



应用实时监控服务 ARMS 应用监控

文档版本: 20220713



法律声明

阿里云提醒您在阅读或使用本文档之前仔细阅读、充分理解本法律声明各条款的内容。 如果您阅读或使用本文档,您的阅读或使用行为将被视为对本声明全部内容的认可。

- 您应当通过阿里云网站或阿里云提供的其他授权通道下载、获取本文档,且仅能用 于自身的合法合规的业务活动。本文档的内容视为阿里云的保密信息,您应当严格 遵守保密义务;未经阿里云事先书面同意,您不得向任何第三方披露本手册内容或 提供给任何第三方使用。
- 未经阿里云事先书面许可,任何单位、公司或个人不得擅自摘抄、翻译、复制本文 档内容的部分或全部,不得以任何方式或途径进行传播和宣传。
- 由于产品版本升级、调整或其他原因,本文档内容有可能变更。阿里云保留在没有 任何通知或者提示下对本文档的内容进行修改的权利,并在阿里云授权通道中不时 发布更新后的用户文档。您应当实时关注用户文档的版本变更并通过阿里云授权渠 道下载、获取最新版的用户文档。
- 4. 本文档仅作为用户使用阿里云产品及服务的参考性指引,阿里云以产品及服务的"现状"、"有缺陷"和"当前功能"的状态提供本文档。阿里云在现有技术的基础上尽最大努力提供相应的介绍及操作指引,但阿里云在此明确声明对本文档内容的准确性、完整性、适用性、可靠性等不作任何明示或暗示的保证。任何单位、公司或个人因为下载、使用或信赖本文档而发生任何差错或经济损失的,阿里云不承担任何法律责任。在任何情况下,阿里云均不对任何间接性、后果性、惩戒性、偶然性、特殊性或刑罚性的损害,包括用户使用或信赖本文档而遭受的利润损失,承担责任(即使阿里云已被告知该等损失的可能性)。
- 5. 阿里云网站上所有内容,包括但不限于著作、产品、图片、档案、资讯、资料、网站架构、网站画面的安排、网页设计,均由阿里云和/或其关联公司依法拥有其知识产权,包括但不限于商标权、专利权、著作权、商业秘密等。非经阿里云和/或其关联公司书面同意,任何人不得擅自使用、修改、复制、公开传播、改变、散布、发行或公开发表阿里云网站、产品程序或内容。此外,未经阿里云事先书面同意,任何人不得为了任何营销、广告、促销或其他目的使用、公布或复制阿里云的名称(包括但不限于单独为或以组合形式包含"阿里云"、"Aliyun"、"万网"等阿里云和/或其关联公司品牌,上述品牌的附属标志及图案或任何类似公司名称、商号、商标、产品或服务名称、域名、图案标示、标志、标识或通过特定描述使第三方能够识别阿里云和/或其关联公司)。
- 6. 如若发现本文档存在任何错误,请与阿里云取得直接联系。

通用约定

格式	说明	样例
⚠ 危险	该类警示信息将导致系统重大变更甚至故 障,或者导致人身伤害等结果。	介 危险 重置操作将丢失用户配置数据。
▲ 警告	该类警示信息可能会导致系统重大变更甚 至故障,或者导致人身伤害等结果。	會学者 重启操作将导致业务中断,恢复业务 时间约十分钟。
〔) 注意	用于警示信息、补充说明等,是用户必须 了解的内容。	大) 注意 权重设置为0,该服务器不会再接受新 请求。
? 说明	用于补充说明、最佳实践、窍门等 <i>,</i> 不是 用户必须了解的内容。	⑦ 说明 您也可以通过按Ctrl+A选中全部文件。
>	多级菜单递进。	单击设置> 网络> 设置网络类型。
粗体	表示按键、菜单、页面名称等UI元素。	在 结果确认 页面,单击 确定 。
Courier字体	命令或代码。	执行 cd /d C:/window 命令 <i>,</i> 进入 Windows系统文件夹。
斜体	表示参数、变量。	bae log listinstanceid Instance_ID
[] 或者 [alb]	表示可选项,至多选择一个。	ipconfig [-all -t]
{} 或者 {a b}	表示必选项,至多选择一个。	switch {act ive st and}

目录

1.应用监控概述	80
2.接入应用监控	09
2.1. 应用监控接入概述	09
2.2. 开始监控Java应用	09
2.2.1. 为Java应用手动安装Agent	09
2.2.2. 使用脚本为Java应用快速安装探针	17
2.2.3. 为函数计算中的Java应用安装探针	19
2.2.4. 为容器服务Kubernetes版Java应用安装探针	20
2.2.5. 为开源Kubernetes环境中的应用安装探针	23
2.2.6. 为Docker中的Java应用安装Agent	26
2.2.7. 通过OpenTelemetry接入ARMS	32
2.3. 开始监控除Java之外的应用	36
3.控制台功能	38
3.1. 应用监控3D拓扑图	38
3.2. 调用链路查询	42
3.3. Trace Explorer	44
3.4. ARMS OpenTelemetry Collector	50
3.5. 应用总览	58
3.6. 应用详情	60
3.6.1. 概览	60
3.6.2. JVM监控	63
3.6.3. 池化监控	65
3.6.4. 主机监控	67
3.6.5. Pod监控	68
3.6.6. SQL调用分析	71
3.6.7. NoSQL调用分析	72

3.6.8. 异常分析	73
3.6.9. 错误分析	74
3.6.10. 上游应用	76
3.6.11. 调用链查看	77
3.6.12. 日志	78
3.6.13. 内存快照	80
3.7. 定时任务	86
3.8. 接口调用	90
3.9. 事件中心	92
3.10. 数据库调用	97
3.11. 外部调用	100
3.12. MQ监控	102
3.13. 主动剖析	104
3.14. 应用诊断	107
3.14.1. 实时诊断	107
3.14.2. 启动分析	109
3.14.3. 异常分析	111
3.14.4. 线程分析	114
3.14.5. 日志分析(直接采集)	115
3.14.6. 日志分析(日志服务SLS)	118
3.14.7. 日志分析(旧版)	120
3.14.8. Arthas诊断(新版)	123
3.15. JVM配置优化	130
3.16. 应用环境	133
3.17. 应用设置	134
3.17.1. 自定义配置	134
3.17.2. 标签管理	139
3.17.3. 监控方法自定义	140

	3.17.4. 删除应用	141
	3.18. 应用监控告警规则(新版)	142
4.	.最佳实践	- 146
	4.1. 使用调用链采样策略	146
	4.2. 使用线程剖析诊断代码层面的问题	153
	4.3. 诊断服务端报错问题	155
	4.4. 诊断应用卡顿问题	158
	4.5. 通过诊断报告排查异常情况	161
	4.6. 业务日志关联调用链的TraceId信息	164
	4.7. 非阿里云环境下部署ARMS	165
	4.8. 在用户专有网络中部署ARMS	- 167
	4.9. 通过调用链路和日志分析定位业务异常问题	169
	4.10. 将ARMS页面嵌入自建Web应用	172
	4.11. 使用ARMS监控异步任务	176
	4.12. 通过TraceExplorer实时分析链路数据	179
	4.13. 通过OpenTelemetry Java SDK进行手工埋点	183
	4.14. 在阿里云Prometheus监控下查看应用监控大盘	188
5.	.参考信息	- 189
	5.1. ARMS Java探针性能压测报告	189
	5.2. ARMS应用监控支持的Java组件和框架	193
	5.3. ARMS应用监控支持的Kafka版本	196
	5.4. Agent版本说明	198
	5.5. 关键统计指标说明	202
	5.6. ARMS SDK使用说明	206
	5.7. Trace Explorer参数说明	208
	5.8. Trace Explorer查询用法说明	209
	5.9. ARMS OpenTelemetry Collector指标说明	211
	5.10. ARMS OpenTelemetry Collector配置说明	215

5.11. 应用监控指标说明	217
6.升级探针	225
7.应用监控常见问题	226
8.故障排除	242
8.1. ARMS Agent安装成功,为什么控制台上仍无监控数据?	242
8.2. 为什么容器服务Kubernetes版集群中的Java应用安装Agent后,应	242

1.应用监控概述

ARMS应用监控是一款应用性能管理(Application Performance Management,简称APM)产品。您无需修改代码,只需为应用安装一个探针,ARMS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。

自动发现应用拓扑

ARMS应用监控探针能够自动发现应用的上下游依赖关系。具体而言,该探针能够有效捕获、智能计算、自动展示不同应用之间通过RPC框架(例如Dubbo、HTTP、HSF等协议)组成的调用链。您可以通过应用拓扑 轻松发现系统中的性能瓶颈和异常调用。

3D拓扑

3D拓扑图能立体展示应用、服务和主机的健康状况,以及应用的上下游依赖关系,帮助您快速定位诱发故障的服务、被故障影响的应用和关联的主机等,全方位地诊断故障根源,从而快速排除故障。

捕获异常事务和慢事务

您可以进一步获取接口的慢SQL、MQ堆积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。

自动发现并监控接口

ARMS应用监控能够自动发现和监控应用代码中常见的Web框架和RPC框架,并自动统计Web接口和RPC接口的调用量、响应时间、错误数等指标。

实时诊断

当您需要密切监控一小段时间内的应用性能时,例如发布应用或者对应用进行压测时,可以使用ARMS应用 监控的实时诊断功能。开启实时诊断后,ARMS应用监控会持续监控应用5分钟,并在这5分钟内全量上报调 用链数据。接下来,您就能以出现性能问题的调用链路为起点,通过方法栈瀑布图和线程剖析等功能定位问 题原因。

多维排查

您可以查看分布式及本地方法栈明细,并按应用、IP、耗时等维度进行多维分析。

集成阿里云中间件PaaS平台

ARMS应用监控支持一键集成阿里云PaaS平台EDAS,让运行于阿里中间件分布式架构平台上的应用监控更加 有效。



2. 接入应用监控

2.1. 应用监控接入概述

应用安装探针的准备工作方便您使用ARMS控制台查看丰富的应用监控指标。本文按应用部署环境和应用语 言的维度列出了所有安装探针的文档。

按部署环境开始监控

按应用语言开始监控

⑦ 说明 上文的星号(*)表示需开通链路追踪Tracing Analysis服务。

2.2. 开始监控Java应用

2.2.1. 为Java应用手动安装Agent

为Java应用安装Agent后,ARMS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。

前提条件

确保您使用的云服务器ECS实例的安全组已开放8442、8443、8883三个端口的TCP出方向权限。为云服务器ECS开放出方向权限的方法,请参见添加安全组规则。

⑦ 说明 ARMS不仅可接入阿里云ECS上的应用,还能接入其他能访问公网的服务器上的应用。

- 确保您使用的第三方组件或框架在应用监控兼容性列表范围内,请参见应用监控兼容性列表。
- 检查您的JDK版本。ARMS应用监控支持的JDK版本如下:
 - JDK 1.7.0+
 - JDK 1.8.0_25+

