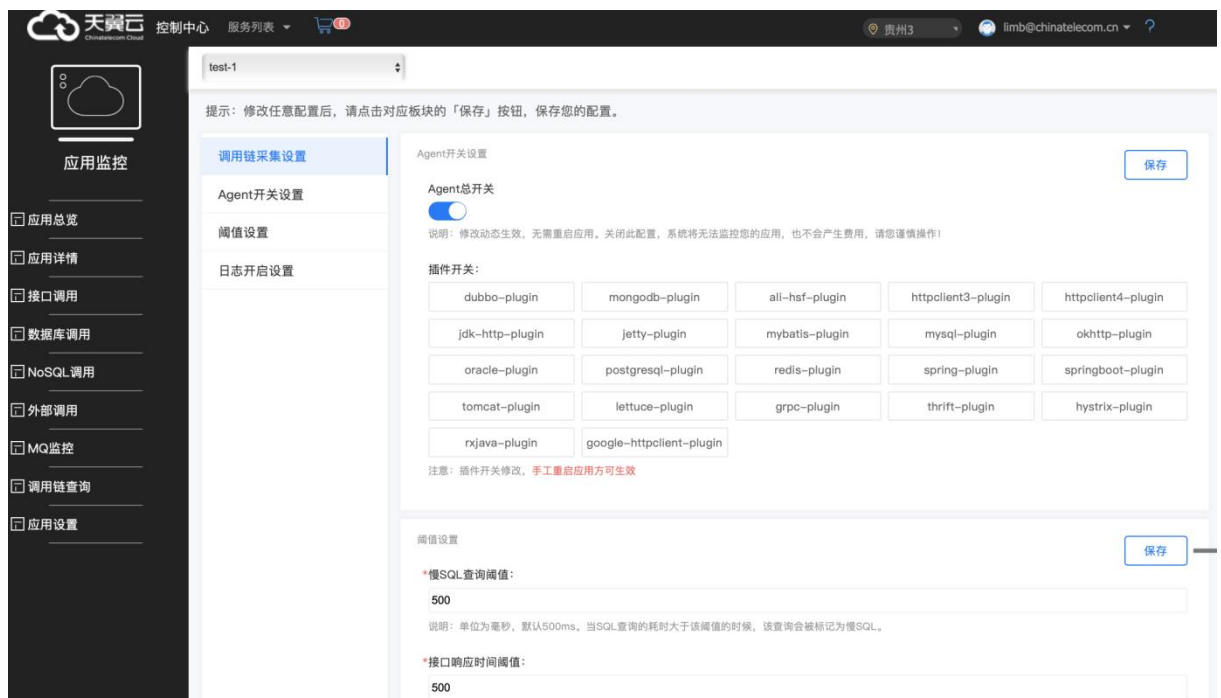
自定义配置可以让您具体某个应用进行单独管理，包括 agent 开关、阈值、调用链采样

率等等。

功能入口

- (1) 选择目标资源池，并登录 APM 组件控制台。
- (2) 在左侧导航栏中选择「应用列表」。
- (3) 在应用列表中选择您想查看的应用，点击「应用名称」打开新的应用详情链接。
- (4) 在左侧导航栏中选择「应用设置」，您可以在应用设置中修改各项可配置信息。

功能说明



(1) Agent 开关设置

支持控制该应用的 agent 是否生效，也可单独针对 agent 操作设置开关。如关闭，agent 将停止采集。

(2) 阈值设置

支持设置慢 SQL 查询阈值、接口响应时间阈值、限流阈值。

- 慢 sql 阈值：影响慢 sql 的判断，下图展示数据会随着配置改变。



总请求量	平均响应时间	错误数	Full GC	慢SQL ⓘ	异常	慢调用
342750	612.596ms	1次	0次	60次	2个	4606
周同比 100% 日同比 43.69%	周同比 100% 日同比 -28.36%	周同比 100% 日同比 100%	周同比 -100.00% 日同比 0%	周同比 100% 日同比 11.11%	周同比 100% 日同比 100.00%	周同比 100% 日同比 100.00%

- 接口响应阈值：影响慢调用的判断，下图展示数据会随着配置改变。



总请求量	平均响应时间	错误数	Full GC	慢SQL ⓘ	异常	慢调用
342750	612.596ms	1次	0次	60次	2个	4606
周同比 100% 日同比 43.69%	周同比 100% 日同比 -28.36%	周同比 100% 日同比 100%	周同比 -100.00% 日同比 0%	周同比 100% 日同比 11.11%	周同比 100% 日同比 100.00%	周同比 100% 日同比 100.00%

- 限流阈值：Agent 端每秒最大可处理请求数。大于该阈值的调用链，不被收集。

(3) 调用链采集设置

支持控制是否采集调用链以及设置调用链采样率。采样率设置说明：

- 采样率默认为 10。
- 调大采样率可能会消耗额外的系统资源，有最大每秒采集条数限制。
- 每秒采集 100 条的情况下，开销在 300m 内存左右。如需调整每秒最大采集数，可修改限流阈值。

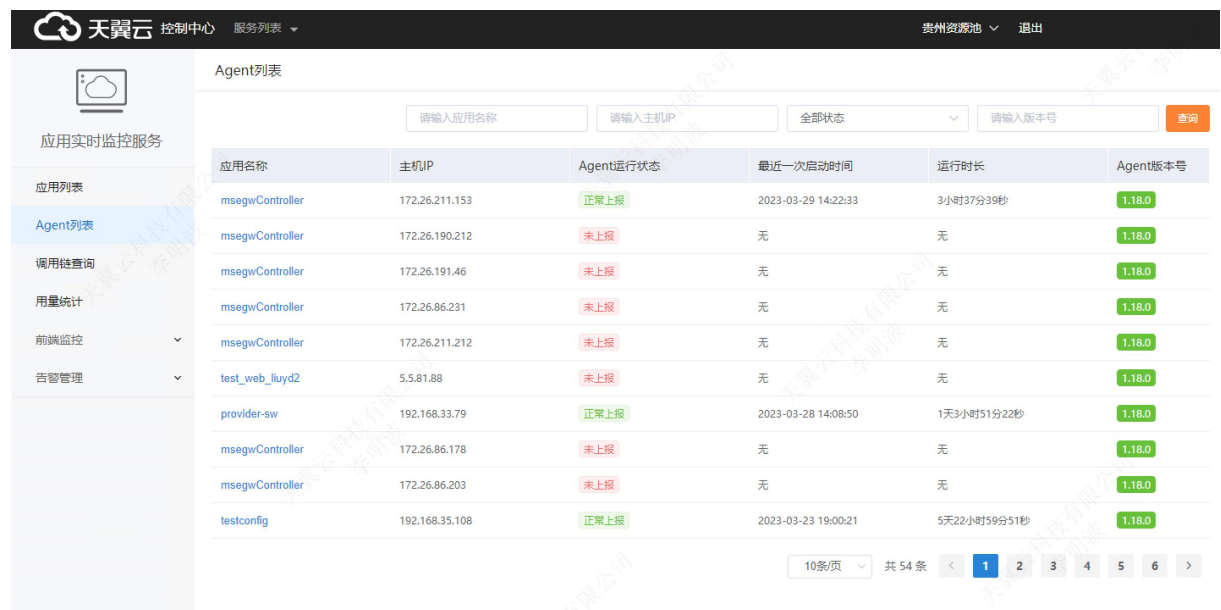
4.2、agent 管理

Agent 列表用于管理您安装的所有探针，包括“正常上报”和“未上报”的探针，显示各个探针的关键指标信息，提供查看应用详情的快捷入口和升级探针的功能入口。

功能入口

- (1) 选择目标资源池，并登录 APM 组件控制台
- (2) 在左侧导航栏中选择「Agent 列表」

功能说明



应用名称	主机IP	Agent运行状态	最近一次启动时间	运行时长	Agent版本号
msegwController	172.26.211.153	正常上报	2023-03-29 14:22:33	3小时37分39秒	1.18.0
msegwController	172.26.190.212	未上报	无	无	1.18.0
msegwController	172.26.191.46	未上报	无	无	1.18.0
msegwController	172.26.86.231	未上报	无	无	1.18.0
msegwController	172.26.211.212	未上报	无	无	1.18.0
test_web_liuyd2	5.5.81.88	未上报	无	无	1.18.0
provider-sw	192.168.33.79	正常上报	2023-03-28 14:08:50	1天3小时51分22秒	1.18.0
msegwController	172.26.86.178	未上报	无	无	1.18.0
msegwController	172.26.86.203	未上报	无	无	1.18.0
testconfig	192.168.35.108	正常上报	2023-03-23 19:00:21	5天22小时59分51秒	1.18.0

(1) 关键信息展示与查询

- **应用名称**：显示该探针（Agent）接入的应用的名称
- **主机 IP**：显示该探针（Agent）接入的应用实例所在的主机 IP

- **Agent 运行状态**: 根据 APM 是否获取到探针 (Agent) 采集数据来判断状态为“正常上报/未上报”
- **最近一次启动时间**: 正常上报的探针 (Agent), 最近一次启动时间
- **运行时长**: 探针 (Agent) 最近一次启动后运行的时长
- **Agent 版本号**: 显示该探针 (Agent) 当前版本号

(2) 查看应用详情

点击「应用名称」, 可以查看具体的应用信息及监控指标明细等等, 包括:

- [应用总览](#)
- 应用详情
- 接口调用
- SQL 调用
- NoSQL 调用
- 外部调用
- MQ 监控
- 调用链查询

(3) 升级探针

如果当前探针 (Agent) 的版本不是最新, 那么可在操作列点击「升级」按钮, 根据升级指引进行探针 (Agent) 升级。

4.3、链路信息

4.3.1、调用链查询

展示当前租户下所有调用链路信息。您可以根据多个筛选条件租户查询您想看的调用链，可以点击「TraceID」查看具体的调用链详情。

功能入口

(1) 查看该租户下所有调用链路

- 选择目标资源池，并登录 APM 组件控制台
- 在左侧导航栏中选择「调用链查询」

(2) 查看某个应用/agent 相关的调用链

- 选择目标资源池，并登录 APM 组件控制台
- 在左侧导航栏中选择「应用列表」
- 在应用列表中选择您想查看的应用，点击「应用名称」打开新的应用详情链接
- 在左侧导航栏中选择「调用链查询」查看该应用实例/接口的调用链信息

功能说明

test_web_for_default 近三个月

参数类型 参数值 添加到查询条件 常用查询条件 查询

查询条件:

TraceID	产生日志时间	接口名称	所属应用名称	状态	耗时(ms)	服务端	客户端	操作
d757fa69d34a5...	2023-03-20 16:16:10.793	/users/get	test_web_for_def...	●	40.383	10.246.1.43:8346	10.246.1.41	查看
8948425f928f0c...	2023-03-20 16:16:09.522	/users/get	test_web_for_def...	●	566.085	10.246.1.43:8346	10.246.1.41	查看
db3a5b60b0a5fc...	2023-03-20 16:16:04.703	/testTag	test_web_for_def...	●	16.073	10.246.1.43:8346	10.246.1.41	查看
db3a5b60b0a5fc...	2023-03-20 16:16:04.703	/testTag	test_web_for_def...	●	16.073	10.246.1.43:8346	10.246.1.41	查看
7ef06b7c4a8950...	2023-03-20 16:16:03.977	/testTag	test_web_for_def...	●	107.932	10.246.1.43:8346	10.246.1.41	查看
7ef06b7c4a8950...	2023-03-20 16:16:03.977	/testTag	test_web_for_def...	●	107.932	10.246.1.43:8346	10.246.1.41	查看
19f7f10591c47cf...	2023-03-20 16:15:58.290	/slow/test	test_web_for_def...	●	1005.082	10.246.1.43:8346	10.246.1.41	查看
5640d92e795c0c...	2023-03-20 16:15:54.637	/jdbc/test	test_web_for_def...	●	14.412	10.246.1.43:8346	10.246.1.41	查看
79e36d61f537bf...	2023-03-20 16:15:54.135	/jdbc/test	test_web_for_def...	●	412.956	10.246.1.43:8346	10.246.1.41	查看
f3c7cd0fe30cb5...	2023-03-20 16:15:48.819	/exception/test	test_web_for_def...	●	8.745	10.246.1.43:8346	10.246.1.41	查看

10条/页 共 82 条 < 1 2 3 4 5 6 ... 9 >

(1) 关键信息展示与查询

- **TraceID**: 调用链的唯一标识。
- **产生日志时间**: 该调用链产生日志的时间点
- **接口名称**: 显示发起 API 调用时的接口名称, 完整调用链中涉及的接口信息在 trace 详情中查看
- **所属应用名称**: 显示当前应用实例所属应用的名称, 该调用链涉及的其他应用信息在 trace 详情中查看
- **状态**: 显示正常/异常
- **耗时**: 一个完整的链路调用所消耗的时间
- **服务端**: 调用链中第一段的被调用的应用的 IP 端口
- **客户端**: 调用链中第一段的发起调用的应用的 IP 地址
- **操作**: 「查看」, 点击显示详情弹窗如 “Trace 详情”

(2) 具体链路详情

详见下章“Trace 详情”