? 说明

- Kubernetes集群应用部署建议: JDK 1.8.0_191+。
- 如果JDK版本为1.8.0_25或者1.8.0_31,可能会出现无法安装探针的情况,请升级至1.8.X最新版本。

• JDK 11.0.8+

⑦ 说明 JDK 1.8及以下版本和JDK 11版本对应的探针安装包不同,请根据不同的JDK版本下载对应的 探针安装包或调整应用监控组件ack-onepilot的配置。

操作步骤

- 1. 登录ARMS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选择目标地域,然后单击接入应用。

- 3. 在接入应用面板单击Java,然后选择手动安装。
- 4. 下载Agent。
 - 方法一:手动下载。在接入Java面板的STEP1区域根据JDK版本下载对应的Agent。
 - 方法二:使用Wget命令下载。根据您的地域下载对应的Agent安装包。

⑦ 说明 请使用公网地址,如无法下载则使用VPC地址。

查看各地域对应的Agent(JDK 8及以下版本)安装包下载命令

地域	公网地址	VPC地址
华东1(杭州)	wget "http://arms-apm-cn- hangzhou.oss-cn- hangzhou.aliyuncs.com/ArmsAge nt.zip" -O ArmsAgent.zip	wget "http://arms-apm-cn- hangzhou.oss-cn-hangzhou- internal.aliyuncs.com/ArmsAg ent.zip" -O ArmsAgent.zip
华东2(上海)	wget "http://arms-apm-cn- shanghai.oss-cn- shanghai.aliyuncs.com/ArmsAge nt.zip" -O ArmsAgent.zip	wget "http://arms-apm-cn- shanghai.oss-cn-shanghai- internal.aliyuncs.com/ArmsAg ent.zip" -0 ArmsAgent.zip
华北1(青岛)	wget "http://arms-apm-cn- qingdao.oss-cn- qingdao.aliyuncs.com/ArmsAgen t.zip" -O ArmsAgent.zip	wget "http://arms-apm-cn- qingdao.oss-cn-qingdao- internal.aliyuncs.com/ArmsAg ent.zip" -O ArmsAgent.zip
华北2(北京)	wget "http://arms-apm-cn- beijing.oss-cn- beijing.aliyuncs.com/ArmsAgen t.zip" -O ArmsAgent.zip	wget "http://arms-apm-cn- beijing.oss-cn-beijing- internal.aliyuncs.com/ArmsAg ent.zip" -O ArmsAgent.zip
华北3(张家口)	wget "http://arms-apm-cn- zhangjiakou.oss-cn- zhangjiakou.aliyuncs.com/Arms Agent.zip" -O ArmsAgent.zip	wget "http://arms-apm-cn- zhangjiakou.oss-cn- zhangjiakou- internal.aliyuncs.com/ArmsAg ent.zip" -O ArmsAgent.zip

应用监控·接入应用监控

应用实时监控服务 ARMS

地域	公网地址	VPC地址
华北5(呼和浩 特)	wget "http://arms-apm-cn- huhehaote.oss-cn- huhehaote.aliyuncs.com/ArmsAg ent.zip" -O ArmsAgent.zip	wget "http://arms-apm-cn- huhehaote.oss-cn-huhehaote- internal.aliyuncs.com/ArmsAg ent.zip" -O ArmsAgent.zip
华北6(乌兰察 布)	wget "http://arms-apm-cn- wulanchabu.oss-cn- wulanchabu.aliyuncs.com/ArmsA gent.zip" -O ArmsAgent.zip	wget "http://arms-apm-cn- wulanchabu.oss-cn- wulanchabu- internal.aliyuncs.com/ArmsAg ent.zip" -O ArmsAgent.zip
华南1(深圳)	<pre>wget "http://arms-apm-cn- shenzhen.oss-cn- shenzhen.aliyuncs.com/ArmsAge nt.zip" -0 ArmsAgent.zip</pre>	<pre>wget "http://arms-apm-cn- shenzhen.oss-cn-shenzhen- internal.aliyuncs.com/ArmsAg ent.zip" -0 ArmsAgent.zip</pre>
华南2(河源)	wget "http://arms-apm-cn- heyuan.oss-cn- heyuan.aliyuncs.com/ArmsAgent .zip" -O ArmsAgent.zip	wget "http://arms-apm-cn- heyuan.oss-cn-heyuan- internal.aliyuncs.com/ArmsAg ent.zip" -O ArmsAgent.zip
华南3(广州)	wget "http://arms-apm-cn- guangzhou.oss-cn- guangzhou.aliyuncs.com/ArmsAg ent.zip" -O ArmsAgent.zip	wget "http://arms-apm-cn- guangzhou.oss-cn-guangzhou- internal.aliyuncs.com/ArmsAg ent.zip" -O ArmsAgent.zip
西南1(成都)	wget "http://arms-apm-cn- chengdu.oss-cn- chengdu.aliyuncs.com/ArmsAgen t.zip" -O ArmsAgent.zip	wget "http://arms-apm-cn- chengdu.oss-cn-chengdu- internal.aliyuncs.com/ArmsAg ent.zip" -O ArmsAgent.zip
中国(香港)	wget "http://arms-apm-cn- hongkong.oss-cn- hongkong.aliyuncs.com/ArmsAge nt.zip" -O ArmsAgent.zip	wget "http://arms-apm-cn- hongkong.oss-cn-hongkong- internal.aliyuncs.com/ArmsAg ent.zip" -O ArmsAgent.zip

应用实时监控服务 ARMS

应用监控·接入应用监控

地域	公网地址	VPC地址
亚太东南1(新加 坡)	<pre>wget "http://arms-apm-ap- southeast-1.oss-ap-southeast- 1.aliyuncs.com/ArmsAgent.zip" -0 ArmsAgent.zip</pre>	<pre>wget "http://arms-apm-ap- southeast-1.oss-ap- southeast-1- internal.aliyuncs.com/ArmsAg ent.zip" -0 ArmsAgent.zip</pre>
亚太东南2(悉 尼)	<pre>wget "http://arms-apm-ap- southeast-2.oss-ap-southeast- 2.aliyuncs.com/ArmsAgent.zip" -0 ArmsAgent.zip</pre>	<pre>wget "http://arms-apm-ap- southeast-2.oss-ap- southeast-2- internal.aliyuncs.com/ArmsAg ent.zip" -0 ArmsAgent.zip</pre>
亚太东南3(吉隆 坡)	<pre>wget "http://arms-apm-ap- southeast-3.oss-ap-southeast- 3.aliyuncs.com/ArmsAgent.zip" -0 ArmsAgent.zip</pre>	<pre>wget "http://arms-apm-ap- southeast-3.oss-ap- southeast-3- internal.aliyuncs.com/ArmsAg ent.zip" -0 ArmsAgent.zip</pre>
亚太东南5(雅加 达)	<pre>wget "http://arms-apm-ap- southeast-5.oss-ap-southeast- 5.aliyuncs.com/ArmsAgent.zip" -0 ArmsAgent.zip</pre>	<pre>wget "http://arms-apm-ap- southeast-5.oss-ap- southeast-5- internal.aliyuncs.com/ArmsAg ent.zip" -0 ArmsAgent.zip</pre>
亚太东北1(东 京)	<pre>wget "http://arms-apm-ap- northeast-1.oss-ap-northeast- 1.aliyuncs.com/ArmsAgent.zip" -0 ArmsAgent.zip</pre>	<pre>wget "http://arms-apm-ap- northeast-1.oss-ap- northeast-1- internal.aliyuncs.com/ArmsAg ent.zip" -0 ArmsAgent.zip</pre>
欧洲中部1(法兰 克福)	<pre>wget "http://arms-apm-eu- central-1.oss-eu-central- 1.aliyuncs.com/ArmsAgent.zip" -0 ArmsAgent.zip</pre>	wget "http://arms-apm-eu- central-1.oss-eu-central-1- internal.aliyuncs.com/ArmsAg ent.zip" -O ArmsAgent.zip
欧洲西部1(伦 敦)	<pre>wget "http://arms-apm-eu- west-1.oss-eu-west- 1.aliyuncs.com/ArmsAgent.zip" -0 ArmsAgent.zip</pre>	wget "http://arms-apm-eu- west-1.oss-eu-west-1- internal.aliyuncs.com/ArmsAg ent.zip" -O ArmsAgent.zip

应用监控·接入应用监控

应用实时监控服务 ARMS

地域	公网地址	VPC地址
美国东部1(弗吉 尼亚)	<pre>wget "http://arms-apm-us- east-1.oss-us-east- 1.aliyuncs.com/ArmsAgent.zip" -0 ArmsAgent.zip</pre>	wget "http://arms-apm-us- east-1.oss-us-east-1- internal.aliyuncs.com/ArmsAg ent.zip" -O ArmsAgent.zip
美国西部1(硅 谷)	<pre>wget "http://arms-apm-us- west-1.oss-us-west- 1.aliyuncs.com/ArmsAgent.zip" -0 ArmsAgent.zipwget "http://arms-apm-us-west- 1.oss-us-west-1- internal.aliyuncs.com/ArmsAge nt.zip" -0 ArmsAgent.zip</pre>	wget "http://arms-apm-us- west-1.oss-us-west-1- internal.aliyuncs.com/ArmsAg ent.zip" -O ArmsAgent.zip
亚太南部1(孟 买)	<pre>wget "http://arms-apm-ap- south-1.oss-ap-south- 1.aliyuncs.com/ArmsAgent.zip" -0 ArmsAgent.zip</pre>	<pre>wget "http://arms-apm-ap- south-1.oss-ap-south-1- internal.aliyuncs.com/ArmsAg ent.zip" -0 ArmsAgent.zip</pre>
华东1金融云	无	wget "http://arms-apm-cn- hangzhou-finance.oss-cn- hzjbp-b- internal.aliyuncs.com/ArmsAg ent.zip" -0 ArmsAgent.zip
华东2金融云	无	<pre>wget "http://arms-apm-cn- shanghai-finance-1.oss-cn- shanghai-finance-1- internal.aliyuncs.com/ArmsAg ent.zip" -0 ArmsAgent.zip</pre>