4.3.2、Trace 详情

显示 Trace 的详细信息，包括调用栈及具体每个方法的耗时、日志详情等等。

功能入口

(1) 查看该租户下所有调用链路中某个的链路详情

- 选择目标资源池，并登录 APM 组件控制台
- 在左侧导航栏中选择「调用链查询」
- 点击「TraceID」，打开 Trace 详情弹窗

(2) 查看应用/agent 相关的调用链路中某个的链路详情

- 选择目标资源池，并登录 APM 组件控制台
- 在左侧导航栏中选择「应用列表」
- 在应用列表中选择您想查看的应用，点击「应用名称」打开新的应用详情链接
- 在左侧导航栏中选择「调用链查询」查看该应用实例/接口的调用链信息
- 点击「TraceID」，打开 Trace 详情弹窗

(3) 查看应用实例/接口相关的调用链路中某个的链路详情

- 选择目标资源池，并登录 APM 组件控制台
- 在左侧导航栏中选择「应用列表」
- 在应用列表中选择您想查看的应用，点击「应用名称」打开新的应用详情链接

- 在左侧导航栏中选择「应用详情」或「接口调用」，您可以在应用详情页面切换至「调用链查询」页签，在左侧关键指标中选择不同的应用实例/接口，可查看该应用实例/接口相应的概览信息。
- 点击「TraceID」，打开 Trace 详情弹窗

功能说明

详情 ×

应用名称: MSEGW	接口名称: /mall/getProductList
IP地址: 192.168.10.127:27151	TraceId: 3b2ae9db79f3172200b51d69b9c2797f 查看日志
产生时间: 2023-03-22 21:34:32.209	耗时(ms): 332.171

调用方法	操作	拓展信息	时间轴(ms)
▼ /mall/getProductList	详情 查看日志		332.171
▼ /mall/getProductList	详情 查看日志		312.975
▼ ProductController.getProductList	详情 查看日志		312.286
SELECT mall_demo.t_product	详情 查看日志		15.399
SELECT mall_demo.t_product	详情 查看日志		7.241
▼ HTTP GET	详情 查看日志		284.755
▼ /order/getProductsSales	详情 查看日志		279.355
▼ OrderApiImpl.getProductSales	详情 查看日志		276.467
GET	详情 查看日志		6.528
GET	详情 查看日志		3.647
GET	详情 查看日志		3.885
GET	详情 查看日志		216.98
GET	详情 查看日志		5.368

(1) 基础信息

- **应用名称:** 显示当前选中的“调用方法”所属的应用的名称
- **接口名称:** 显示当前选中的“调用方法”所属的接口的名称
- **IP 地址:** 显示当前选中的“调用方法”所属的应用实例的 IP 地址

- **TraceID:** 显示 TraceID，当前调用链的唯一标识。点击「查看日志」打开“云日志服务-检索分析”页面
- **产生时间:** 发起调用的时间点
- **耗时:** 显示当前选中的「调用方法」从发起调用到返回结果的时间

(2) 调用栈

以调用树的方式显示具体的调用信息

- **调用方法:** 显示调用方法名称
- **操作:**
 - **详情:** 点击详情，打开该调用方法的拓展信息弹窗。显示 Agent、http、主机、Process、Thread、DB、Messaging、目标服务信息。
 - **查看日志:** 点击打开“云日志服务-检索分析”页
- **拓展信息:** 当该调用存在异常/错误信息时，会在此处显示异常堆栈信息
- **时间轴:** 以时间轴的形式显示各个调用方法的耗时

(3) 拓展信息

点击「详情」按钮，打开弹窗如下

User-Agent

名称	kube-probe/1.23
IP	192.168.10.78

Http

URL	http://192.168.11.158:8083/pay/DemoController/test?echo=123
Domain	192.168.11.158
Status-Code	200

主机信息

名称	dsxm-mall-pay-server-10-6d9c8dd77f-scc1b
架构	amd64
操作系统平台	linux
操作系统名称	Linux 5.4.224-1.el7.elrepo.x86_64

显示该方法其他相关信息，包括

● User-Agent

- 名称：接入该应用的探针名称
- IP：该探针接入的应用实例的主机 IP

● Http

- URL (Uniform Resource Locator)：统一资源定位符，是一种用于标识互联网上资源的地址，包括协议类型、主机名、端口号、路径和查询参数等信息。
- Domain (域名)：是指互联网中的机器和服务的名称，类似于电话号码的概念。域名由多个部分组成，按照从右到左的顺序，依次表示顶级域名、二级域名、三级域名等。

○ Status-Code（状态码）：是指 HTTP 服务器返回给客户端的响应状态码，用于表示请求的处理结果。

■ 5xx：服务器异常，服务器在处理请求的过程中发生错误

■ 4xx：客户端异常，请求包含语法错误或无法完成请求

■ 3xx：重定向问题，需要进一步操作

■ 2xx：成功，服务器成功接收请求并执行

■ 200：请求成功

● 主机信息

○ 名称：主机的名称，通常是由用户指定的一个字符串，用于标识主机的身份。

○ 架构：指主机的硬件架构，如 x86、x86_64、ARM 等

○ 操作系统平台：指主机所使用的操作系统的平台，如 Windows、Linux、macOS 等。

○ 操作系统名称：指主机所使用的具体操作系统的名称和版本号，如 Windows 10、Ubuntu 20.04 LTS、macOS Big Sur 等。

● Process（进程）是指正在运行的一个程序的实例

○ Pid（Process ID）：进程的唯一标识符，是一个由操作系统分配的整数，用于标识系统中的不同进程，每个进程都有一个不同的 PID。

○ Command-Line（命令行）：是指启动进程时指定的命令行参数，包括程序的路径、参数等信息。它是一个字符串，可以通过操作系统提供的接口获取到。

● Thread（线程）是指进程中的一个执行单元

- ID (Thread ID) : 线程的唯一标识符, 是一个由操作系统分配的整数, 用于标识系统中的不同线程。每个线程都有一个不同的 ID。

- Name (线程名) : 是指用户指定的线程名称, 用于标识线程的身份。线程名称可以方便用户在程序中进行调试和监控。

- DB (数据库)

- Connection-String (连接字符串) : 是指连接到数据库所需要的信息, 包括主机名、端口号、用户名、密码等。它是一个字符串, 用于在程序中建立到数据库的连接。

- Operation (操作) : 是指对数据库执行的操作, 包括查询、插入、更新、删除等。它是一个字符串, 用于表示当前执行的数据库操作。

- Instance (实例) : 是指数据库的实例名, 用于标识不同的数据库实例。它是一个字符串, 通常是在建立数据库连接时指定的。

- Type (类型) : 是指数据库的类型, 包括关系型数据库、非关系型数据库等。它是一个字符串, 用于表示当前使用的数据库类型。

- Statement (语句) : 是指在数据库中执行的 SQL 语句, 包括查询语句、插入语句、更新语句、删除语句等。它是一个字符串, 用于表示当前执行的 SQL 语句。

- User (用户) : 是指对数据库进行操作的用户, 包括数据库管理员、应用程序用户等。它是一个字符串, 用于表示当前执行操作的用户。

- Messaging (消息传递) 是指在分布式系统中, 通过消息传递实现不同组件之间的通信。

- System（消息系统）：是指消息传递所使用的消息系统，如 Apache Kafka、RabbitMQ、ActiveMQ 等。它是一个软件系统，用于实现异步消息传递。
- Operation（操作）：是指对消息执行的操作，包括发送、接收、确认等。它是一个字符串，用于表示当前执行的操作。
- Destination-Kind（目标类型）：是指消息的目标类型，包括队列（Queue）和主题（Topic）两种。队列用于点对点的消息传递，主题用于发布-订阅模式的消息传递。它是一个字符串，用于表示当前消息的目的地类型。
- **目标服务（Target Service）**是指客户端需要访问的远程服务。
 - 名称（Name）：是指目标服务的名称，用于唯一标识服务。它是一个字符串，通常由服务提供方指定，并在服务注册中心中注册。
 - 类型（Type）：是指目标服务的类型，用于表示服务的功能类别。例如，Web 服务、消息队列服务、数据库服务等。它是一个字符串，通常由服务提供方指定。
 - 实例（Instance）：是指目标服务的实例，用于标识不同的服务实例。在分布式系统中，通常会有多个服务实例提供相同的服务。它是一个字符串，通常由服务注册中心分配。

4.4、统计分析

展示当前租户下的探针用量情况，包括实例数、探针时、Span 存储量

功能入口

- (1) 选择目标资源池，并登录 APM 组件控制台

(2) 在左侧导航栏中选择「用量统计」

功能说明



(1) 关键指标

- **APM 实例数**：筛选时间段内，接入的实例数量
- **APM Agent*Hour**：筛选时间段内，已消耗的探针时
- **Span 存储量**：筛选时间段内，存储过的 Span 总数

(2) 用量趋势

支持切换不同的时间间隔查看 APM 实例数、APM Agent*Hour 和 Span 存储量的变化趋势。

4.5、告警管理

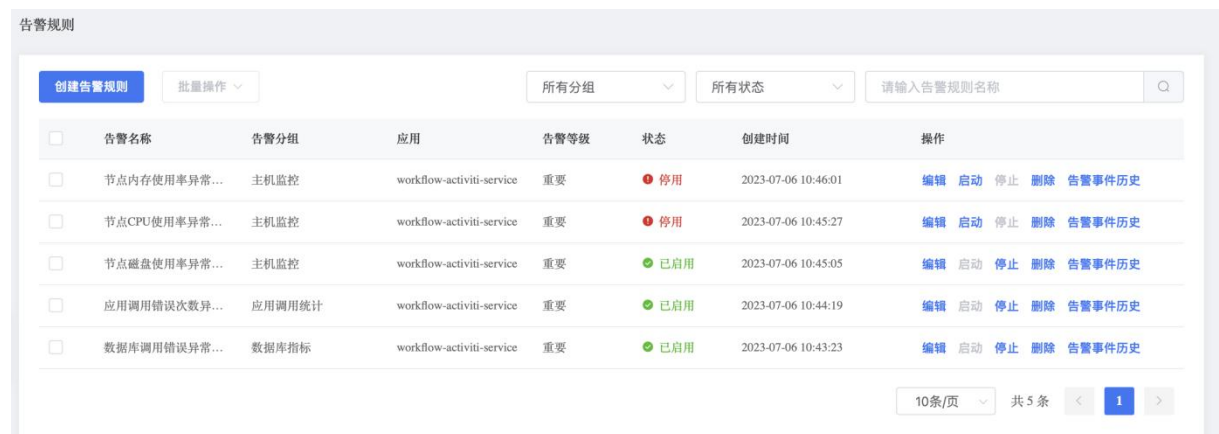
4.5.1、告警规则

展示当前租户下所有告警规则信息，支持增删改查、起停和查看告警事件历史

功能入口

- (1) 选择目标资源池，并登录 APM 组件控制台。
- (2) 在左侧导航栏中选择「告警管理」-「告警规则」。

功能说明



告警规则

创建告警规则 批量操作 所有分组 所有状态 请输入告警规则名称

<input type="checkbox"/>	告警名称	告警分组	应用	告警等级	状态	创建时间	操作
<input type="checkbox"/>	节点内存使用率异常...	主机监控	workflow-activiti-service	重要	停用	2023-07-06 10:46:01	编辑 启动 停止 删除 告警事件历史
<input type="checkbox"/>	节点CPU使用率异常...	主机监控	workflow-activiti-service	重要	停用	2023-07-06 10:45:27	编辑 启动 停止 删除 告警事件历史
<input type="checkbox"/>	节点磁盘使用率异常...	主机监控	workflow-activiti-service	重要	已启用	2023-07-06 10:45:05	编辑 启动 停止 删除 告警事件历史
<input type="checkbox"/>	应用调用错误次数异...	应用调用统计	workflow-activiti-service	重要	已启用	2023-07-06 10:44:19	编辑 启动 停止 删除 告警事件历史
<input type="checkbox"/>	数据库调用错误异常...	数据库指标	workflow-activiti-service	重要	已启用	2023-07-06 10:43:23	编辑 启动 停止 删除 告警事件历史

10条/页 共 5 条 < 1 >

展示当前租户下所有告警规则信息，支持基础的增删改查、起停、查看告警事件历史操作。

- (1) 创建/编辑告警规则

编辑告警规则

* 告警名称: 节点磁盘使用率异常告警act 14/50

* 告警分组: 主机监控

* 告警指标: 节点磁盘使用率

* 告警条件: 当节点磁盘使用率 大于 80 %时, 满足告警条件。

* 筛选条件: 节点机IP 任意

* 告警内容: 应用(名称: {{ carms_obj_name }}, ip地址: {{ ip }})节点{{ path }}磁盘使用率过高, 当前值为{{ \$value }}%。

* 持续时间: 当告警条件满足时, 直接产生告警事件 当告警条件持续满足 0 分钟时, 才产生告警事件

* 告警等级: 重要

通知策略: 通用通知策略

* 通知频率: 10分钟

[高级设置 >](#)

- **告警名称:** 您可自定义告警名称
- **应用:** 展示应用列表中, 所有已接入应用, 支持多选
 - 新增应用自动在此告警规则中追加: 开启后, 新增应用则自动使用该告警规则
- **告警详情:** 包含告警分组、告警指标、告警条件、筛选条件、告警内容

告警分组	告警指标	告警条件
主机监控	节点 cpu 使用率	该指标大于、大于等于、小于、小于等于、等于某个百分比时生效
	节点内存使用率	
	节点磁盘使用率	
	JVM 实例数	
应用调用统计	调用错误次数	某个时间段内, 该指标大于、大于等于、小于、小于
	调用错误率	

	调用次数	等于、等于某个数值/百分比时生效
	调用平均响应时间	
HTTP 状态码异常	状态码 5xx 次数	某个时间段内，该指标大于、大于等于、小于、小于等于、等于某个数值/百分比时生效
	状态码 4xx 次数	
异常接口调用	调用次数	等于、等于某个数值/百分比时生效
	调用响应时间	
应用调用类型统计	应用提供服务调用总次数	某个时间段内，该指标大于、大于等于、小于、小于等于、等于某个数值/百分比时生效
	应用提供服务调用错误率	
	应用依赖服务调用错误率	
	应用依赖服务调用总次数	
	应用提供服务调用错误次数	
	应用依赖服务调用错误次数	
	应用提供服务调用响应时间	
	应用依赖服务调用响应时间	
数据库指标	数据库调用次数	某个时间段内，该指标大于、大于等于、小于、小于等于、等于某个数值/百分比时生效
	数据库调用响应时间	
	数据库调用错误次数	

JVM 监控	JVM_FullGC 次数瞬时值	该指标大于、大于等于、小于、小于等于、等于某个百分比时生效
	JVM_FullGC 耗时瞬时值	
	JVM_YoungGC 次数瞬时值	
	JVM_YoungGC 耗时瞬时值	
	JVM 堆内使用内存量	
	JVM 非堆使用内存量	
	JVM 死锁线程数	
	JVM 阻塞线程数	
	JVM 等待线程数	
	JVM 可运行线程数	
	JVM 线程总数	

- **持续时间**：支持设置延迟告警
- **告警等级**：一般、次要、重要、等级
- **通知策略**：可以选择通知策略菜单里设置的策略
- **标签**：自定义标签，用于筛选

(2) 启动/停止

对于暂时不用的告警策略，您可以操作停止

(3) 告警事件历史

点击跳转告警事件历史菜单，显示该告警规则触发的实际告警记录

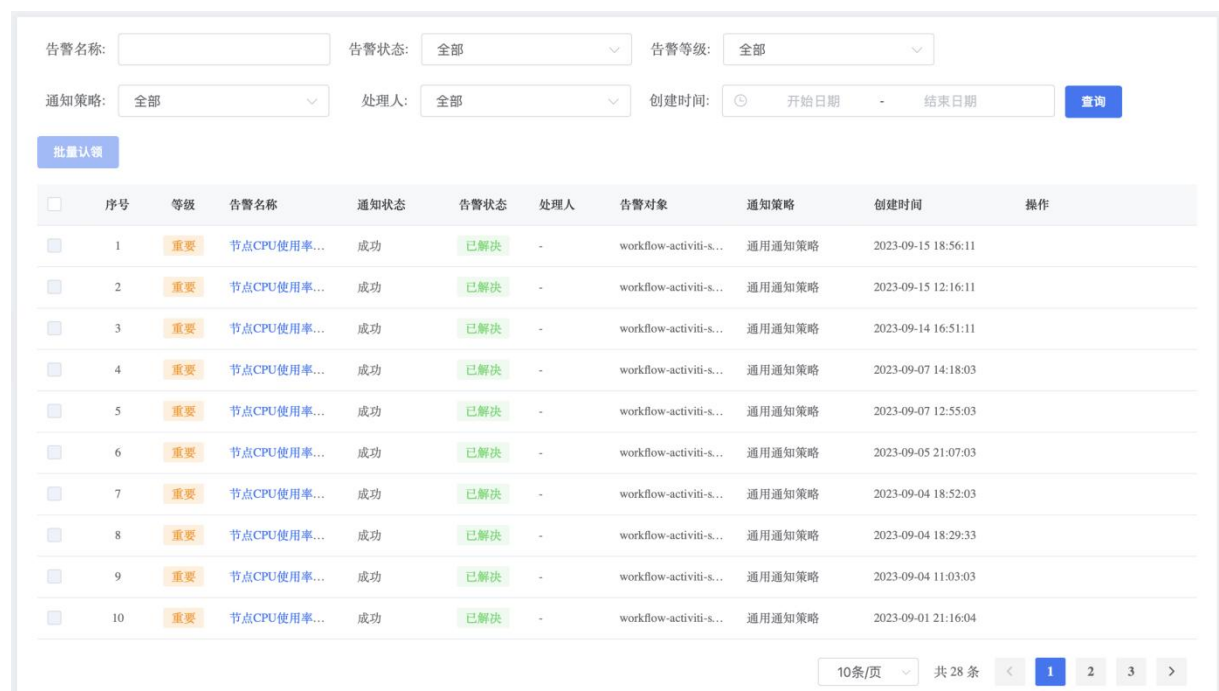
4.5.2、告警发送历史

展示当前租户下所有发送的告警历史，支持筛选和对告警信息进行操作。

功能入口

- (1) 选择目标资源池，并登录 APM 组件控制台。
- (2) 在左侧导航栏中选择「告警管理」-「告警发送历史」。

功能说明



告警名称:	告警状态:	告警等级:	通知策略:	处理人:	创建时间:	查询			
<input type="text"/>	全部	全部	全部	全部	<input type="text"/> 开始日期 - <input type="text"/> 结束日期	<input type="button" value="查询"/>			
<input type="button" value="批量认领"/>									
序号	等级	告警名称	通知状态	告警状态	处理人	告警对象	通知策略	创建时间	操作
1	重要	节点CPU使用率...	成功	已解决	-	workflow-activiti-s...	通用通知策略	2023-09-15 18:56:11	
2	重要	节点CPU使用率...	成功	已解决	-	workflow-activiti-s...	通用通知策略	2023-09-15 12:16:11	
3	重要	节点CPU使用率...	成功	已解决	-	workflow-activiti-s...	通用通知策略	2023-09-14 16:51:11	
4	重要	节点CPU使用率...	成功	已解决	-	workflow-activiti-s...	通用通知策略	2023-09-07 14:18:03	
5	重要	节点CPU使用率...	成功	已解决	-	workflow-activiti-s...	通用通知策略	2023-09-07 12:55:03	
6	重要	节点CPU使用率...	成功	已解决	-	workflow-activiti-s...	通用通知策略	2023-09-05 21:07:03	
7	重要	节点CPU使用率...	成功	已解决	-	workflow-activiti-s...	通用通知策略	2023-09-04 18:52:03	
8	重要	节点CPU使用率...	成功	已解决	-	workflow-activiti-s...	通用通知策略	2023-09-04 18:29:33	
9	重要	节点CPU使用率...	成功	已解决	-	workflow-activiti-s...	通用通知策略	2023-09-04 11:03:03	
10	重要	节点CPU使用率...	成功	已解决	-	workflow-activiti-s...	通用通知策略	2023-09-01 21:16:04	

10条/页 共 28 条 < 1 2 3 >

展示当前租户下所有告警发送信息。

- (1) 支持对告警名称、告警状态、告警等级、通知策略和创建时间进行筛选。

- **告警名称:** 显示告警规则名称

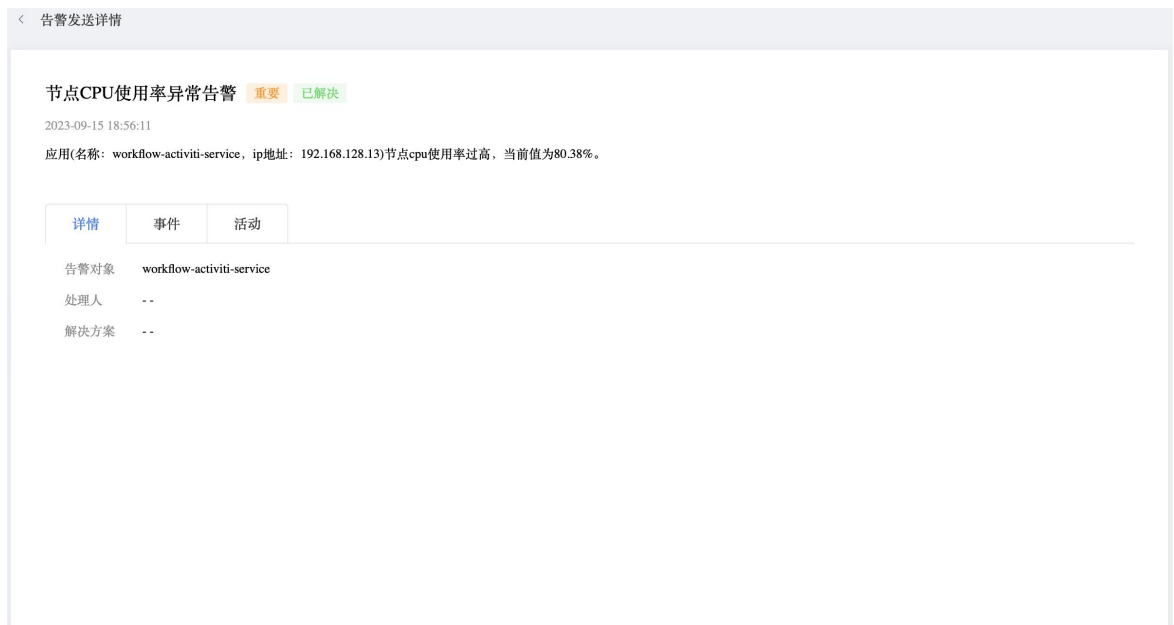
- **告警状态**：显示事件当前状态是待认领、已解决、处理中
- **告警等级**：显示告警的重要层级分布是一般、次要、重要、紧急
- **通知策略**：告警对应的通知策略
- **处理人**：告警的最新解决/认领人
- **创建时间**：告警产生的时间

(2) 支持认领、解决、指定处理人操作

- **认领**：当告警处于待认领状态时，可主动领取当前告警
- **解决**：当告警处于待认领/处理中状态时，点击解决可更新告警状态为已解决
- **指定处理人**：指定他人处理该告警

(3) 支持查看告警详情

- **详情**：显示告警发送的基础信息如告警对象、处理人、解决方案。



- **事件**：显示触发告警的事件名称、状态和触发时间，点击可查看详情。



- **活动：**显示包括认领、取消认领、指派处理人、告警关闭等在内的各种活动信息，支持筛选。



4.5.3、告警事件历史

展示当前租户下所有告警事件历史，支持筛选及查看详情。

功能入口

- (1) 选择目标资源池，并登录 APM 组件控制台。
- (2) 在左侧导航栏中选择「告警管理」-「告警事件历史」。

功能说明

告警事件

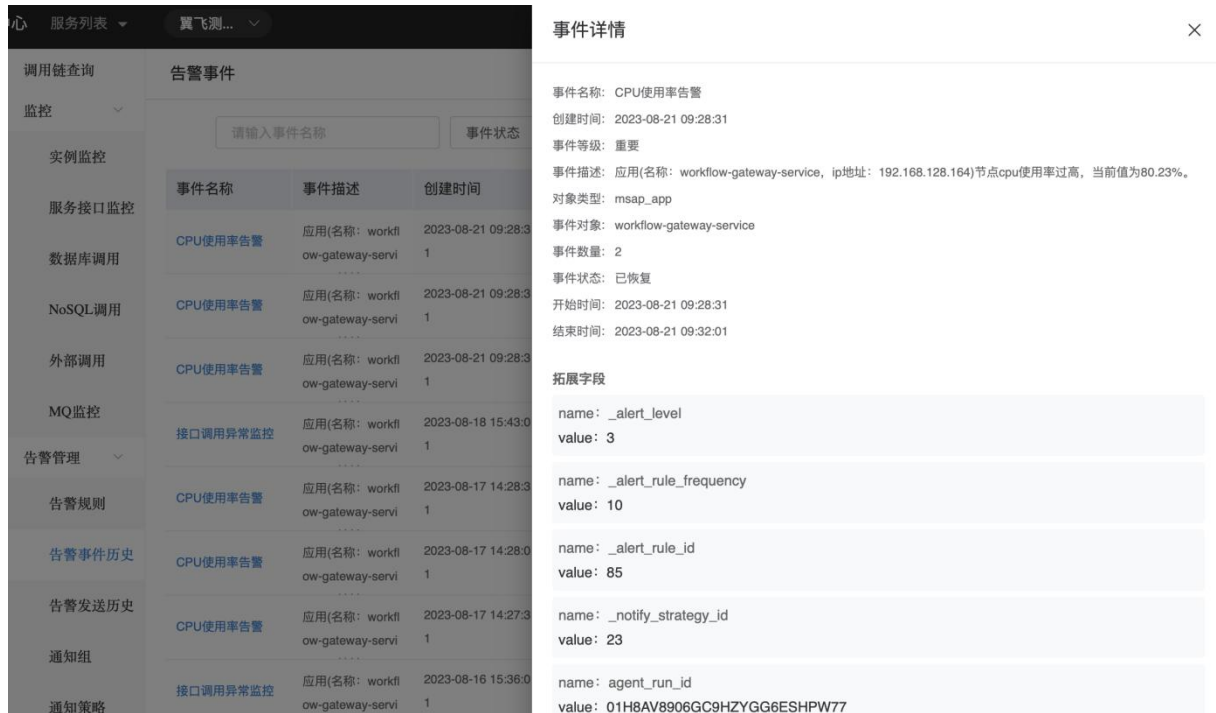
<input type="text" value="请输入事件名称"/>	<input type="text" value="事件状态"/>	<input type="text" value="请输入事件对象"/>	<input type="text" value="请输入对象类型"/>	<input type="button" value="查询"/>				
事件名称	事件描述	创建时间	事件数量	事件状态	关联告警	事件对象	对象类型	通知策略
CPU使用率告警	应用(名称: workflow-gateway-service)	2023-08-21 09:28:31	2	已恢复	CPU使用率告警	workflow-gateway-service	msap_app	通用通知策略
CPU使用率告警	应用(名称: workflow-gateway-service)	2023-08-21 09:28:31	2	已恢复	CPU使用率告警	workflow-gateway-service	msap_app	通用通知策略
CPU使用率告警	应用(名称: workflow-gateway-service)	2023-08-21 09:28:31	2	已恢复	CPU使用率告警	workflow-gateway-service	msap_app	通用通知策略
接口调用异常监控	应用(名称: workflow-gateway-service)	2023-08-18 15:43:01	2	已恢复	--	workflow-gateway-service	msap_app	通用通知策略
CPU使用率告警	应用(名称: workflow-gateway-service)	2023-08-17 14:28:31	2	已恢复	--	workflow-gateway-service	msap_app	通用通知策略
CPU使用率告警	应用(名称: workflow-gateway-service)	2023-08-17 14:28:01	2	已恢复	--	workflow-gateway-service	msap_app	通用通知策略