地域	公网地址	VPC地址
华南1金融云	无	<pre>wget "http://arms-apm-cn- shenzhen-finance-1.oss-cn- shenzhen-finance-1- internal.aliyuncs.com/ArmsAg ent.zip" -0 ArmsAgent.zip</pre>
政务云	<pre>wget "http://arms-apm-cn- north-2-gov-1.oss-cn-north-2- gov- 1.aliyuncs.com/ArmsAgent.zip" -0 ArmsAgent.zip</pre>	<pre>wget "http://arms-apm-cn- north-2-gov-1.oss-cn-north- 2-gov-1- internal.aliyuncs.com/ArmsAg ent.zip" -0 ArmsAgent.zip</pre>

5. 安装Agent。

i. 进入Agent安装包所在目录,并执行以下命令来解压安装包到任意工作目录下。

unzip ArmsAgent.zip -d /{user.workspace}/

⑦ 说明 {user.workspace}是示例目录,请替换为真实的目录。

ii. 添加AppName和LicenseKey参数。

<i>STEP2</i> ● 安装	1. 进入Agent压缩包标在目录并将其解压至任意工作目录下。	
Agent	unzip ArmsAgent.zip -d /(user.workspace)/	a
	● "(user.workspace)"是予例路径,此处及以下均衡使振興体环境整绝为正确的路径。	
	2. 原DIAppNameUJQUCarasKey参数(同時方法任団員一) 方法一: 复制ULTIVM参数,源DIS回用部务的制設辦本中。 四用名称 my-service O	
	-jevasgent:/(user.workspace)/ams-gent/ams-bootstrap-1.7.e-SNAPSHOT.jan Larms.licenseksy-do -Darms.appName-my-service	٥
	 ams.ilcensetey 是例门为您自动注意的。 ams.appName 是您的应用名称, 请希 my-service 曹操成您自己的应用名 	
	方法二: 棕花ams-agent.config, 曾映ams.iicenseKey汲ams.appName配且定义。	
	erns.llcensekey-bb erns.appiame.my-service	٥

⑦ 说明 将示例代码中的 {LicenseKey} 替换成STEP2区域获取License Key, {AppName} 替换为您实际的应用名称(应用名不可包含中文字符), {user.workspace} 替换成实际 Agent安装包的解压目录,将demoApp.jar替换为真实的JAR包地址。

■ 方法一: 根据您的应用运行环境修改JVM参数。

运行环境 步骤

运行环境	步骤
Tomcat(Lin ux或macOS 操作系统)	在{TOMCAT_HOME}/bin/setenv.sh文件中添加以下配置。
	<pre>JAVA_OPTS="\$JAVA_OPTS - javaagent:/{user.workspace}/ArmsAgent/arms-bootstrap-1.7.0- SNAPSHOT.jar -Darms.licenseKey={LicenseKey} -Darms.appName= {AppName}"</pre>
	如果您的Tomcat版本没有 <i>setenv.sh</i> 配置文件,请打开 <i>{TOMCAT_HOME}/bin/catalina <i>.sh</i>文件,并在JAVA_OPTS后添加上述配置,具体示例,请参见<mark>catalina.sh</mark>的第256 行。</i>
	在{TOMCAT_HOME}/bin/catalina.bat文件中添加以下配置。
Tomcat (Wi	<pre>set "JAVA_OPTS=%JAVA_OPTS% - javaagent:/{user.workspace}/ArmsAgent/arms-bootstrap-1.7.0- SNAPSHOT.jar -Darms.licenseKey={LicenseKey} -Darms.appName= {AppName}"</pre>
ndows操作 系统)	如果以上配置未生效,可以尝试在 <i>{TOMCAT_HOME}/bin/catalina.bat</i> 文件中添加以下 配置。
	<pre>set "CATALINA_OPTS=-javaagent:/{user.workspace}/arms-bootstrap- 1.7.0-SNAPSHOT.jar -Darms.licenseKey={LicenseKey} - Darms.appName={AppName}"</pre>
	在{JETTY_HOME}/start.in配置文件中添加以下配置。
Jetty	exec -javaagent:/{user.workspace}/ArmsAgent/arms-bootstrap-1.7.0- SNAPSHOT.jar -Darms.licenseKey={LicenseKey} -Darms.appName={AppName}
	启动Spring Boot进程时,在启动命令后加上-javaagent参数。
Spring Boot	java -javaagent:/{user.workspace}/ArmsAgent/arms-bootstrap- 1.7.0-SNAPSHOT.jar -Darms.licenseKey={LicenseKey} - Darms.appName={AppName} -jar demoApp.jar

运行环境	步骤
	启动Resin进程时,在 <i>conf/resin.xml</i> 配置文件中添加以下标签。
	<pre><server-default> <jvm-arg>-javaagent:{user.workspace}/ArmsAgent/arms- bootstrap-1.7.0-SNAPSHOT.jar</jvm-arg> <jvm-arg>-Darms.licenseKey={LicenseKey}</jvm-arg> <jvm-arg>-Darms.appName={AppName}</jvm-arg> </server-default></pre>
Resin	在 <i>conf/app-default.xml</i> 文件中添加以下标签。
	<library-loader path="{user.workspace}/ArmsAgent/plugin"></library-loader>
	使用CMD启动Java进程时,在挂载Agent路径中使用反斜线(\)作为分隔符。
Windows	<pre>java -javaagent:\{user.workspace}\ArmsAgent\arms-bootstrap- 1.7.0-SNAPSHOT.jar -Darms.licenseKey={LicenseKey} - Darms.appName={AppName} -jar {user.workspace}\demoApp.jar</pre>

如需在一台服务器上部署同一应用的多个实例,可以通过-Darms.agentId参数(逻辑编号)来区 分接入的JVM进程,例如:

```
java -javaagent:/{user.workspace}/ArmsAgent/arms-bootstrap-1.7.0-SNAPSHOT.jar -Da
rms.licenseKey={LicenseKey} -Darms.appName={AppName} -Darms.agentId=001 -jar demo
App.jar
```

■ 方法二:在arms-agent.config文件中添加以下配置。

arms.licenseKey={LicenseKey} arms.appName={AppName}

6. 在Java应用的启动脚本中添加以下参数。

-javaagent:/{user.workspace}/ArmsAgent/arms-bootstrap-1.7.0-SNAPSHOT.jar

7. 重启Java应用。

结果验证

约一分钟后,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。

卸载Agent

当您不需要使用ARMS监控您的Java应用时,请按照以下步骤卸载Agent。

- 1. 删除步骤7中添加的{AppName}、{LicenseKey}等所有参数。
- 2. 重启Java应用。

快速更改应用名称

如需更改应用名称,例如忘记将示例应用名称Java-Demo修改为自定义名称,您可以在不重启应用、不重装 Agent的情况下更改应用名称,具体操作,请参见以通用方式安装Agent的普通Java应用如何更改应用名称?。

相关文档

• 应用监控常见问题

2.2.2. 使用脚本为Java应用快速安装探针

ARMS提供一键接入方式为Java应用安装探针,安装成功后无需重启应用即可开始监控,适用于新手用户。 当应用重启时,探针会自动加载,该Java应用将自动接入ARMS应用监控。

前提条件

确保您使用的云服务器ECS实例的安全组已开放8442、8443、8883三个端口的TCP出方向权限。为云服务器ECS开放出方向权限的方法,请参见添加安全组规则。

⑦ 说明 ARMS不仅可接入阿里云ECS上的应用,还能接入其他能访问公网的服务器上的应用。

- 确保您使用的第三方组件或框架在应用监控兼容性列表范围内,请参见应用监控兼容性列表。
- 检查您的JDK版本。ARMS应用监控支持的JDK版本如下:
 - JDK 1.7.0+
 - JDK 1.8.0_25+
 - ? 说明
 - Kubernetes集群应用部署建议: JDK 1.8.0_191+。
 - 如果JDK版本为1.8.0_25或者1.8.0_31,可能会出现无法安装探针的情况,请升级至1.8.X最新版本。

• JDK 11.0.8+

② 说明 JDK 1.8及以下版本和JDK 11版本对应的探针安装包不同,请根据不同的JDK版本下载对应的 探针安装包或调整应用监控组件ack-onepilot的配置。

● 若您的应用已经按照手动接入方式接入ARMS应用监控,则需先卸载探针才能正常使用一键接入方式,请 参见<mark>卸载探针</mark>。

操作步骤

- 1. 登录ARMS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选择目标地域,然后单击接入应用。
- 3. 在接入应用面板单击Java,然后选择脚本安装。
- 4. 执行STEP1区域的安装脚本。

← 擴入 Java 縦					
手动安装 建议生产环境试用,手动下载安装Agent。		脚本安装 建议能成环境使用,使用邮本快速安装Agent,简单方便,	容器服务安装 适用于使用容器服务Kubernetes的用户。		
STEP1 ● 执行脚本	在本机使用場应用相同的用户执行安装脚本				
安装	安美 viget -G- http://anns-apm-hangzhou.ess-cn-hangzhou.aliyuncs.com/install.sh sh 品 -/.anns/supervisor/cli.sh				
	● 型目动生成的、无带惊骇、只带高(ara-Demo曹操为归的应用名称。				
STEP2 🔵	STEP2 • 这员一 约瑟上只有一个isro世纪、安徽体本就从边探察进程、无需能的操作				
Agent	2% Lgent 杨荣二 机藏上存在多个Invo进程。脚半会列出它们,请选择你的应用进程				
STEP3 大売量価回用、毎号付約局約7日23時16番到上時約235枚援 単行数投資 人売量価回用、毎号付約局約7日23時16番到上時約235枚援 上段 C 持備量用4,44mm 時間第四周前1602 以早板後下,所以可認知為中が自動和構成性。					
THE END					

wget -O- http://arms-apm-hangzhou.oss-cn-hangzhou.aliyuncs.com/install.sh | sh && ~/.ar ms/supervisor/cli.sh ****** Java-Demo

? 说明

- 将 Java-Demo 替换成您的应用名,应用名暂不支持中文。
- 执行安装脚本后, 该脚本会自动下载最新探针。
- 若您的服务器只有一个Java进程,安装脚本会默认选择该进程安装探针;若您的服务器有多 个Java进程,请根据提示选择一个进程安装探针。