展示当前租户下所有告警事件信息。

(1) 支持对事件名称、事件状态、事件对象和对象类型进行筛选。

- **事件名称**: 显示告警规则名称
- **事件状态**: 显示事件当前状态是告警中、已恢复、静默
- **事件对象**: 监控对象, 比如应用名称、集群名称等
- **对象类型**: 告警事件对象的类型

(2) 支持查看事件详情



The screenshot shows a monitoring dashboard with a sidebar on the left and a main content area on the right. The sidebar includes sections for '调用链查询', '告警事件', '监控', '实例监控', '服务接口监控', '数据库调用', 'NoSQL调用', '外部调用', 'MQ监控', '告警管理', '告警规则', '告警事件历史', '告警发送历史', '通知组', and '通知策略'. The main content area is titled '事件详情' and displays the following information:

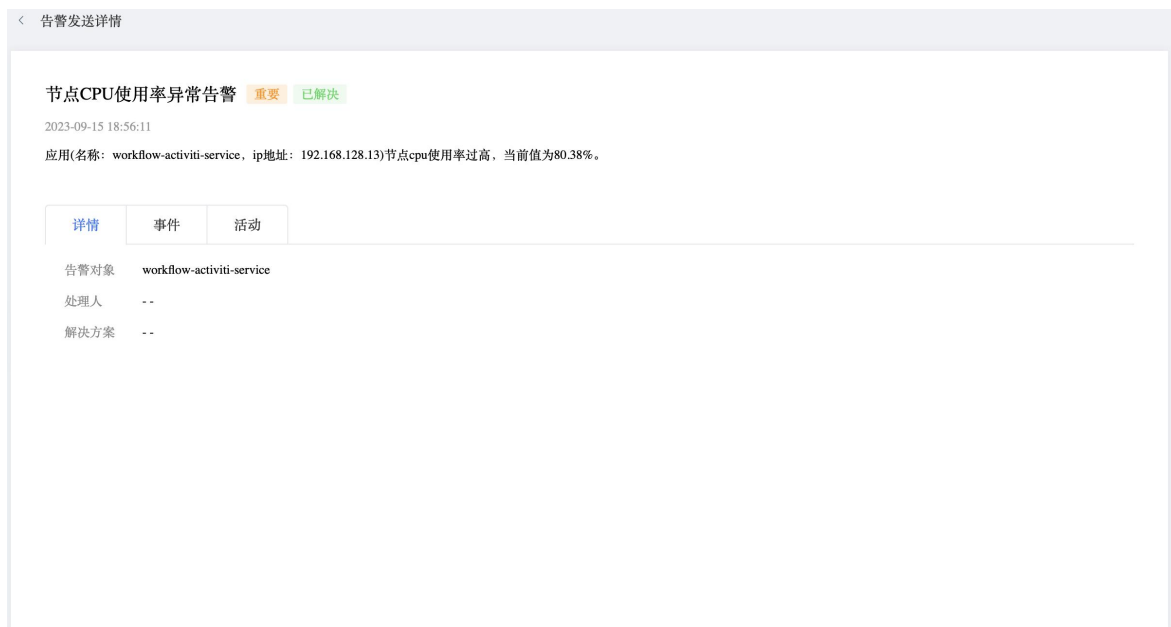
- 事件名称: CPU使用率告警
- 创建时间: 2023-08-21 09:28:31
- 事件等级: 重要
- 事件描述: 应用(名称: workflow-gateway-service, ip地址: 192.168.128.164)节点cpu使用率过高, 当前值为80.23%。
- 对象类型: msap_app
- 事件对象: workflow-gateway-service
- 事件数量: 2
- 事件状态: 已恢复
- 开始时间: 2023-08-21 09:28:31
- 结束时间: 2023-08-21 09:32:01

Below the event details, there is a '拓展字段' section with the following key-value pairs:

- name: _alert_level, value: 3
- name: _alert_rule_frequency, value: 10
- name: _alert_rule_id, value: 85
- name: _notify_strategy_id, value: 23
- name: agent_run_id, value: 01H8AV8906GC9HZYGG6ESHWP77

(3) 支持查看告警详情

- **详情**: 显示告警发送的基础信息如告警对象、处理人、解决方案。



The screenshot shows the '告警发送详情' (Alert Send Details) page. The main content area displays the following information:

- 节点CPU使用率异常告警 重要 已解决
- 2023-09-15 18:56:11
- 应用(名称: workflow-activiti-service, ip地址: 192.168.128.13)节点cpu使用率过高, 当前值为80.38%。

Below the alert details, there is a tabbed interface with three tabs: '详情' (selected), '事件', and '活动'. The '详情' tab shows the following information:

- 告警对象: workflow-activiti-service
- 处理人: --
- 解决方案: --

- **事件**: 显示触发告警的事件名称、状态和触发时间，点击可查看详情。



- **活动：**显示包括认领、取消认领、指派处理人、告警关闭等在内的各种活动信息，支持筛选。



4.5.4、通知对象

告警支持通过短信、邮箱、翼连等方式将告警信息通知给对应接收人，我们可以在通知组中设置接收人信息。

功能入口

- (1) 选择目标资源池，并登录 APM 组件控制台。
- (2) 在左侧导航栏中选择「告警管理」-「通知组」。

功能说明

通知对象

联系人 翼连 WebHook集成

新建联系人组 新建联系人 批量删除 数据脱敏

输入联系人组搜索

所有联系人组

<input type="checkbox"/>	姓名	手机号	Email	所属联系人组	操作
<input type="checkbox"/>	邵金斌	173****0393	@chinatelecom.cn	--	编辑 删除

10条/页 共 1 条 < 1 >

(1) 联系人（短信/邮箱）

如需要通过短信/邮箱进行告警，可在此处维护联系人信息。支持对联系人信息进行增删改查，需要录入基础信息。

新建联系人 ×

* 姓名 0/20

手机号码

邮箱

联系人组

同时也支持对联系人进行分组，对联系人组进行增删改查。

(2) 翼连

支持增删改查翼连群信息，将某个翼连群作为一个“联系人组”。

联系人	翼连	WebHook集成			
<input type="checkbox"/>	名称	翼连群号	通知策略	创建时间	操作
<input type="checkbox"/>	翼飞运维告警群	M1*****E9	通用通知策略	2023-07-04 15:01:49	编辑 删除

10条/页 共 1 条 < 1 >

新建翼连群

* 名称 0/50

* 群号 0/50

取消

确定

(3) WebHook 集成

支持以 WebHook 的方式对第三方通知对象（钉钉、企业微信、飞书等）发送告警信息。支持增删改查 WebHook 信息。

联系人	翼连	WebHook集成			
<input type="checkbox"/>	名称	地址	通知策略	创建时间	操作
暂无数据					

10条/页 共 0 条 < 1 >

新建WebHook

* 名称 0/50

* webhook调用地址

* 渲染方式 无 模板渲染

取消

确定

4.5.5、通知策略

创建告警通知策略，当告警触发时，只要不符合静默策略，就会依照通知策略在对应渠道发送告警信息给对应的通知对象。

功能入口

- (1) 选择目标资源池，并登录 APM 组件控制台。
- (2) 在左侧导航栏中选择「告警管理」-「通知策略」。

功能说明



支持增删改查告警通知策略信息

- 通知对象: 可从通知组进行选择。

当告警生成时

* 通知对象

+ 添加通知对象

- 通知模板：可跟进不同通知渠道（邮件、短信、翼连）设置不同的通知模板。

通知模板

邮件	短信	翼连
* 告警模板	<pre>监控告警{{{alerts length}}}
 告警规则: {{{commonLabels.alertname}}
 告警级别: {{{alertLevelDesc}}
 {% for alert in alerts %}{% if loop.index le 5 %} 【告警{{{loop.index}}}】
 告警区域: {{{alert.labels.region_code}}
 告警时间: {{{alert.startsAt}}
 告警内容: {{{alert.annotations.description}}
 {% endif %}{% endfor %} {% if alerts length gt 5 %}最多展示前5个。{% endif %}</pre>	
* 恢复模板	<pre>监控告警恢复{{{alerts length}}}
 告警规则: {{{commonLabels.alertname}}
 告警级别: {{{alertLevelDesc}}
 {% for alert in alerts %}{% if loop.index le 5 %} 【告警{{{loop.index}}}】
 告警区域: {{{alert.labels.region_code}}
 告警时间: {{{alert.startsAt}}
 恢复时间: {{{alert.endsAt}}
 告警内容: {{{alert.annotations.description}}
 {% endif %}{% endfor %}</pre>	

- 通知时段：可以设置通知时段，默认是告警触发时通知。

通知时段 ~

不配置默认全天通知时段

4.5.6、静默策略

如果满足静默策略则不会触发告警通知，您可以通过静默策略对告警进行收敛，避免同一时间出现大量重复告警或关联告警。

功能入口

- (1) 选择目标资源池，并登录 APM 组件控制台。
- (2) 在左侧导航栏中选择「告警管理」-「静默策略」。

功能说明

静默策略

<input type="checkbox"/>	静默策略名称	创建时间	策略生效状态	操作
暂无数据				

10条/页 共0条 < 1 >

支持增删改查静默策略

< 新建静默策略

* 静默事件匹配规则

事件规则

条件列表

条件1 事件字段名 请选择 事件字段值 删除

+ 添加条件

+ 添加规则

* 静默规则生效时间

是否限制时间: 是 否

* 静默时段: 起始时间 ~ 结束时间

确认 取消

- 事件规则：支持通过且或关系组合创建事件规则，使得在满足当前事件规则时，触发静默策略。
- 静默时段：设置静默规则的生效时间段。

五、最佳实践

5.1、使用调用链采样策略

本文介绍 APM 支持的调用链采样模式，帮助您以较低成本获取想要的调用链数据。

对于绝大多数的分布式系统，不是每一条调用链都值得被可观测平台记录，因为其中包含大量重复的、低关注度的信息。因此需要引入采样技术降低整体可观测成本，并过滤对用户没有帮助的噪音。

调用链采样的基本原则是优先记录您最关心、最有可能访问的调用链。APM 提供固定采样率这种采样模式。

固定采样率

固定比例采样就是根据 TraceId 顺序号记录一定比例的调用链数据。例如，固定比例为 10%，则每 10 条调用链数据记录 1 条。固定比例采样不会导致调用链数据本身不完整，要么保留整条链路数据，要么丢弃整条链路数据。

设置固定比例采样的操作步骤如下：

1. 登录 APM 控制台，在左侧导航栏选择**应用监控 > 应用列表**。
2. 在**应用列表**页面单击目标应用名称。
3. 在左侧导航栏中单击**应用设置**，并在右侧页面单击调用链采集设置页签。

4. 在采样率设置区域设置采样率。在采样率设置字段输入百分比的数字部分，例如输入 1 代表采样 1%，如下图所示。



调用链采样设置

是否采集调用链

说明：关闭此配置，调用链功能将关闭，请您谨慎操作！

* 采样率设置(%)

1

说明：

- 1、采样率默认为1。
- 2、调大采样率可能会消耗额外的系统资源，有最大每秒采集条数限制。
- 3、每秒采集100条的情况下，开销在300m内存左右。如需调整每秒最大采集数，可修改限流阈值。

5.2、诊断服务端报错问题

诊断服务端报错问题

(1) 问题描述

网页抛错，尤其是 5xx 错误是互联网应用最常见的问题之一。5xx 错误通常发生于服务端。服务端是业务逻辑最复杂，也是整条网络请求链路中最容易出错、出了错之后最难诊断原因的地方。运维工程师或研发工程师往往需要登录机器查看日志来定位问题。

对于逻辑不太复杂、上线时间不长的应用来说，登录机器查看日志的方式能够解决大部分网站抛错的问题。但在以下场景中，传统的问题诊断方式往往没有用武之地。

1. 在一个分布式应用集群中，需知道某一类错误的发生时间和频率。
2. 某系统已运行了很长时间，但是不想关心遗留的异常，只想知道今天和昨天相比、发布后和发布前相比多了哪些异常。
3. 查看一个异常对应的 Web 请求和相关参数。
4. 客服人员提供了一个用户下单失败的订单号，分析该用户下单失败的原因。

(2) 解决方案

为应用安装 APM 探针后，即可在不改动应用代码的情况下，利用应用性能监控 APM 的异常自动捕捉、收集、统计和溯源等能力，全面掌握应用的各种错误信息。

步骤一：安装 APM 探针

为应用安装 APM 探针后，才能对应用进行全方位监控。请根据实际需求选择一种方式来安装探针。

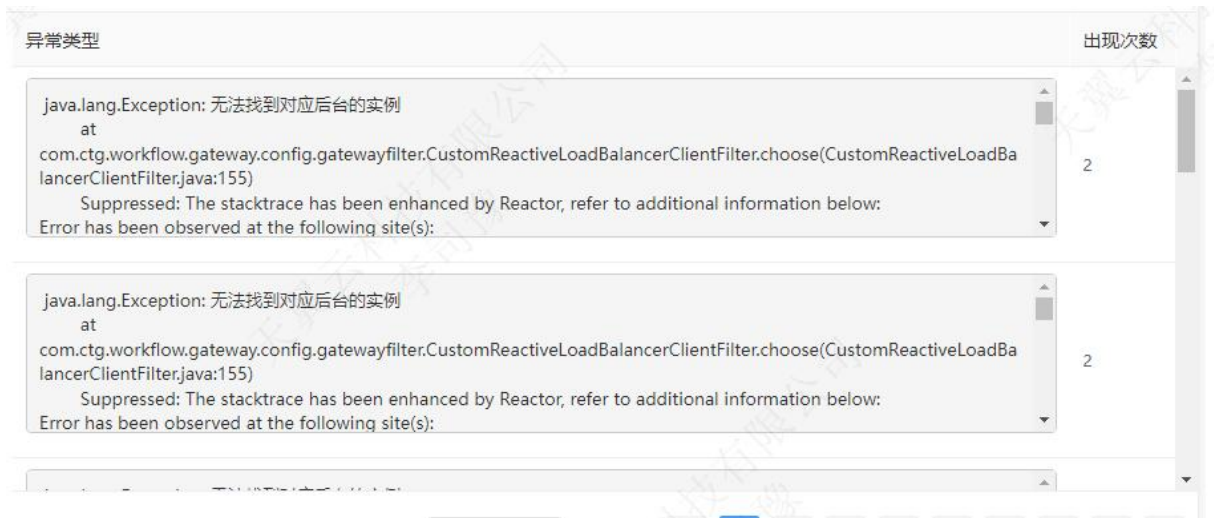
步骤二：查看关于应用异常的统计信息

为应用安装 APM 探针后，APM 会收集和展示选定时间内应用的总请求量、平均响应时间、错误数、实时实例数、FullGC 次数、慢 SQL 次数、异常次数和慢调用次数，以及这些指标和上一天的环比、上周的同比升降幅度。请按以下步骤查看应用异常的统计信息。

1. 在**应用总览**页面的概览页签下方，查看异常的总数、周同比和日同比数据，如下图所示。



2. 滑动页面至概览页签底部的统计分析区域的异常类型，查看各类型异常出现的次数，如下图所示。



3. 在左侧导航栏，点击监控 > 实例监控，然后在页面右侧单击异常分析页签，查看异常统计图、错误数、异常堆栈等，如下图所示。



步骤三：诊断异常出现的原因

掌握应用异常的统计信息还不足以诊断异常出现的原因。虽然日志中异常堆栈包含调用的代码片段，但并不包含这次调用的完整上下游信息和请求参数。

APM 探针采用了字节码增强技术，让您能够以很小的性能消耗捕获异常上下游的完整调用快照，进而找出导致异常出现的具体原因。

1. 在异常分析页签下，找到要诊断的异常类型，在其右侧操作列，单击调用链查询。调用链查询页签下显示与该异常类型相关的调用链路信息，如下图所示。



产生时间	接口名称	所属应用	耗时(ms)	状态	TraceId
2023-08-14 15:06:21.450	HTTP GET	workflow-gateway-service	17.125	●	1758c6289025c078e00716761e06fabd
2023-08-14 15:06:21.444	HTTP GET	workflow-gateway-service	20.429	●	1758c6289025c078e00716761e06fabd
2023-08-14 15:02:11.370	HTTP GET	workflow-gateway-service	23.455	●	ef78a51b29c8cb4b70b5a3dffb0ff8

2. 在调用链查询页签下，单击某个错误调用的 TraceId，如下图所示。



调用方法	操作	拓展信息	时间轴(ms)
HTTP GET	详情 查看日志		23.455
FilteringWebHandlehandle	详情 查看日志	java.lang.Exception: 无法找到对应, at com.ctc.workflow.exten... Suppressed: The stacktrace Error has been observed at the fu...	0.124

3. 在弹出的页面，查看异常的调用链路信息。

操作至此，您已发现了应用异常的原因，这将有效地帮助您进行下一步的代码优化工作。您还可以返回调用链查询页签，查看列表中其他异常，逐一解决。

5.3、诊断应用卡顿问题

定位、排查应用卡顿问题的原因有诸多难点。针对这类问题，APM 提供线程剖析、调用链路诊断、接口监控等一套解决方案，帮助您快速准确定位应用中所有慢调用，进而解决应用卡顿问题。

问题分析

网站卡顿、页面加载过慢是互联网应用最常见的问题之一。排查、解决网站卡顿、页面加载过慢等问题过程复杂，耗时较长，原因如下：

- 应用链路太长从前端页面到后台网关，从 Web 应用服务器到后台数据库，任何一个环节出现故障都有可能导致整体卡顿。
 - 采用微服务架构的应用，链路更加复杂，而且不同组件可能由不同的团队和人员维护，加剧了问题排查的难度。
- 日志不全或质量欠佳应用日志是排查线上问题的主要方法，但出现问题的位置往往无法预期，而且“慢”通常是偶发现象，要真正找到“慢”的原因，需要在每个可能出现问题的地方打印日志，记录每一次调用，但是成本太高。
- 监控不足业务发展过快、应用快速迭代导致应用频繁修改接口、增加依赖等情况，进而导致代码质量恶化。应用需要一个完善的监控体系来自动监控应用的每一个接口，自动记录出现问题的调用。

解决方案

为应用安装 APM 探针后，即可在不改动应用代码的情况下，使用 APM 应用监控的线程剖析、调用链路诊断、接口监控等功能，全方位监控应用中所有慢调用。

步骤一：安装 APM 探针

为应用安装 APM 探针后，才能对应用进行全方位监控。请根据实际需求选择一种方式来安装探针。

步骤二：查看慢 SQL 的统计信息

为应用安装 APM 探针后，APM 会收集和展示选定时间内应用的总请求量、平均响应时间、错误数、实时实例数、Full GC 次数、慢 SQL 次数、异常次数和慢调用次数，以及这些指标的周同比和日同比。请按以下步骤查看慢 SQL 的统计信息。

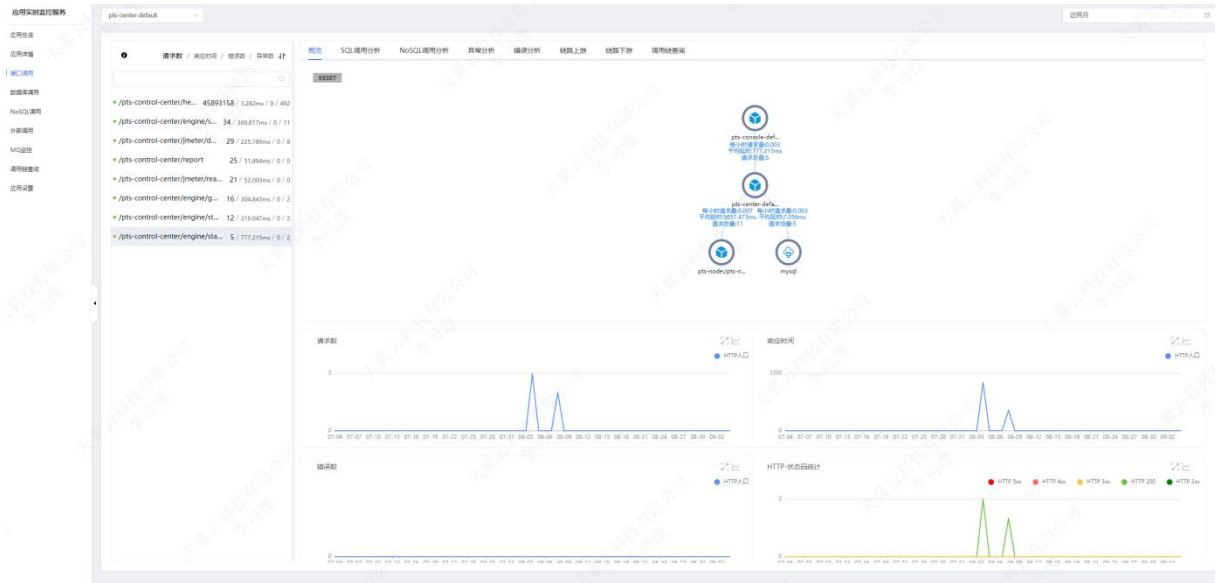
1. 登录 AMRS 控制台，在左侧导航栏选择**应用监控 > 应用列表**。
2. 在**应用列表**页面顶部单击目标应用名称。
3. 在**应用总览**页面的**概览**页签下，查看慢 SQL 的总数、周同比和日环比数据。



步骤三：发现并锁定慢调用

APM 在**接口调用**页面展示了被监控的应用提供的所有接口以及这个接口的调用次数和耗时，慢接口会被标注出来，帮助您发现和锁定慢接口。

1. 在左侧导航栏，单击**接口调用**。
2. 在**接口调用**页面的左侧，单击调用次数最多的慢接口，在右侧查看慢接口的详细信息。



步骤四：查看并锁定问题代码

锁定慢接口后，需要找到问题代码来解决问题。快照是对一次调用的完整链路调用的完整记录，包括每一次调用所经过的代码及耗时，可以精准定位问题代码。

1. 在接口调用页面右侧，单击调用链查询页签。调用链查询页签下显示该接口的所有调用链。



2. 在调用链查询页签下，单击某个调用链路的 TraceId。

详情

应用名称: _____ 接口名称: _____
 IP地址: _____ TraceId: _____
 产生时间: _____ 耗时(ms): _____

调用栈

调用方法	操作	拓展信息	时间轴(ms)
▼	详情		0
▼ /pts-control-center/engine/startSceneDebug	详情		430.474
▼ EngineController.startSceneDebug	详情	feign.FeignException\$ServiceUnava: at feign.FeignException.s: at feign.FeignException.es: at feign.FeignException.es	419.156
HikariDataSource.getConnection	详情		0.683
SELECT pts	详情		1.101
HTTP POST	详情		6.828

3. 在弹出的页面，查看异常的调用链路信息，将鼠标悬停在拓展信息部分，可以获得异常的具体信息。

详情

应用名称: _____ 接口名称: _____
 IP地址: _____ TraceId: _____
 产生时间: _____ 耗时(ms): _____

调用栈

调用方法

```

feign.FeignException$ServiceUnavailable: [503 Service Unavailable] during [POST] to [http://gateway-paas/api/openApi/]
at feign.FeignException.serverResponseStatus(FeignException.java:256)
at feign.FeignException.errorStatus(FeignException.java:197)
at feign.FeignException.errorStatus(FeignException.java:185)
at feign.codec.ErrorDecoder$Default.decode(ErrorDecoder.java:92)
at feign.AsyncResponseHandler.handleResponse(AsyncResponseHandler.java:96)
at feign.SynchronousMethodHandler.executeAndDecode(SynchronousMethodHandler.java:138)
at feign.SynchronousMethodHandler.invoke(SynchronousMethodHandler.java:89)
at feign.ReflectiveFeign$FeignInvocationHandler.invoke(ReflectiveFeign.java:100)
at com.sun.proxy.$Proxy175.nodeRunDebug(Unknown Source)
at com.ctyun.pts.control.service.EngineService.startSceneDebug(EngineService.java:254)
at com.ctyun.pts.control.service.EngineService$$FastClassBySpringCGLIB$$6424e690.invoke(<generated>)
at org.springframework.cglib.proxy.MethodProxy.invoke(MethodProxy.java:218)
at org.springframework.aop.framework.CglibAopProxy.invokeMethod(CglibAopProxy.java:386)
at org.springframework.aop.framework.CglibAopProxy.access$000(CglibAopProxy.java:85)
at org.springframework.aop.framework.CglibAopProxy$DynamicAdvisedInterceptor.intercept(CglibAopProxy.java:704)
at com.ctyun.pts.control.service.EngineService$$EnhancerBySpringCGLIB$$17ca065a.startSceneDebug(<generated>)
at com.ctyun.pts.control.controller.EngineController.startSceneDebug(EngineController.java:58)
at java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
at java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
at java.base/jdk.internal.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
at java.base/java.lang.reflect.Method.invoke(Method.java:566)
at org.springframework.web.method.support.InvocableHandlerMethod.doInvoke(InvocableHandlerMethod.java:205)
at org.springframework.web.method.support.InvocableHandlerMethod.invokeForRequest(InvocableHandlerMethod.java)
at org.springframework.web.servlet.mvc.method.annotation.ServletInvocableHandlerMethod.invokeAndHandle(Servle
    
```