结果验证

约一分钟后,若您的应用出现在应用列表中且有数据上报,则说明接入成功。

卸载探针

1. 当您不需要使用ARMS监控您的Java应用时,执行 jps -1 命令查看所有进程,并在执行结果中找到 c om.alibaba.mw.arms.apm.supervisor.daemon.Daemon 对应的进程号。

在本示例中, com.alibaba.mw.arms.apm.supervisor.daemon.Daemon 对应的进程号为: 62857。



2. 执行命令 kill -9 <进程号> 。

例如: kill -9 62857 。

- 3. 执行 rm -rf /.arms /root/.arms 。
- 4. 重启您的应用。

快速更改应用名称

如果因为某些原因希望更改应用名称,例如忘记将示例应用名称Java-Demo修改为自定义名称,您可以在不重启应用、不重装探针的情况下更改应用名称,详情参见以快速方式安装探针的普通Java应用如何更改应用名称。

常见问题

1 加里在执行—键接λlava应田脚木时出现以下netcwd相关错误该怎么外理?

shell-init: error retrieving current directory: getcwd: cannot access parent directorie
s: No such file or directory Error occurred during initialization of VM java.lang.Error
: Properties init: Could not determine current working directory. at java.lang.System.i
nitProperties(Native Method) at java.lang.System.initializeSystemClass(System.java:1119
)

可能原因是执行脚本过程中误删了当前目录。解决办法为:先执行 cd ,然后重新运行脚本。

 使用一键接入方式安装探针后,在哪里查看日志?
 日志的默认目录为: /root /.arms/supervisor/logs/arms-supervisor.log,若此目录下没有日志,请执行 命令 ps -ef |grep arms 查看日志所在目录。

相关文档

• 应用监控常见问题

2.2.3. 为函数计算中的Java应用安装探针

只需安装ARMS应用监控组件(探针),即可对部署在函数计算中的Java应用进行监控,查看应用拓扑、接口调用、异常事务和慢事务等方面的监控数据。本文介绍如何为函数计算中的Java应用安装探针。

前提条件

使用控制台创建函数

步骤一:获取License Key

- 1. 登录ARMS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选择目标地域,然后单击接入应用。
- 3. 在接入中心面板单击Java,然后在STEP2区域获取License Key。



步骤二:配置函数变量

- 1. 登录函数计算控制台。
- 2. 在顶部菜单栏,选择地域。
- 3. 在左侧导航栏中,单击服务及函数。
- 4. 在服务列表区域单击目标服务。
- 5. 在函数管理页面找到目标函数,单击操作列的配置。
- 6. 在编辑函数配置页面的环境变量区域选择使用表单编辑,然后单击+添加变量。

7. 设置变量为*FC_EXTENSIONS_ARMS_LICENSE_KEY*, 值为步骤一: 获取License Key中获取的License Key, 然后单击**保存**。

结果验证

在函数计算控制台多次调用该函数后,若Java应用出现在ARMS控制台的**应用列表**页面中且有数据上报,则 说明接入成功。

相关文档

• 应用监控常见问题

2.2.4. 为容器服务Kubernetes版Java应用安装探针

您只需安装ARMS应用监控组件(探针)ack-onepilot,即可监控部署在容器服务Kubernetes版中的Java应用,并查看应用拓扑、接口调用、异常事务和慢事务等相关监控数据。本文介绍如何为容器服务Kubernetes版Java应用安装探针。

前提条件

- 创建Kubernetes集群。您可按需选择创建Kubernetes专有版集群、创建Kubernetes托管版集群或创建 Serverless Kubernetes集群。
- 创建命名空间,具体操作,请参见管理命名空间。本文示例中的命名空间名称为 arms-demo。
- 检查您的JDK版本。ARMS应用监控支持的JDK版本如下:
 - JDK 1.7.0+
 - JDK 1.8.0_25+
 - ? 说明
 - Kubernetes集群应用部署建议: JDK 1.8.0_191+。
 - 如果JDK版本为1.8.0_25或者1.8.0_31,可能会出现无法安装探针的情况,请升级至1.8.X最新版本。
 - JDK 11.0.8+

② 说明 JDK 1.8及以下版本和JDK 11版本对应的探针安装包不同,请根据不同的JDK版本下载对应的 探针安装包或调整应用监控组件ack-onepilot的配置。

步骤一:安装ARMS应用监控组件

- 1. 登录容器服务管理控制台。
- 在左侧导航栏选择市场 > 应用市场,在右侧应用目录页签通过关键字搜索ack-onepilot,然后单击 目标卡片。
- 3. 在ack-onepilot页面上,单击右上角的一键部署。
- 4. 在创建面板中选择安装组件的集群、命名空间,并输入组件发布名称,然后单击下一步。
- 5. 单击确定。

步骤二:授予ARMS资源的访问权限

- 如果需监控ASK (Serverless Kubernetes) 或对接了ECI的集群应用,请在云资源访问授权页面完成授权, 然后重启ack-onepilot组件下的所有Pod。
- 如果需监控ACK集群应用,但ACK集群中不存在ARMS Addon Token,请执行以下操作手动为集群授予

ARMS资源的访问权限。

? 说明

查看是否存在ARMS Addon Token的操作,请参见查看集群是否存在ARMS Addon Token。 集群存在ARMS Addon Token时,ARMS会进行免密授权。Kubernetes托管版集群默认存在ARMS Addon Token,但对于部分早期创建的Kubernetes托管版集群,可能会存在没有ARMS Addon Token 的情况,因此,对于Kubernetes托管版集群,建议首先检查ARMS Addon Token是否存在。若不存 在,需进行手动授权。

- i. 登录容器服务管理控制台。
- ii. 在左侧导航栏选择集群,在集群列表页面,单击目标集群名称或其右侧操作列的详情。

iii. 在目标集群的集群信息页面上单击集群资源页签,然后单击Worker RAM角色右侧的链接。

	asian • 🕑 air	Q HERCER, HERC, APL, MOLTHRING 2011 2011 2011 2011 2011 2011 2011 201
<	集群:	Riff Bitz Courses (1988)
集群结果		
节点管理	965 88°°136 409036 900746	
书点法	进步编辑 ROS	
	虚拟带有网络 VPC	vpc-user the second
工作负载	节点虚拟交换机	A2M- 1999
服务	安全相	19-10-1-1-1-1-1-1-1-1-1-1-1-1-1-1-1-1-1-
路由	Worker RAM 角色	KubernetesWorkerRole
	Noine Indress SI 8	b min i linking

- iv. 在访问控制RAM控制台的RAM角色管理页面上,单击权限管理页签上的权限策略名称。
- v. 在策略内容页签上单击修改策略内容。
- vi. 在脚本编辑页签中添加以下内容,然后单击下一步。

{	
	"Action": "arms:*"
	"Resource": "*",
	"Effect": "Allow"
1	

•	← 编辑权限策略			
	6 %E	在使用全新设计的权限策略编揭器,新版致力于让自定义策略更易懂易用。我们非常重视您的使用反馈,请告诉我们您的想法。		
	可视化编	编 Beta 脚本编辑		
	策略文档	长度 924 个字符		
	30	"Effect": "Allow"		
	31			
	33 •	"Action": [
	34	"w fathether i set i all dea",		
	35	"w Listhetweihigeist".		
	36	"m FullBapenitory".		
	37	"a fielder av Ppfloheint"		
	38			
	39 -	Kesource : L		
	40			
1	42	"Effect": "Allow"		
	43	b		
	44 -	{		
	45	"Action": "arms:*",		
	46	Resource": "#",		
	47	"Effect": "Allow"		
	48			
	50 }			
	下一步	超回		

vii. 确认策略内容无误后,单击确定。

步骤三:为Java应用开启ARMS应用监控

如需在创建新应用的同时开启ARMS应用监控,请完成以下操作。

- 1. 在容器服务Kubernetes版控制台左侧导航栏单击集群,在集群列表页面上的目标集群右侧操作列单击应用管理。
- 2. 在无状态页面右上角单击使用YAML创建资源。
- 3. 选择示例模板,并在模板(YAML格式)中将以下 labels 添加到 spec.template.metadata层级下。

label arm arm 的应用 one	ls: msPilotAutoEnable: "on" msPilotCreateAppName: " <your-deployment-name>" #请将<your-deployment-name>替换为您 月名称。 e-agent.jdk.version: "OpenJDK11" #如果应用的JDK版本是JDK 11,则需要配置此参数。</your-deployment-name></your-deployment-name>
示例模板	Resource - basic Deployment ~
模板	<pre>1 apiVersion: v1 2 kind: Namespace 3 - metadata: anme: arms-demo 5 6 apiVersion: apps/v1 # for versions before 1.8.0 use apps/v1betal 7 kind: Deployment 8 - metadata: 9 name: arms-springboot-demo 10 namespace: arms-demo 11 - Labels: 12 app: arms-springboot-demo 13 - spec: 14 replicas: 2 15 - selector: 16 app: arms-springboot-demo 18 - template: 19 metadata; 10 app: arms-springboot-demo 18 - template: 19 metadata; 20 armsPilotAutoEnaple: "on" 21 app: arms-springboot-demo 22 armsPilotAutoEnaple: "on" 23 armsPilotAutoEnaple: "on" 24 armsPilotAutoEnaple: "on" 25 armsPilotAutoEnaple: "on" 26 armsPilotAutoEnaple: "on" 27 armsPilotAutoEnaple: "on" 28 armsPilotAutoEnaple: "on" 29 armsPilotAutoEnaple: "on" 20 armsPilotAutoEnaple: "on" 21 armsPilotAutoEnaple: "on" 22 armsPilotAutoEnaple: "on" 23 armsPilotAutoEnaple: "on" 24 armsPilotAutoEnaple: "on" 25 armsPilotAutoEnaple: "on" 26 armsPilotAutoEnaple: "on" 27 armsPilotAutoEnaple: "on" 28 armsPilotAutoEnaple: "on" 29 armsPilotAutoEnaple: "on" 20 armsPilotAutoEnaple: "on" 20 armsPilotAutoEnaple: "on" 21 armsPilotAutoEnaple: "on" 23 armsPilotAutoEnaple: "on" 24 armsPilotAutoEnaple: "on" 25 armsPilotAutoEnaple: "on" 26 armsPilotAutoEnaple: "on" 27 armsPilotAutoEnaple: "on" 28 armsPilotAutoEnaple: "on" 29 armsPilotAutoEnaple: "on" 20 armsPilotAutoEnaple: "on" 21 armsPilotAutoEnaple: "on" 22 armsPilotAutoEnaple: "on" 23 armsPilotAutoEnaple: "on" 24 armsPilotAutoEnaple: "on" 25 armsPilotAutoEnaple: "on" 26 armsPilotAutoEnaple: "on" 27 armsPilotAutoEnaple: "on" 28 armsPilotAutoEnaple: "on" 29 armsPilotAutoEnaple: "on" 20 armsPilotAutoEnaple: "on" 20 armsPilotAutoEnaple: "on" 20 armsPilotAutoEnaple: "on" 20 armsPilotAutoEnaple: "on" 21 armsPilotAutoEnaple: "on" 22 armsPilotAutoEnaple: "on" 23 armsPilotAutoEnaple: "on" 24 armsPilotAutoEnaple: "on" 25 armsPilotAutoEnaple: "on" 26 armsPilotAutoEnaple: "on" 27 armsPilotAutoEnaple: "</pre>