拓展信息

拓展信息	时间轴(ms)
	0
	430.474
feign.FeignException\$ServiceUnava: at feign.FeignException.s: at feign.FeignException.es: at feign.FeignException.es	419.156
	0.683
	1.101
	6.828

操作至此，您已发现了系统中的某个慢调用的原因，这将有效地帮助您进行下一步的代码优化工作。您还可以返回接口调用页面，查看列表中其他慢调用，逐一解决。

后续操作

为避免在出现问题后被动诊断错误原因,您还可以使用 APM 的告警功能针对一个接口或全部接口创建告警,即可在出现问题的第一时间向运维团队发送通知。

5.4、业务日志关联调用链的 Traceld 信息

您可以在应用的业务日志中关联调用链的 Traceld 信息,从而在应用出现问题时,能够通过调用链的 Traceld 快速关联到业务日志,及时定位分析、解决问题。

背景信息

APM 在业务日志中关联调用链 Traceld 的功能基于 MDC

(Mapped Diagnostic Context) 机制实现,支持主流的 Log4j、Log4j2 和 Logback 日志框架。

开启关联业务日志与 Traceld 开关

1. 在左侧导航栏选择**应用监控 > 应用列表**。
2. 在**应用列表**页面顶部选择目标地域,然后单击目标应用名称。
3. 在左侧导航栏中单击**应用设置**,找到日志开启设置栏目。
4. 打开**关联业务日志与 Traceld**的开关,选择日志项目,日志单元,日志规则。然后保存

如下图:

日志开启设置

关联业务日志与TraceId

开启后，可在调用链详情中查看对应的日志信息。支持Log4/Log4j2/Logback日志组件。

* 日志项目

hurm-test

* 日志单元

hurm-test-unit

* 日志规则

请选择

5.5、通过调用链路和日志分析定位业务异常问题

定位业务异常问题难度大、效率低，一直是应用性能监控的性能瓶颈。应用性能监控通过结合调用链路和日志分析，可以快速、准确地定位业务异常问题，提升微服务框架下的开发诊断效率。

背景信息

- 在使用调用链路和日志分析定位业务异常问题前，需要先了解 Metrics、Tracing 和 Logging 三个概念。Metrics：应用的关键性能指标，如应用提供服务请求量、应用提供服务平均响应时间、应用依赖服务请求量等。
- Tracing：调用链路，应用的任何接口调用、请求响应等动作都会绑定到完整的链路。

- Logging: 业务日志，应用的任何接口调用、请求响应等动作都会输出完整的业务日志。

当应用出现业务异常问题时，应用指标统计图会出现明显波动，您可据此粗略地分析异常问题；通过完整的调用链路和业务日志分析，可以精准定位业务异常问题。

开启日志设置

开启日志设置的操作步骤如下：

1. 登录 APM 控制台，在左侧导航栏选择**应用监控 > 应用列表**。
2. 在**应用列表**页面单击目标应用名称。
3. 在左侧导航栏中单击**应用设置**，并在右侧页面单击**日志开启设置**页签。
4. 在**日志开启设置**区域开启**关联业务日志与 Traceld**，并设置日志项目、日志单元、日志规则，如下图所示。



日志开启设置

关联业务日志与Traceld

开启后，可在调用链详情中查看对应的日志信息。支持Log4j/Log4j2/Logback日志组件。

* 日志项目

请选择

* 日志单元

请选择

* 日志规则

请选择

从应用指标的角度排查业务异常问题

在左侧导航栏单击**应用总览**，在顶部选择**概览**，然后在右上角选择或自定义设置目标时间段。

概览页面展示目标应用的关键指标，如**应用提供服务请求量**、**应用提供服务平均响应时间**、**应用依赖服务请求量**等。

1. 在调用链路列表面板选择**状态异常**的调用链路记录。



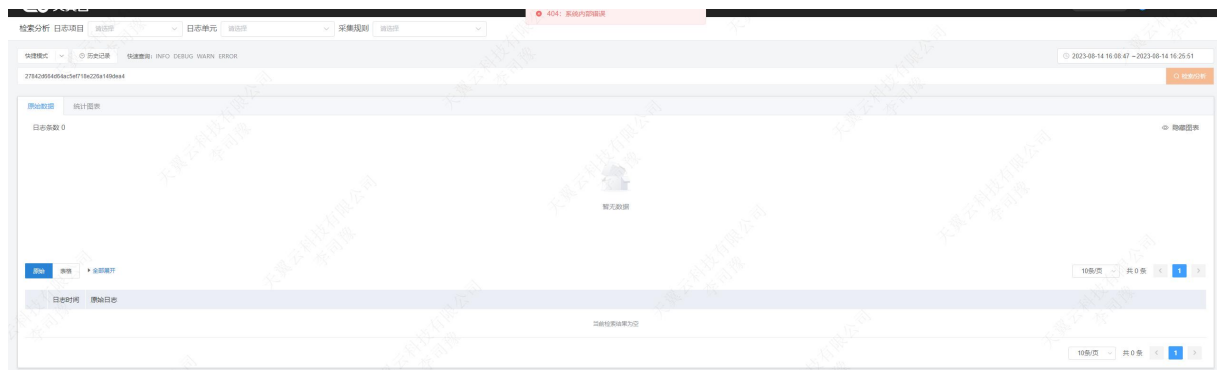
产生时间	接口名称	所属应用	耗时(ms)	状态	TraceId
2023-08-14 15:06:21.450	HTTP GET	workflow-gateway-service	17.125	●	1758c6289025c078e00716761e06fabd
2023-08-14 15:06:21.444	HTTP GET	workflow-gateway-service	20.429	●	1758c6289025c078e00716761e06fabd
2023-08-14 15:02:11.370	HTTP GET	workflow-gateway-service	23.455	●	ef78a51b29c8cbc4b70b5a3fdffb0ff8

2. 单击该调用链路记录 **TraceId** 列下的 **TraceId** 值。



调用方法	操作	拓展信息	时间轴(ms)
HTTP GET	详情 查看日志		0
FilteringWebHandlehandle	详情 查看日志	java.lang.Exception: 无法找到对应... at com.ctg.workflow.gatew... Suppressed: The stacktrac... Error has been observed at the fo...	7.388

3. 单击查看日志，即可查看日志并定位业务异常原因（目前返回 404）。



日志时间	日志内容
	404: 资源不存在

从接口调用的角度排查业务异常问题

1.登录 APM 控制台，在左侧导航栏选择**监控 > 实例监控**。

2.在**实例监控**页面顶部单击**调用链查询**页签。

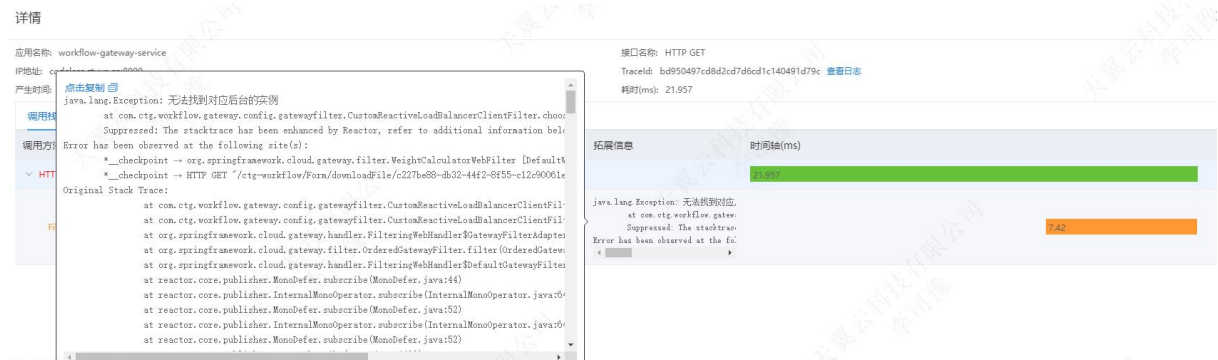


产生时间	接口名称	所属应用	耗时(ms)	状态	TraceId
2023-08-14 16:51:31.475	HTTP POST	workflow-gateway-service	41.582	●	0c3f21318945c19be955a8256627ed7a
2023-08-14 16:51:31.327	HTTP POST	workflow-gateway-service	47.531	●	092c4081c43e09dd306138a58fcbfce
2023-08-14 16:51:31.315	HTTP POST	workflow-gateway-service	57.3	●	4eaa405aa58d450c8a6960a5cdac7905
2023-08-14 16:51:31.308	HTTP POST	workflow-gateway-service	52.126	●	98e92a6171ac998687bc7e921875e20d
2023-08-14 16:51:30.372	HTTP GET	workflow-gateway-service	410.4	●	588ab7cf4722f6c30003e9301c9e2c6
2023-08-14 16:50:12.070	HTTP POST	workflow-gateway-service	1712.356	●	abc0f7de859957f40020600ec4d40b37
2023-08-14 16:49:26.551	HTTP GET	workflow-gateway-service	789.823	●	dc40e5b8e13f94f876fc3e882efa265
2023-08-14 16:49:25.388	HTTP GET	workflow-gateway-service	409.604	●	13574d6289581ac1f930b350a75ef445
2023-08-14 16:49:24.222	HTTP GET	workflow-gateway-service	21.957	●	bd990497cd8d2cd7d6cd1c140491d79c
2023-08-14 16:48:11.470	HTTP GET	workflow-gateway-service	15.652	●	ffe007968fb203cb5a7085ab26f143

3.在**调用链查询**页签选择**状态异常**的接口调用记录，异常状态显示为 ●。

4.在目标接口调用记录的 **TraceId** 列下单击 **TraceId** 的值。

5. 在链路详情信息页面查找错误信息，鼠标悬停在错误信息上可查看异常原因。



应用名称: workflow-gateway-service
IP地址: ...
产生时间: ...

接口名称: HTTP GET
TraceId: bd990497cd8d2cd7d6cd1c140491d79c
耗时(ms): 21.957

异常信息: java.lang.Exception: 无法找到对应后台的实例
at com.ctg.workflow.gateway.config.gatewayfilter.CustomReactiveLoadBalancerClientFilter.choose...
Error has been observed at the following site(s):

6.单击**查看日志**，即可查看日志并定位业务异常原因。

5.6、通过调用链路和日志分析定位业务异常问题

定位业务异常问题难度大、效率低，一直是应用性能监控的性能瓶颈。应用性能监控通过结合调用链路和日志分析，可以快速、准确地定位业务异常问题，提升微服务框架下的开发诊断效率。

背景信息

- 在使用调用链路和日志分析定位业务异常问题前，需要先了解 Metrics、Tracing 和 Logging 三个概念。Metrics：应用的关键性能指标，如应用提供服务请求量、应用提供服务平均响应时间、应用依赖服务请求量等。
- Tracing：调用链路，应用的任何接口调用、请求响应等动作都会绑定到完整的链路。
- Logging：业务日志，应用的任何接口调用、请求响应等动作都会输出完整的业务日志。

当应用出现业务异常问题时，应用指标统计图会出现明显波动，您可据此粗略地分析异常问题；通过完整的调用链路和业务日志分析，可以精准定位业务异常问题。

开启日志设置

开启日志设置的操作步骤如下：

1. 登录 APM 控制台，在左侧导航栏选择**应用监控 > 应用列表**。
2. 在**应用列表**页面单击目标应用名称。

3. 在左侧导航栏中单击**应用设置**，并在右侧页面单击**日志开启设置**页签。
4. 在**日志开启设置**区域开启**关联业务日志与 Traceld**，并设置**日志项目**、**日志单元**、**日志规则**，如下图所示。



日志开启设置

关联业务日志与Traceld

开启后，可在调用链详情中查看对应的日志信息。支持Log4j/Log4j2/Logback日志组件。

* 日志项目

请选择

* 日志单元

请选择

* 日志规则

请选择

从应用指标的角度排查业务异常问题

在左侧导航栏单击**应用总览**，在顶部选择**概览**，然后在右上角选择或自定义设置目标时间段。

概览页面展示目标应用的关键指标，如**应用提供服务请求量**、**应用提供服务平均响应时间**、**应用依赖服务请求量**等。

1. 在调用链路列表面板选择**状态异常**的调用链路记录。



产生时间	接口名称	所属应用	耗时(ms)	状态	Traceld
2023-08-14 15:06:21.450	HTTP GET	workflow-gateway-service	17.125	●	1758c6289025c078e00716761e06fabd
2023-08-14 15:06:21.444	HTTP GET	workflow-gateway-service	20.429	●	1758c6289025c078e00716761e06fabd
2023-08-14 15:02:11.370	HTTP GET	workflow-gateway-service	23.455	●	ef78a51b29c8c4b70b5a3fdff0f8

- 2.单击该调用链路记录 **Traceld** 列下的 **Traceld** 值。

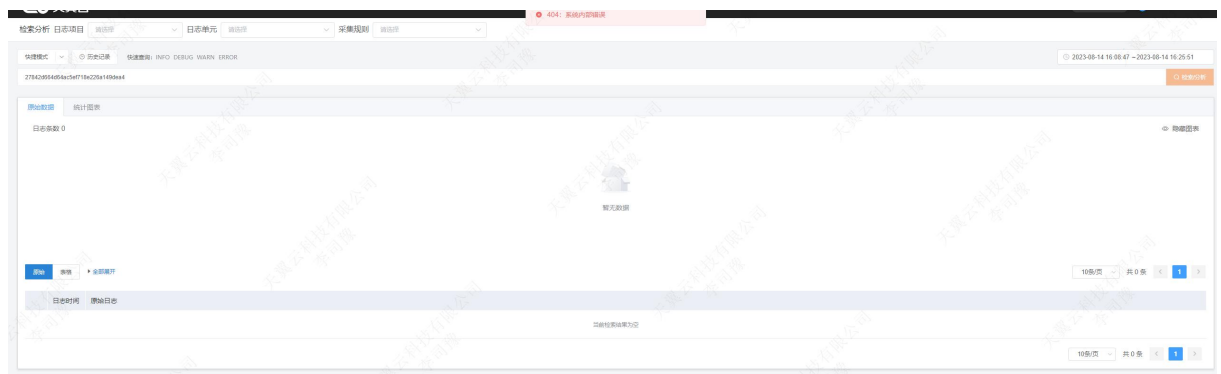
详情

应用名称:	接口名称:
IP地址:	TraceId: 查看日志
产生时间:	耗时(ms):

调用链

调用方法	操作	拓展信息	时间轴(ms)
HTTP GET	详情 查看日志		0
FilteringWebHandler.handle	详情 查看日志	java.lang.Exception: 无法找到对应, at com.ctg.workflow.gatew: Suppressed: The stacktrace Error has been observed at the f.c.	7.388

3.单击查看日志，即可查看日志并定位业务异常原因（目前返回 404）。



从接口调用的角度排查业务异常问题

1.登录 APM 控制台，在左侧导航栏选择监控 > 实例监控。

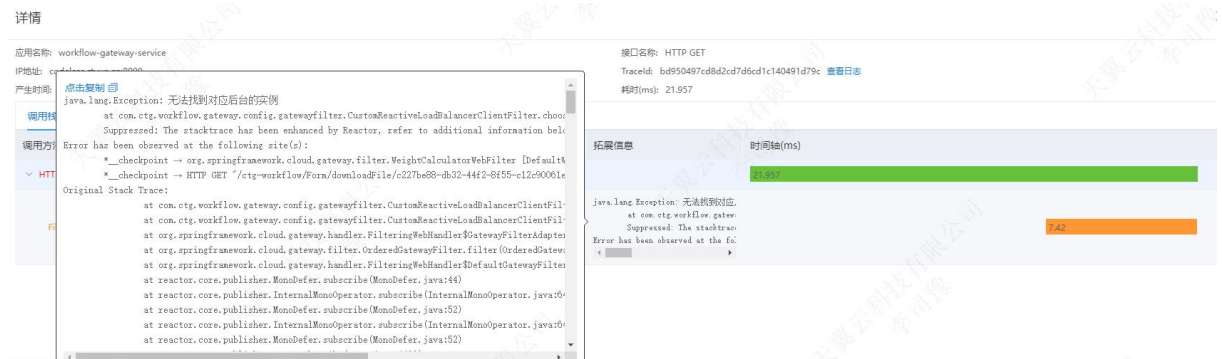
2.在实例监控页面顶部单击调用链查询页签。

产生时间	接口名称	所属应用	耗时(ms)	状态	TraceId
2023-08-14 16:51:31.475	HTTP POST	workflow-gateway-service	41.582	●	0c3f21318945c19be955a8256627e7a
2023-08-14 16:51:31.327	HTTP POST	workflow-gateway-service	47.531	●	092c4061c43e09dd306138a58fcbf9e
2023-08-14 16:51:31.315	HTTP POST	workflow-gateway-service	57.3	●	4eaa405aa58d450c8a6960a5cdac7905
2023-08-14 16:51:31.308	HTTP POST	workflow-gateway-service	52.126	●	98e92a6171ac698667bc7e921875e20d
2023-08-14 16:51:30.372	HTTP GET	workflow-gateway-service	410.4	●	588ab7c1f722f6c30003e95301c9e2c6
2023-08-14 16:50:12.070	HTTP POST	workflow-gateway-service	1712.356	●	abc0f7da89595740020600ec4d40b37
2023-08-14 16:49:26.551	HTTP GET	workflow-gateway-service	789.823	●	dc40e5b8e13f354f876fc3e882efa265
2023-08-14 16:49:25.388	HTTP GET	workflow-gateway-service	409.604	●	13574d6289581ac1f30b330e75fe4f5
2023-08-14 16:49:24.222	HTTP GET	workflow-gateway-service	21.957	●	bd950497cd8d2cd7d5cd1c140491d79c
2023-08-14 16:48:11.470	HTTP GET	workflow-gateway-service	15.652	●	ffe007968fbfb203cb5a7485ab26f143

3.在调用链查询页签选择状态异常的接口调用记录，异常状态显示为 ●。

4.在目标接口调用记录的 TraceId 列下单击 TraceId 的值。

5. 在链路详情信息页面查找错误信息，鼠标悬停在错误信息上可查看异常原因。



6. 单击查看日志，即可查看日志并定位业务异常原因。

5.7、使用 APM 监控异步任务

对于通过 Spring @Async 标签或者 Java Executors 实现的异步任务，APM 默认支持对其进行监控。若您的异步任务出现接口超时等异常，可以通过调用链路查看异步任务上下游以便及时处理潜在问题。

前置条件

该功能要求 APM 探针版本为公测版本 v1.1.1 及以上。

方式一：默认支持 Spring @Async 标签

APM 默认支持监控使用 Spring @Async 标签实现的异步任务。对于下列 Spring 框架中默认的 Executor 和 Task，APM 会自动完成增强：

- Executor:
 - org.springframework.scheduling.concurrent.ConcurrentTaskExecutor
 - org.springframework.core.task.SimpleAsyncTaskExecutor
 - org.springframework.scheduling.quartz.SimpleThreadPoolTaskExecutor

- org.springframework.core.task.support.TaskExecutorAdapter
- org.springframework.scheduling.concurrent.ThreadPoolTaskExecutor
- org.springframework.scheduling.concurrent.ThreadPoolTaskScheduler
- org.springframework.jca.work.WorkManagerTaskExecutor
- org.springframework.scheduling.commonj.WorkManagerTaskExecutor
- Task: org.springframework.aop.interceptor.AsyncExecutionInterceptor\$1
- org.springframework.aop.interceptor.AsyncExecutionInterceptor\$\$Lambda\$
\$

方式二：通过增强 Runnable 接口、Callable 接口和 Executor

当您没有使用 spring 框架时，上述场景无法满足需求时，也可以通过自动增强

Runnable 接口、Callable 接口和 Executor，在异步线程中完成调用链串联，实现异步任务监控。

1、增强 Runnable 接口和 Callable 接口

```
public class MyRunnable implements Runnable {  
  
    @Override  
  
    public void run() {  
  
        // 在这里定义需要在新线程中执行的任务逻辑  
  
        for (int i = 0; i < 5; i++) {
```

```
        System.out.println("Thread ID: " + Thread.currentThread().getId() + " - Cou  
nt: " + i);  
    }  
}  
  
public static void main(String[] args) {  
    // 创建一个 Runnable 实例  
  
    Runnable myRunnable = new MyRunnable();  
  
    // 创建线程并将 Runnable 实例传递给线程  
  
    Thread thread1 = new Thread(myRunnable);  
  
    Thread thread2 = new Thread(myRunnable);  
  
    // 启动线程  
  
    thread1.start();  
  
    thread2.start();  
  
}  
}
```

2、 增强 Executor

```
import java.util.concurrent.Executor;  
  
import java.util.concurrent.Executors;
```

```
public class ExecutorDemo {

    public static void main(String[] args) {

        // 创建一个线程池，这里使用固定大小的线程池

        Executor executor = Executors.newFixedThreadPool(3);

        // 提交任务给线程池执行

        for (int i = 0; i < 5; i++) {

            final int taskId = i;

            executor.execute(() -> {

                // 在这里定义需要执行的任务逻辑

                System.out.println("Task " + taskId + " is running on thread " + Thread.c

urrentThread().getName());

            });

        }

        // 关闭线程池（通常在应用程序退出时执行）

        if (executor instanceof java.util.concurrent.ExecutorService) {

            ((java.util.concurrent.ExecutorService) executor).shutdown();

        }

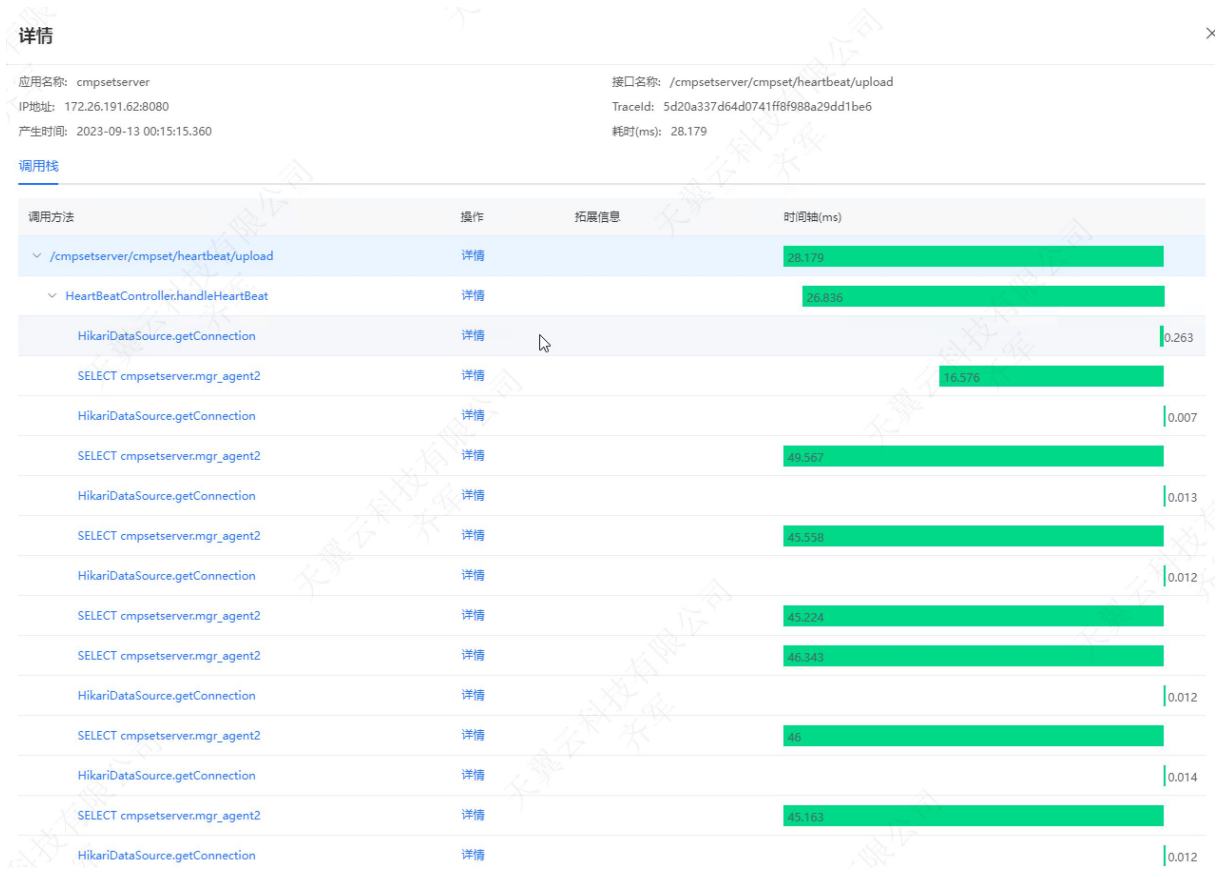
    }

}
```

```
}  
  
}
```

执行结果

配置完成后，您可以在调用链路详情页查看异步任务的调用链详情。具体详情，请参见下图。



六、常见问题

6.1、计费类

6.1.1、计费相关问题

1、如何停止应用计费

如您后期还需要重新对该应用进行监控,那么在自定义配置中,关闭 agent 总开关即可;
如果您后期不再需要监控该应用以及看历史监控数据,也可在应用列表中直接删除应用。