创建一个无状态(Deployment)应用并开启ARMS应用监控的完整YAML示例模板如下:

执行结果

在无状态页面上,目标应用的操作列将出现ARMS控制台按钮。

请输入查询内容	٩				使用镜像创建		使用模板创建	刷新
□ 名称		标签	容器组数量	镜像	创建时间			攝作
nginx-		app:nginx	0/2	nginx:1.7.9	2020-09-22 09:51:07	详情	編辑 伸縮 ARMS控制	监控 台 更多▼

卸载探针

- 1. 登录容器服务管理控制台。
- 2. 在左侧导航栏单击集群,在集群列表页面上的目标集群右侧操作列单击应用管理。
- 3. 在左侧导航栏选择应用 > Helm。
- 4. 在Helm页面, ARMS Agent对应的发布名称ack-onepilot右侧操作列, 单击删除。
- 5. 在删除应用对话框单击确定。
- 6. 重启您的业务Pod。

常见问题

如何更改应用名称

如需更改应用名称,您需要修改Deployment内的armsPilotCreateAppName参数。具体操作,请参见部署在 容器服务K8s集群中的Java应用如何更改应用名称。

为什么Agent安装失败?

可能原因:目标集群中不存在ARMS Addon Token。

解决方案:查看集群是否存在ARMS Addon Token并为集群授予ARMS资源的访问权限。具体操作,请参见容器服务K8s集群中的应用安装Agent失败怎么处理?。

如何跨区域上报数据?

在容器集群arms-pilot Namespace下的Deployment中添加ARMS_REPORT_REGION环境变量,值为ARMS所 支持的RegionId(例如cn-hangzhou、cn-beijing),然后重启现有应用。更多信息,请参见ACK集群如何跨区 域上报数据?。

相关文档

- 创建Kubernetes专有版集群
- 管理命名空间
- 应用监控常见问题

2.2.5. 为开源Kubernetes环境中的应用安装探针

借助ARMS应用监控,您可以对开源Kubernetes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。本文将帮助您将开源Kubernetes环境中的应用接入ARMS应用监控。

前提条件

- 确保您的Kubernetes api-server组件接口版本在1.10及以上。
- 确保您的集群连通公网。
- 检查您的JDK版本。ARMS应用监控支持的JDK版本如下:
 - JDK 1.7.0+
 - JDK 1.8.0_25+

? 说明

- Kubernetes集群应用部署建议: JDK 1.8.0_191+。
- 如果JDK版本为1.8.0_25或者1.8.0_31,可能会出现无法安装探针的情况,请升级至1.8.X最新版本。

• JDK 11.0.8+

⑦ 说明 JDK 1.8及以下版本和JDK 11版本对应的探针安装包不同,请根据不同的JDK版本下载对应的 探针安装包或调整应用监控组件ack-onepilot的配置。

方式一:通过ACK注册集群接入ARMS

- 1. 将开源Kubernetes接入ACK One。具体操作,请参见创建注册集群并接入本地数据中心集群。
- 2. 为注册集群安装ack-arms-pilot组件。具体操作,请参见将应用实时监控服务ARMS接入注册集群。

方式二:将开源Kubernetes直接接入ARMS

步骤1: 安装Helm3

步骤2:安装探针

ARMS应用监控目前仅支持无状态(Deployment)和有状态(StatefulSet)两种类型的应用接入,两种类型的应用接入方法相同。此处以将开源Kubernetes环境中的无状态(Deployment)类型的应用接入ARMS应用监控为例。

1. 执行以下 wget 命令下载ack-onepilot安装包。

```
wget 'https://arms-apm-cn-hangzhou.oss-cn-hangzhou.aliyuncs.com/ack-onepilot/ack-onepi
lot-2.0.6.tgz'
```

2. 执行以下命令解压ack-onepilot安装包。

tar xvf ack-onepilot-2.0.6.tgz

3. 编辑安装包下的 values. yaml 文件,根据实际情况修改以下参数,然后保存。

```
image: registry-vpc.__ACK_REGION_ID__.aliyuncs.com/ack-onepilot/ack-onepilot
cluster_id: __ACK_CLUSTER_ID__
accessKey: __ACCESSKEY__
accessKeySecret: __ACCESSKEY_SECRET__
uid: "__ACK_UID__"
region_id: __ACK_REGION_ID__
```

• image : ack-onepilot镜像地址。

⑦ 说明 以上示例为VPC环境镜像地址,公网环境镜像获取地址如下:

registry. ACK REGION ID .aliyuncs.com/ack-onepilot/ack-onepilot

- ACK REGION ID : 阿里云地域ID,应用监控支持的地域,请参见应用监控目前支持的地域。
- _ACK_CLUSTER_ID__: 自定义Kubernetes集群ID,集群的唯一标识。建议格式为 <uid>-<cluste rid>。
- ____ACCESSKEY___和 ___ACCESSKEY_SECRET___: 阿里云账号的AccessKey ID和AccessKey Secret。
 获取方法,请参见获取AccessKey。
- __ACK_UID__: 阿里云账号ID。将鼠标悬浮于阿里云控制台右上角的头像上可以获取。
- 4. 执行以下命令安装ack-onepilot。

helm3 upgrade --install ack-onepilot ack-onepilot --namespace ack-onepilot --create-nam
espace

步骤3:获取License Key

- 1. 登录ARMS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选择目标地域,然后单击接入应用。
- 3. 在接入中心面板单击Java,然后在STEP2区域获取License Key。

<i>STEP2</i> ● 安装 Agent	1. 进入Agent压缩包新在目录并将其解压至任意工作目录下。	
	unzip ArmsAgent.zip -d /(user.workspace)/	٥
	● "(user.workspace)"是示例描述,此处及以下均谱相属具体环境普遍方正确的描述。	
	2. 滚动AppName以及LicenseKey参数(两种方法任选其一)	
	方法一: 観剣以下IVM巻載、添加到田用観祭的田式御本中。 屈用名称: my-service	
	- favaagent://user.workspace//Armsigent/arms-bootstrap-1.7.0-SNAPSHOT_jar Darms.licenser.pito -Darms.appName=my-service	۵
	 amm.ilcensetky,是我们为您自动生成的。 amms.appName是您的应用名称,请请 my-service 整独成您自己的应用名 	
	方法二:修改ams-agent.config,譬映ams.licenseKey汲ams.appName配置定义。	
	arms.licensexcy=b5	٥

步骤4:修改应用的YAML文件

1. 执行以下命令查看目标无状态(Deployment)应用的YAML文件。

kubectl get deployment {deploymenta b} -o yaml

⑦ 说明 若您不清楚 {deployment名称},请先执行以下命令查看所有无状态(Deployment)应用,在执行结果中找到目标无状态(Deployment)应用,再查看目标无状态(Deployment)应用的YAML文件。

kubectl get deployments --all-namespace

2. 启动编辑目标无状态(Deployment)应用的YAML文件。

kubectl edit deployment {Deployment名称} -o yaml

3. 在YAML文件中的 spec.template.met adata 层级下添加以下内容。

```
ARMSApmAppName: xxx
ARMSApmLicenseKey: xxx
one-agent.jdk.version: "OpenJDK11" //如果应用的JDK版本是JDK11,则需要配置此参数。
```

↓ 注意

○ 请将 xxx 分别替换成您的License Key和应用名称,应用名暂不支持中文。

○ 将License Key中的符号 @ 替换为符号 。

如果您需要在开源K8s环境中创建一个新的无状态(Deployment)应用并接入ARMS,则应用的完整 YAML文件如下:

保存配置后,应用将自动重启,以上配置生效。
 2~5分钟后,若您的应用出现在ARMS控制台的应用监控 > 应用列表页面中且有数据上报,则说明接入成功。

卸载探针

当您不需要再监控开源Kubernetes环境中的Java应用时,可以卸载ack-onepilot。

1. 执行以下命令卸载ack-onepilot。

helm3 uninstall --namespace ack-onepilot ack-onepilot

2. 重启您的应用Pod。

相关文档

• 应用监控常见问题

2.2.6. 为Docker中的Java应用安装Agent

为Docker中的Java应用安装ARMS Agent后,ARMS即可开始监控Java应用,且ARMS将自动适配该应用运行的环境,不需要针对Tomcat、Jetty或Spring Boot等应用单独配置运行环境。本文介绍如何为Docker中的 Java应用安装Agent。

前提条件

- 您已在Docker中部署Java应用。
- 检查您的JDK版本。ARMS应用监控支持的JDK版本如下:
 - JDK 1.7.0+
 - JDK 1.8.0_25+

? 说明

- Kubernetes集群应用部署建议: JDK 1.8.0_191+。
- 如果JDK版本为1.8.0_25或者1.8.0_31,可能会出现无法安装探针的情况,请升级至1.8.X最新版本。
- JDK 11.0.8+

⑦ 说明 JDK 1.8及以下版本和JDK 11版本对应的探针安装包不同,请根据不同的JDK版本下载对应的 探针安装包或调整应用监控组件ack-onepilot的配置。

背景信息

对于Java应用镜像*{original-docker-image:tag}*,可以通过编辑*Dockerfile*文件来集成已有镜像,然后构建和 启动新的镜像,即可将Java应用接入ARMS应用监控。

步骤一:获取License Key

- 1. 登录ARMS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选择目标地域,然后单击接入应用。
- 3. 在接入中心面板单击Java,然后在STEP2区域获取License Key。



步骤二:集成已有镜像

参考以下 Dockerfile示例修改您的 Dockerfile文件。

```
##
                          ###
##
     ARMS APM DEMO Docker
                          ###
       For Java
##
                          ###
##
     withAgent V0.1
                         ###
##
                          ###
# 将{original-docker-image:tag}替换为您的镜像地址。
FROM {original-docker-image:tag}
WORKDIR /root/
# 根据所在地域替换Agent的下载地址。
RUN wget "http://arms-apm-cn-hangzhou.oss-cn-hangzhou.aliyuncs.com/ArmsAgent.zip" -O ArmsAg
ent.zip
RUN unzip ArmsAgent.zip -d /root/
# 按照步骤一的说明获取License Key。
# {AppName}为自定义ARMS监控应用名称,不可包含中文字符。
# 若所有镜像都接入同一个应用监控任务,配置此处的arms licenseKey和arms appName即可。
ENV arms licenseKey={LicenseKey}
ENV arms appName={AppName}
ENV JAVA TOOL OPTIONS ${JAVA TOOL OPTIONS} '-javaagent:/root/ArmsAgent/arms-bootstrap-1.7.0
-SNAPSHOT.jar -Darms.licenseKey='${arms licenseKey}' -Darms.appName='${arms appName}
### for check the args
RUN env | grep JAVA TOOL OPTIONS
### 可在下方添加自定义Dockerfile逻辑。
### .....
```

请按照以下说明替换上述配置文件中的示例值。

- 将 {original-docker-image:tag} 替换为您的镜像地址。若您没有自定义镜像, 可使用系统镜像。
- 根据所在地域替换Agent的下载地址。

⑦ 说明 请使用公网地址,如无法下载则使用VPC地址。

查看各地域对应的Agent(JDK 8及以下版本)安装包下载命令

应用实时监控服务 ARMS

应用监控·接入应用监控

地域	公网地址	VPC地址
华东1(杭州)	wget "http://arms-apm-cn- hangzhou.oss-cn- hangzhou.aliyuncs.com/ArmsAgen t.zip" -O ArmsAgent.zip	wget "http://arms-apm-cn- hangzhou.oss-cn-hangzhou- internal.aliyuncs.com/ArmsAgen t.zip" -O ArmsAgent.zip
华东2(上海)	wget "http://arms-apm-cn- shanghai.oss-cn- shanghai.aliyuncs.com/ArmsAgen t.zip" -O ArmsAgent.zip	wget "http://arms-apm-cn- shanghai.oss-cn-shanghai- internal.aliyuncs.com/ArmsAgen t.zip" -O ArmsAgent.zip
华北1(青岛)	wget "http://arms-apm-cn- qingdao.oss-cn- qingdao.aliyuncs.com/ArmsAgent .zip" -O ArmsAgent.zip	wget "http://arms-apm-cn- qingdao.oss-cn-qingdao- internal.aliyuncs.com/ArmsAgen t.zip" -O ArmsAgent.zip
华北2(北京)	wget "http://arms-apm-cn- beijing.oss-cn- beijing.aliyuncs.com/ArmsAgent .zip" -O ArmsAgent.zip	wget "http://arms-apm-cn- beijing.oss-cn-beijing- internal.aliyuncs.com/ArmsAgen t.zip" -O ArmsAgent.zip
华北3(张家口)	wget "http://arms-apm-cn- zhangjiakou.oss-cn- zhangjiakou.aliyuncs.com/ArmsA gent.zip" -O ArmsAgent.zip	wget "http://arms-apm-cn- zhangjiakou.oss-cn- zhangjiakou- internal.aliyuncs.com/ArmsAgen t.zip" -0 ArmsAgent.zip
华北5(呼和浩特)	wget "http://arms-apm-cn- huhehaote.oss-cn- huhehaote.aliyuncs.com/ArmsAge nt.zip" -0 ArmsAgent.zip	wget "http://arms-apm-cn- huhehaote.oss-cn-huhehaote- internal.aliyuncs.com/ArmsAgen t.zip" -0 ArmsAgent.zip
华北6(乌兰察布)	wget "http://arms-apm-cn- wulanchabu.oss-cn- wulanchabu.aliyuncs.com/ArmsAg ent.zip" -O ArmsAgent.zip	wget "http://arms-apm-cn- wulanchabu.oss-cn-wulanchabu- internal.aliyuncs.com/ArmsAgen t.zip" -O ArmsAgent.zip

应用监控·接入应用监控

应用实时监控服务 ARMS

地域	公网地址	VPC地址
华南1(深圳)	wget "http://arms-apm-cn- shenzhen.oss-cn- shenzhen.aliyuncs.com/ArmsAgen t.zip" -0 ArmsAgent.zip	wget "http://arms-apm-cn- shenzhen.oss-cn-shenzhen- internal.aliyuncs.com/ArmsAgen t.zip" -O ArmsAgent.zip
华南2(河源)	wget "http://arms-apm-cn- heyuan.oss-cn- heyuan.aliyuncs.com/ArmsAgent. zip" -O ArmsAgent.zip	wget "http://arms-apm-cn- heyuan.oss-cn-heyuan- internal.aliyuncs.com/ArmsAgen t.zip" -O ArmsAgent.zip
华南3(广州)	wget "http://arms-apm-cn- guangzhou.oss-cn- guangzhou.aliyuncs.com/ArmsAge nt.zip" -O ArmsAgent.zip	wget "http://arms-apm-cn- guangzhou.oss-cn-guangzhou- internal.aliyuncs.com/ArmsAgen t.zip" -O ArmsAgent.zip
西南1(成都)	wget "http://arms-apm-cn- chengdu.oss-cn- chengdu.aliyuncs.com/ArmsAgent .zip" -O ArmsAgent.zip	wget "http://arms-apm-cn- chengdu.oss-cn-chengdu- internal.aliyuncs.com/ArmsAgen t.zip" -O ArmsAgent.zip
中国(香港)	wget "http://arms-apm-cn- hongkong.oss-cn- hongkong.aliyuncs.com/ArmsAgen t.zip" -O ArmsAgent.zip	wget "http://arms-apm-cn- hongkong.oss-cn-hongkong- internal.aliyuncs.com/ArmsAgen t.zip" -O ArmsAgent.zip
亚太东南1(新加 坡)	<pre>wget "http://arms-apm-ap- southeast-1.oss-ap-southeast- 1.aliyuncs.com/ArmsAgent.zip" -0 ArmsAgent.zip</pre>	<pre>wget "http://arms-apm-ap- southeast-1.oss-ap-southeast- 1- internal.aliyuncs.com/ArmsAgen t.zip" -0 ArmsAgent.zip</pre>

应用实时监控服务 ARMS

应用监控·接入应用监控

地域	公网地址	VPC地址
亚太东南2(悉尼)	<pre>wget "http://arms-apm-ap- southeast-2.oss-ap-southeast- 2.aliyuncs.com/ArmsAgent.zip" -0 ArmsAgent.zip</pre>	<pre>wget "http://arms-apm-ap- southeast-2.oss-ap-southeast- 2- internal.aliyuncs.com/ArmsAgen t.zip" -0 ArmsAgent.zip</pre>
亚太东南3(吉隆 坡)	<pre>wget "http://arms-apm-ap- southeast-3.oss-ap-southeast- 3.aliyuncs.com/ArmsAgent.zip" -0 ArmsAgent.zip</pre>	wget "http://arms-apm-ap- southeast-3.oss-ap-southeast- 3- internal.aliyuncs.com/ArmsAgen t.zip" -O ArmsAgent.zip
亚太东南5(雅加 达)	wget "http://arms-apm-ap- southeast-5.oss-ap-southeast- 5.aliyuncs.com/ArmsAgent.zip" -0 ArmsAgent.zip	<pre>wget "http://arms-apm-ap- southeast-5.oss-ap-southeast- 5- internal.aliyuncs.com/ArmsAgen t.zip" -0 ArmsAgent.zip</pre>
亚太东北1(东京)	<pre>wget "http://arms-apm-ap- northeast-1.oss-ap-northeast- 1.aliyuncs.com/ArmsAgent.zip" -0 ArmsAgent.zip</pre>	<pre>wget "http://arms-apm-ap- northeast-1.oss-ap-northeast- 1- internal.aliyuncs.com/ArmsAgen t.zip" -0 ArmsAgent.zip</pre>
欧洲中部1(法兰克 福)	wget "http://arms-apm-eu- central-1.oss-eu-central- 1.aliyuncs.com/ArmsAgent.zip" -0 ArmsAgent.zip	wget "http://arms-apm-eu- central-1.oss-eu-central-1- internal.aliyuncs.com/ArmsAgen t.zip" -O ArmsAgent.zip
欧洲西部1(伦敦)	<pre>wget "http://arms-apm-eu-west- 1.oss-eu-west- 1.aliyuncs.com/ArmsAgent.zip" -0 ArmsAgent.zip</pre>	wget "http://arms-apm-eu-west- 1.oss-eu-west-1- internal.aliyuncs.com/ArmsAgen t.zip" -O ArmsAgent.zip

应用监控·接入应用监控

应用实时监控服务 ARMS

地域	公网地址	VPC地址
美国东部1(弗吉尼 亚)	wget "http://arms-apm-us-east- 1.oss-us-east- 1.aliyuncs.com/ArmsAgent.zip" -O ArmsAgent.zip	wget "http://arms-apm-us-east- 1.oss-us-east-1- internal.aliyuncs.com/ArmsAgen t.zip" -O ArmsAgent.zip
美国西部1(硅谷)	<pre>wget "http://arms-apm-us-west- l.oss-us-west- l.aliyuncs.com/ArmsAgent.zip" -0 ArmsAgent.zipwget "http://arms-apm-us-west- l.oss-us-west-1- internal.aliyuncs.com/ArmsAgen t.zip" -0 ArmsAgent.zip</pre>	wget "http://arms-apm-us-west- 1.oss-us-west-1- internal.aliyuncs.com/ArmsAgen t.zip" -O ArmsAgent.zip
亚太南部1(孟买)	<pre>wget "http://arms-apm-ap- south-1.oss-ap-south- 1.aliyuncs.com/ArmsAgent.zip" -0 ArmsAgent.zip</pre>	wget "http://arms-apm-ap- south-1.oss-ap-south-1- internal.aliyuncs.com/ArmsAgen t.zip" -O ArmsAgent.zip
华东1金融云	无	wget "http://arms-apm-cn- hangzhou-finance.oss-cn-hzjbp- b- internal.aliyuncs.com/ArmsAgen t.zip" -O ArmsAgent.zip
华东2金融云	无	<pre>wget "http://arms-apm-cn- shanghai-finance-1.oss-cn- shanghai-finance-1- internal.aliyuncs.com/ArmsAgen t.zip" -0 ArmsAgent.zip</pre>
华南1金融云	无	<pre>wget "http://arms-apm-cn- shenzhen-finance-1.oss-cn- shenzhen-finance-1- internal.aliyuncs.com/ArmsAgen t.zip" -0 ArmsAgent.zip</pre>