2、资源包消耗完后如何计费

产生探针时消耗时,优先判断当前时间是否有可用的资源包,如果有,则优先消耗资源包的探针时;如果没有,则按按需付费的方式,根据实际消耗的探针时产生费用。

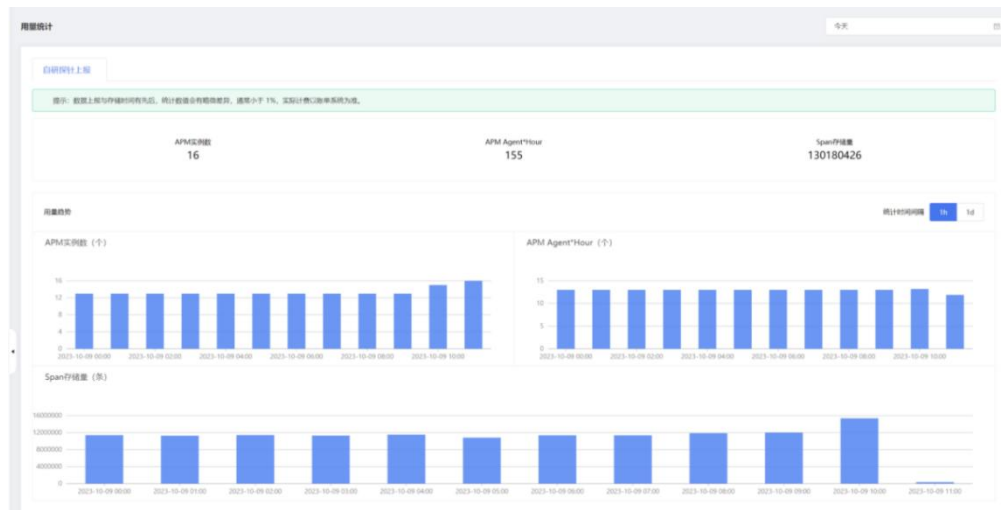
3、公测期间计费

公测期间无论您如何使用都不会产生任何费用,如果公测到期后您选择继续使用,将会按照您选择的使用方式计费。

4、如何查看 agent 用量

应用监控控制台为用户提供查看资源消耗的功能,您可以通过设置过滤条件查看单个或全部应用在特定时间内的资源消耗。

1. 在 AMS 控制台左侧导航栏中选择用量统计。
2. 在用量统计页面右上角设置需要查看的时间段。



- APM 实例数：用户所使用的探针实例个数，一个 APM 实例表示一个正在运行的 Java 服务实例
- APM Agent*Hour：1 个应用实例的 1 天消耗 = 24 Agent*Hour
- Span 存储量：span 数据的存储数

6.2、购买类

公测期间，无需操作购买

6.3、操作类

6.3.1、网络配置相关问题

如何测试网络连通性

可使用 telnet 命令测试目标主机与 APM 服务器网络是否连通。例如，以测试内蒙 6 地域的连通性为例，请登录应用所部署的机器，并输入以下命令：

```
telnet arms-data-proxy.cq2b.inner.ctyun.cn 27149
```

具体每个资源池的服务器地址参考[应用接入](#)

6.3.2、升级探针

对于在 ARMS 中监控的各类型应用，请按照本文所述的相应方法更新 Java 探针版本。

如何为虚拟机部署的应用更新探针？

如果您需要更新虚拟机部署的应用 Java 探针版本，您需要在 [应用接入](#) 里面点击下载按钮，下载最新 Java 探针，替换掉旧版本的探针，然后重新启动应用即可。

如何为 ccse 部署的应用更新探针？

如果您需要更新容器服务应用的 Java 探针版本，在开启 APM 的基础上，重新部署工作负载即可。

1. 登录[天翼云 CCSE 控制中心](#)，在左侧导航栏选择集群，在右侧页面进入集群详情。
2. 在左侧导航栏选择工作负载 > 有状态/无状态，找到对应的工作负载。
3. 点击重新部署，选择需要的部署方式，点击确定。

如何为其他类型的应用更新探针？

目前仅支持两种接入方式，其他类型正在扩充中，请关注产品动态的更新提醒。

6.3.3、为 Java 应用手动安装 Agent 的常见问题

(1) APM Agent 和其他 APM 产品 Agent（例如 SkyWalking）是否兼容？

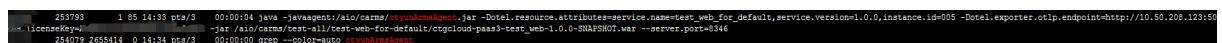
APM Agent 是基于 opentelemetry 开源项目的 Agent 进行的二次开发，和其他 APM 产品 Agent 都不兼容。APM 大多是基于 ASM 框架进行字节码插桩实现的，同时安装两个 Agent 相当于对您的代码插桩两次，由于不同厂家的插装代码实现不同，代码冲突可能造成各类问题（例如应用启动缓慢，类冲突等问题），因此强烈建议您不要同时安装多个 APM Agent。

(2) 如何检查 APM Agent 是否安装成功

使用 ps 命令查看命令行启动参数中是否成功安装 APM Agent。

```
ps -ef | grep 'ctyunArmsAgent'
```

成功安装时，如下图所示：



```
253793 1 65 14:33 pts/0 00:00:04 java -javaagent:/aio/carma/ctyunArmsAgent.jar -Dotel.resource.attributes=service.name=test_web_for_default,service.version=1.0.0,instance.id=006 -Dotel.exporter.otlp.endpoint=http://10.50.208.123:4315 --spring.datasource.url=jdbc:mysql://10.50.208.123:3306/test --spring.datasource.username=root --spring.datasource.password=123456 --spring.datasource.driver-class-name=com.mysql.jdbc.Driver --spring.jpa.hibernate.ddl-auto=update --spring.jpa.show-sql=true --spring.jpa.properties.hibernate.format_sql=true --server.port=8346
254079 2655414 0 14:34 pts/0 00:00:00 grep --color=auto ctyunArmsAgent
```

6.3.4、APM Agent 安装成功，为什么控制台上仍无监控数据？

问题现象

APM Agent 安装成功，控制台上仍无监控数据。

可能原因

应用无持续的外部请求访问、或者 APM Agent 安装目录权限不正确等原因都有可能导致控制台无监控数据。

解决方案

- 如果 Agent 日志中出现 “send agent metrics. no metrics.”，请确认您的应用是否有持续的外部请求访问，包括 HTTP 请求、HSF 请求和 Dubbo 请求，并确认开发框架是否在 APMAgent 的支持范围内。关于 APM 应用监控对第三方组件和框架的支持情况，请参见 APM 应用监控支持的 Java 组件和框架。
- 确认选择的查询时间范围是否正确。请您将查询时间条件设为最近 5 分钟，然后再次确认是否有监控数据。
- 如果是通过 -jar 命令行启动的，请检查命令行设置，确保 -javaagent 参数在 -jar 之前。

```
java -javaagent:{user.workspace}/ctyunArmsAgent.jar -Dotel.exporter.otlp.endpoint=xxx.xxx.xxx.xxx:xxxxxx -Darms.licenseKey=xxx -jar testWeb.jar
```

- 如果 {user.workspace}/ctyunjavaagent/ctyunarmsagent.log 的日志中出现 transData span is faile 或 transData metrics is faile 异常，请您检查应用与启动参数中的 otel.exporter.otlp.endpoint 指向的地址是否连通。

- 如果{user.workspace}/ctyunjavaagent/ctyunarmsagent.log 的日志中出现 gateway sdk initFail 异常, 请您检查应用所属地域与 Agent 所属地域是否一致。
- 如果{user.workspace}/sdklogs/gw-sdk.log 的日志中出现 LICENSE_IS_DISABLE 异常, 请您检查 license 是否已过期或是超出配额限制。
- 如果应用启动之后{user.workspace}/目录下无 ctyunjavaagent 和子目录, 是由于 ctyunArmsAgent.jar 未被成功加载导致的, 请您检查 APMAgent 安装目录的权限是否正确。

6.3.5、安装 Agent 后应用启动时报 OutOfMemoryError 错误怎么办？

请在 Java 启动命令后面加上堆内存大小配置项, 以适当调大 JVM 参数。以下示例配置项表示堆内存初始值 (Xms) 为 512 M, 堆内存最大值 (Xmx) 为 2 G。

说明 请根据实际情况适当调节。对于 Tomcat 等其他环境, 请在配置文件的

JAVA_OPTS 中加入此参数。

```
-Xms512M
```

```
-Xmx2048M
```

如果出现 OutOfMemoryError: PermGen space 错误, 请添加以下配置。

```
-XX:PermSize=256M
```

```
-XX:MaxPermSize=512M
```

如果出现 OutOfMemoryError: metaspace 错误, 请添加以下配置。

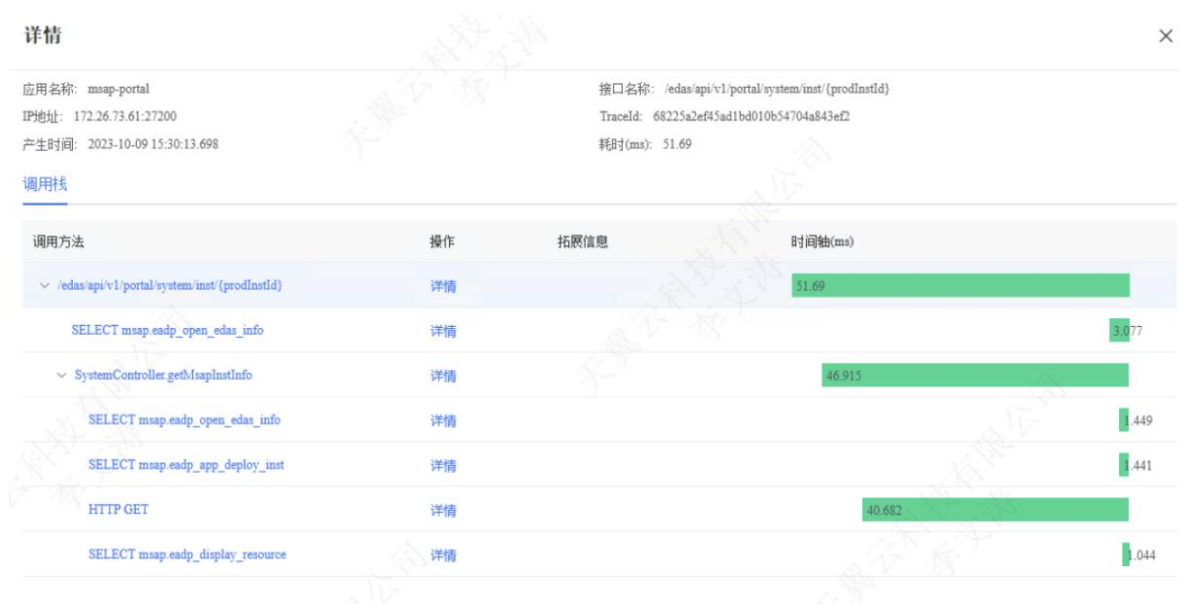
-XX:MetaspaceSize=256M

-XX:MaxMetaspaceSize=512M

6.3.6、调用链的时间线是如何绘制的？

问题现象

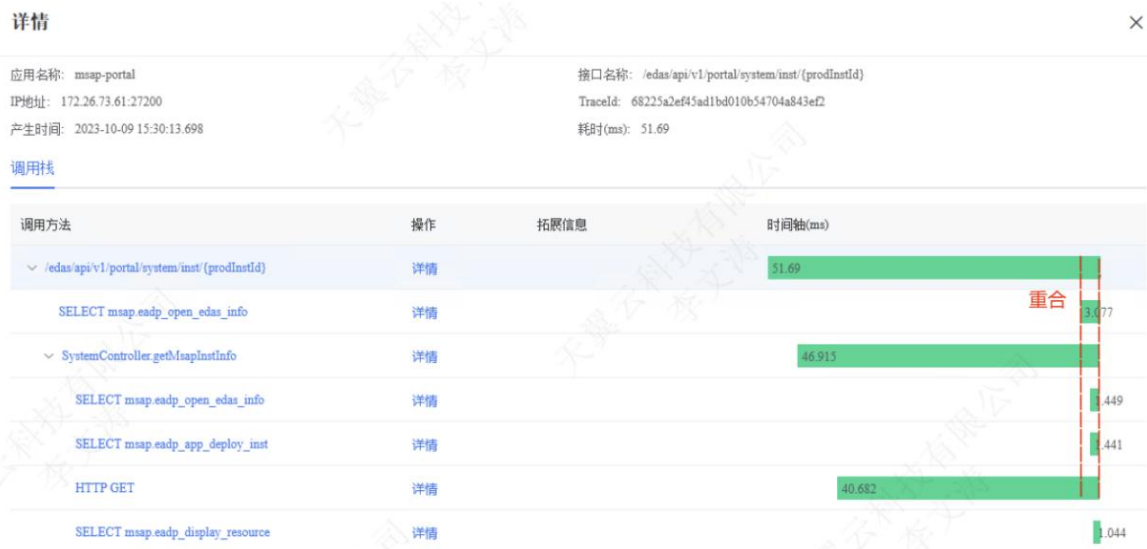
调用链的时间线是如何绘制的？当调用链的时间线类似如下图所示时，是绘制错误了吗？



问题解答

每个方法的时间线是表示该方法的起始位置与总耗时。在调用链界面中，方法线用灰色绘制，每一个方法的耗时用绿色填充。绿色填充段表示总耗时。当您看到上图所示的调用链时，可能会有疑惑：方法 1、2 的时间线有部分重合。

其实，并没有绘制错误，由于 APM 的调用链绘图中，每个方法的时间线是绘制的起始位置与总耗时，所以绘制会出现问题现象所描述的情况。



6.3.7、其他问题

(1) 为什么没有看到对应数量的调用链？

问题现象

为什么发起多次请求，但产生的调用链数量较少？

可能原因

调用链数量和设置的调用链采样率有关。

您可以在 [APM 控制台](#) 目标应用的应用设置 > 自定义配置页签的采样率设置区域

查看调用链采样率。默认的采样率为 1%，即只有 1%的调用链会被采集。

采样规则：错误与异常调用的调用链会被默认采样。

(2) GC 次数为什么有小数？

问题现象

正常不会出现垃圾回收一半就暂停的情况，但 GC 次数出现小数

可能原因

GC 时间的单位为 ms，次数按照平均数计算，所以会出现小数。