地域	公网地址	VPC地址
政务云	<pre>wget "http://arms-apm-cn- north-2-gov-1.oss-cn-north-2- gov- 1.aliyuncs.com/ArmsAgent.zip" -0 ArmsAgent.zip</pre>	wget "http://arms-apm-cn- north-2-gov-1.oss-cn-north-2- gov-1- internal.aliyuncs.com/ArmsAgen t.zip" -O ArmsAgent.zip

● 将 {LicenseKey} 替换成您的License Key。将 {AppName} 替换成您的应用名称(应用名不可包含中文 字符)。

步骤三:构建并启动新镜像

1. 运行 docker build 命令来构建镜像。

docker build -t registry.cn-hangzhou.aliyuncs.com/arms-docker-repo/arms-springboot-demo
:v0.1 -f /{workspace}/Dockerfile /{workspace}/

⑦ 说明 将 registry.cn-hangzhou.aliyuncs.com/arms-docker-repo/arms-springboot-demo:v
 0.1 替换为真实镜像名称。

2. 运行 docker run 命令来启动镜像。

docker run -d -p 8081:8080 registry.cn-hangzhou.aliyuncs.com/arms-docker-repo/arms-spri
ngboot-demo:v0.1

⑦ 说明 将 registry.cn-hangzhou.aliyuncs.com/arms-docker-repo/arms-springboot-demo:v
 0.1 替换为真实镜像名称。

结果验证

约一分钟后,若您的应用名称出现在应用列表页面中且有数据上报,则说明接入成功。

卸载Agent

当您不需要再监控Docker集群中的Java应用时,请按照以下步骤卸载Agent。

- 1. 删除按照步骤二:集成已有镜像的说明添加的Dockerfile内容。
- 2. 运行 docker build 命令来构建镜像。
- 3. 运行 docker run 命令来启动镜像。

相关文档

• 应用监控常见问题

2.2.7. 通过OpenTelemetry 接入ARMS

本文介绍如何将OpenTelemetry Trace数据接入ARMS并使用。

接入OpenTelemetry Trace数据

ARMS支持多种方式接入OpenTelemetry Trace数据,您可以将OpenTelemetry Trace数据直接上报至 ARMS,或通过OpenTelemetry Collector转发。

- 通过ARMS Java Agent 接入OpenTelemetry Trace数据 Java应用推荐安装ARMS Java Agent。ARMS Java Agent内置了大量通用组件的链路埋点,能够自动上报 OpenTelemetry格式的Trace数据,开箱即用,无需额外配置。具体操作,请参见监控Java应用。
- 结合ARMS Java Agent与OpenTelemetry Java SDK上报Trace数据 v2.7.1.3及以上版本的ARMS Java Agent支持OpenTelemetry Java SDK扩展。您在使用ARMS Java Agent自 动获取通用组件Trace数据的同时,还可以通过OpenTelemetry SDK扩展自定义的方法埋点。具体操作, 请参见通过OpenTelemetry Java SDK进行手工埋点。
- 通过OpenTelemetry直接上报Trace数据 您也可以使用OpenTelemetry SDK进行应用埋点,并通过Jaeger Exporter直接上报Trace数据。具体操 作,请参见通过OpenTelemetry上报Java应用数据。

通过ARMS for OpenTelemetry Collector转发Trace数据

在容器服务ACK环境下,您可以一键安装ARMS for OpenTelemetry Collector,通过它进行Trace数据的转发。ARMS for OpenTelemetry Collector实现了链路无损统计(本地预聚合,统计结果不受采样率影响), 动态配置调参,状态管理以及开箱即用的Trace Dashboard on Grafana,同时更加易用、稳定、可靠。

ARMS for OpenTelemetry Collector的接入流程如下:

- 1. 通过ACK控制台应用目录安装ARMS for OpenTelemetry Collector。
 - i. 登录容器服务管理控制台。
 - ii. 在左侧导航栏选择市场 > 应用市场。
 - iii. 在应用市场页面通过关键字搜索ack-arms-cmonitor组件,然后单击ack-arms-cmonitor。
 - iv. 在ack-arms-cmonitor页面单击右上角的一键部署。
 - v. 在创建面板中,选择目标集群,然后单击下一步。

? 说明 命名空间默认为arms-prom。

- vi. 单击确定。
- vii. 在左侧导航栏单击集群,然后单击刚刚安装了ack-arms-cmonitor组件的集群名称。
- viii. 在左侧导航栏选择工作负载 > 守护进程集,在页面顶部选择命名空间为arms-prom。

ix. 单击otel-collector-service。

查看otel-collector-service(Service)是否正常运行,如下图所示对外暴露了多种Receivers端口接收OpenTelemetry数据,则表示安装成功。

← otel-collecto	r-service				
Basic Information					
Name:	otel-collector-service				
Namespace:	arms-prom				
Created At:	Feb 10, 2022, 16:59:46 UTC+8				
Labels:	kubernetes.io/service-name:otel-collector-service release:RELEASE-NAME heritage:Helm otel-collector-app:otel-collector chart.ack-arms-cmonitor-1.0.0				
Annotations:					
Туре:	ClusterIP				
Cluster IP:	172.16.				
Internal Endpoint:	otel-collector-service:4317 TCP otel-collector-service:4318 TCP otel-collector-service:4311 TCP otel-collector-service:14250 TCP otel-collector-service:14268 TCP				
External Endpoint:	-				

2. 修改应用SDK中的Exporter Endpoint地址为 otel-collector-service:Port 。

48	// Initializes an OTLP exporter, and configures the corresponding trace and
49	// metric providers.
50	<pre>func initProvider() func() {</pre>
51	<pre>ctx := context.Background()</pre>
52	
53	<pre>otelAgentAddr := "otel-collector-service:4317"</pre>
54	
55	<pre>metricClient := otlpmetricgrpc.NewClient(</pre>
56	otlpmetricgrpc.WithInsecure(),
57	otlpmetricgrpc.WithEndpoint(otelAgentAddr))
58	<pre>metricExp, err := otlpmetric.New(ctx, metricClient)</pre>
59	handleErr(err, "Failed to create the collector metric exporter")
60	
61	<pre>pusher := controller.New(</pre>
62	processor.NewFactory(
63	<pre>simple.NewWithExactDistribution(),</pre>
64	metricExp,
65),
66	controller.WithExporter(metricExp),
67	controller.WithCollectPeriod(2*time.Second),
68)
69	global.SetMeterProvider(pusher)
70	
71	err = pusher.Start(ctx)
72	handleErr(err, "Failed to start metric pusher")
73	
74	<pre>traceClient := otlptracegrpc.NewClient(</pre>
75	otlptracegrpc.WithInsecure(),
76	otlptracegrpc.WithEndpoint(otelAgentAddr),
77	otlptracegrpc.WithDialOption(grpc.WithBlock()))
78	<pre>traceExp, err := otlptrace.New(ctx, traceClient)</pre>
79	handleErr(err, "Failed to create the collector trace exporter")

通过开源OpenTelemetry Collector转发Trace数据

使用开源的OpenTelemetry Collector转发Trace数据至ARMS,只需要修改Exporter中的接入点 (Endpoint)和鉴权信息(Token)。

```
exporters:
  otlp:
    endpoint: <endpoint>:8090
    tls:
        insecure: true
    headers:
        Authentication: <token>
```

? 说明

- 将 <endpoint> 替换为您上报区域对应的Endpoint,例如: http://tracing-analysis-dc-bj
 .aliyuncs.com:8090 。
- 将 <token> 替换为您控制台上获取的Token,例如: b5901hguqs@3a7******9b_b5901hgu qs@53d****8301 。

直接上报

通过OpenTelemetry Collector转发

OpenTelemetry Trace使用指南

为了更好的发挥OpenTelemetry Trace数据价值,ARMS提供了链路详情、预聚合大盘、Trace Explorer后聚 合分析、调用链路关联业务日志等多种诊断能力。

● 链路详情

在链路详情面板左侧可以查看链路的接口调用次序与耗时,面板右侧展示了详细的附加信息和关联指标, 例如数据库SQL,JVM和Host监控指标等。

TraceID I	bo Start Time 2022-02-10 14:30:55	Total Time Consumed L1s		Application Name arms-console-hz X
		0110	1.%	API Name MySQL
* 27	nap	1.15		5984 IIM8 2022-02-10 14:0055-155
				Tag
	org apache.http:protocol.HttpRequestExecutor.execut arms-console-hz	02ma		
				Addisonal Phontsison Metho
	org apache.http.prztocel.HttpRequestExecutor.execut anna-console-tz	33ma		JVM Host
	org apache.http:protocol.Http/fequestExecutor.execut			Inductorsance GC Times
	ams-console hz	1910		
	14 Beste	and the second se		Tace
	avus-consule-hz	6ma		·
	WSQL			1
	ams-console-hz	210		
				14/14/00 14/17/00 14/24/00 14/27/00 14/34/00 14/07/00 14/44/00
	MyHOL	170		- Ed CO Times - YeuroCO Times
	MyRatis			
	ams-consile-tz	6ms		Instantaneous GC Duration
	14.177			Tate Tate
	wysu, awada kr	2110		
	MySQL			
	ams-console-hz	398		600000
	Weiterin			14:14:00 14:29:00 14:29:00 14:34:00 14:29:00 14:44:00
	arms-console-hz	0ms		- Ed. 92 Duration - Wear-92 Duration
	MySQ.			
	RESEARCH TA			oran menuli reaso
	MySQ.			Tace
	avvis-console-bz	4/85		
	org apacest. And pressess interviewed and concerns access	line .		and and and
	org apache.http.protocol.HttpRequestExecutor.execut			14.14.00 14.19.00 14.29.00 14.29.00 14.34.00 14.39.00 14.44.09
	arm-console-hz	503ma		- Young Generation Eden Space - Young Generation Survivor Space - Old Age - Total Usage
	Multiple			
	arm-comole-hz		4m	DBI Threads (B)
	My5Q.		2	7404
	ama-console-tz		47.00	126

• 预聚合大盘

ARMS基于OpenTelemetry Trace数据提供了多种预聚合指标大盘,包括应用总览,接口调用,数据库调用等。更多信息,请参见查看应用性能关键指标和拓扑图。



• Trace Explorer后聚合分析

针对OpenTelemetry Trace数据,ARMS提供了灵活的多维筛选与后聚合分析能力,例如查询特定应用的 异常链路。还可以根据IP、接口等维度对Trace数据进行聚合。更多信息,请参见Trace Explorer。

链路查询

Trace Exp	orer 0						最近30分钟	
900 955 E	District No.	statusCode=2 AND serviceName="apmZxTest"					0 Q C B	
快播碗法								A
状态~	次数 / 利約	span@t	HTTP管涡数			利时百分位		
<mark>⊻</mark> 异☆	3k *	120 16.35.00 70 0.000rt 12 80				40s 30s 20s 10s		∇
₩etins ~	~~	0 16:31:00 16:36:00 16:41:00 16:46:00 16:51:00 16:56:00 count	0 17.01.00 17.05.00 14.32.00 14.3 500	37:00 14:42:00 14:47:00 14:	52:00 14:57:00 17:02:00	0ms 14:31:00 14:34:00 14:41:00 p99 p10 p75 p50 avg_du	16:46:00 16:51:00 16:56:00 ation	17.01:00 17:05:00
Ons -	1m46s	搜索到3k展span						۲
应用名称~	次数 / 16时	traceld	startTime &	serviceName	spanName	duration a	ip	2017
Q 證意serviceNa	10 10 10 10 10 10 10 10 10 10 10 10 10 1	Objected Technology 2010 Control of Control	10/26 17:03:55	apmZxTest	comaliy	Oms	11.165.71.108	26
apmZxTest	3 🔺	Objective restoration and the restored	10/26 17:03:55	apmZvTest	Recy Topological	Oms	11.165.71.108	26
	*	Obstantia	10/26 17:03:54	apmZxTest	comaliy	Oms	11.145.71.108	26
€	•	Object to a state of the state of the state of the	10/26 17:03:54	apmZxTest	Recy Topological	Oms	11.165.71.108	26
接口名称 ~	次数 / 16时	Obelief the residue of the residue of the	10/26 17:03:54	apmZxTest	comaliy	1ms	11.165.71.108	26
Q 搜索spanName	2998998 VL	Object to the second	10/26 17:03:54	apmZxTest	Recy Topological	1ms	11.145.71.108	26
MySQL com alloun noen	2 ^	Obelie Technologie Technologie	10/26 17:03:44	apmZxTest	/demo/second	2.34	11.165.71.108	20
Dans Train Bank	ntun 🔹 👻	Obelie Technology and an and a second second	10/26 17:03:35	apmZxTest	MySQL	29ms	11.165.71.108	26
±101094 v	2787 / 8647	Obalia facilitati amazoni 17 apazona	10/26 17:03:34	apmZxTest	MySQL	36ms	11.165.71.108	
0.0000000000		Obelie Technic Service 1774-242-151	10/26 17:03:34	apmZxTest	MySQL	27ms	11.165.71.108	26
11.165.71.108	3	Observation and a second se	10/26 17:03:34	apmZvTest	MySQL	27ms	11.165.71.108	70
	ų	l ob	10/26 17:03:33	apmZvTest	MySQL	34ms	11.165.71.108	25
4	F.	Objective rests and an interaction	10/26 17:03:33	apmZvTest	MySQL	37ms	11.105.71.108	20

链路分析



• 调用链路关联业务日志

ARMS支持将OpenTelemetry Trace与业务日志相关联,从应用接口角度排查业务异常问题。更多信息, 请参见通过调用链路和日志分析定位业务异常问题。

= (-)阿里云				Q BREATS 19815, API, MIRISBOBE	現円	工単	發雲	企业 支	寺 直河	Ŀ.	٥	R (3) %	2
<	100.000	~ a												
2用总流														
如用洋情	已绑定Project:	Logstore:	and another in a 编辑之											
0.44	@ arms-console-all-lo	g		© 2020-09-09 14.06:00-2020 4	9 09 14 15	00 🕶	自动刷新	分單	童肉分析	1個性	另称为快	速度闪	另存	为吉
附件评值	✓ 1 0bc0594d1599631	788										00	26	VS
2展车混用		3 🔯 09-09 14:10.07		Num-omenculoppon										
oSQL调用 4555			tagpath tagreceive_sme:											
御湯用 4000				nio-8080-exec-1) WARN ic.a.a.c.service.prometheus.	Prometheus	ServiceIm	pi - promet	heus check i	nput param	success	from			
0/2/9		4 🔯 09-09 14:10.07	_source											
用沙断	/		_18gpeth											
(用)注重			tsgreceive_unte_: topic: () content: 2020-09-09 14:10:07.057.09bc0094d15995317881	tp-nio-8000-exec-1] INFO arms statistic registerPror	nCluster.log	per - IRegi	sterPromC	usterServiet	RegisterPn	omCluste	srServiel Fa	iled> pr	anuUseri	d:
			109637003), userid. 1833024105 , accessKey accessSector functionOptions: null, ret: java lang RuntimeException 台語Byrafema219年来死; j at service prometheus grafema GrafemaConfgemus at service prometheus grafema GrafemaConfgemus	I, regionid: on-beijing, clusteriid: claef73c98e3e4a olderName: : reheshVex/Bj.0tgld(GrafanaConfigService Java:223 instal/Bacek98Em(GrafanaConfigService Java:211)	7954aarimii)		pilotid cta	ef73c98e3e4	18795438041	288				
			at service prometheus grafana Grafana ConfigService at service prometheus PrometheusServiceimol addF	createGrafanaExporterDashboard(GrafanaConfigSe iporter[PrometheurServ(celm 當开)	rvice.java:8	7)								

2.3. 开始监控除Java之外的应用

对于Java之外的应用,例如C++、Go、Node.js、.NET等,您可以使用链路追踪Tracing Analysis来监控。链路追踪具备分布式调用链追踪和汇总、应用性能实时汇总和分布式拓扑动态发现等能力,帮助您全方位监控应用。

背景信息
链路追踪可以帮助开发者快速分析和诊断分布式应用架构下的性能瓶颈,提高微服务时代下的开发诊断效率。链路追踪具备以下能力:

- 分布式调用链追踪和汇总: 追踪分布式架构中的所有微服务用户请求, 并将它们汇总成分布式调用链。
- 应用性能实时汇总:通过追踪整个应用程序的用户请求,来实时汇总组成应用程序的单个服务和资源。
- 分布式拓扑动态发现:针对您的分布式微服务应用和相关PaaS产品,链路追踪均可收集到分布式调用信息。
- 开源社区兼容:基于OpenTracing标准,兼容Jaeger、Zipkin等开源产品。
- 下游场景对接: 收集的链路可直接用于日志分析, 且可对接到MaxCompute等下游分析平台。

监控多语言应用

更多信息

- 查看应用列表
- 查看应用性能关键指标和拓扑图
- 查看应用详情
- 查看接口调用情况
- 查询调用链
- 调用链分析
- 管理应用和标签

3.控制台功能

3.1. 应用监控3D拓扑图

ARMS应用监控3D拓扑图能立体展示应用、服务和主机的健康状况,以及应用的上下游依赖关系。借助3D拓 扑图,您可以快速定位诱发故障的服务、被故障影响的应用和关联的主机等,全方位地诊断故障根源,从而 快速排除故障。

快速入门

请查看以下交互式文档,了解ARMS应用监控3D拓扑图的功能。为获得最佳体验,请在新窗口中打开查看。

功能入口

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏选择应用监控 > 应用列表,并在顶部菜单栏选择目标地域。
- 3. 在应用列表页面单击目标应用操作列的3D拓扑。

总览层 (Overview)

在默认展示的Overview页面上,您可以看到服务层、应用层和主机层的全部内容。页面右上角显示的是主机、应用和服务的数量。



在总览层,您可以执行以下操作:

- 在页面左上角单击时间范围区域,然后在弹出的时间范围选择器内选择精确的起止时间。
- 在页面顶部的时间轴上, 随意拖动时间滑块来改变当前视图对应的时间范围。
- 在页面右上角的搜索框内, 输入关键字并按回车进行搜索。
- 用鼠标拖放,以任意角度查看三层数据。
- 单击视图中的任意对象,在右侧面板中查看该对象的相关指标。

服务层 (Service)

服务层展示应用所依赖的服务信息。



每个应用下的服务对应一个板块。调用次数越多,所占面积越大。服务的不同状态以不同颜色表示。

- : 服务调用正常■
- : 服务出错率较高■
- : 服务无返回数据■

⑦ 说明 服务的响应时长阈值是可以配置的。在左侧导航栏中单击服务层(Service)右侧的三角形图标,即可展开耗时阈值设置框。在该设置框中拖动滑块即可设置阈值。

单击一个服务后,右侧面板将展示该服务的以下信息:

- 服务名称
- QPS: Query Per Second (每秒查询数)
- RT(ms): Response Time (响应时间,单位为毫秒)
- Error: Error QPS (每秒错误查询数)

⑦ 说明 在QPS、RT (ms)和Error区域框中, 左侧的数值是所选时间范围内的平均值, 右侧是所选时间范围内的曲线图。

应用层(Application)

应用层展示应用及其上下游依赖关系,包括依赖的中间件在内。通过连接线的流向,您可以看到调用方向。



单击一个应用后,右侧面板将展示该应用的以下信息:

- 应用名称
- QPS: Query Per Second (每秒查询数)
- RT(ms): Response Time (响应时间,单位为毫秒)
- Error: Error QPS (每秒错误查询数)

⑦ 说明 在QPS、RT (ms)和Error区域框中, 左侧的数值是所选时间范围内的平均值, 右侧是所选时间范围内的曲线图。

主机层 (Docker/ECS)

主机层展示应用的主机详情。



每个方块代表一个主机。所有主机按应用分区。主机的不同状态以不同颜色表示。

- : 正常■
- ●:缓慢■
- ●:警告■
- ●:异常■
- ●:离线■

单击一个主机后,右侧面板将展示该主机的以下信息:

- 主机IP地址及基础信息:
 - 。 响应时间
 - 请求数
 - 错误数
- CPU: CPU使用率
- MEM: 内存使用率
- DISK: 磁盘使用率
- GC TIME: GC耗时
- GC COUNT: GC次数

⑦ 说明 在CPU、MEM和DISK区域框中,左侧的数值是所选时间范围内的平均值,右侧是所选时间范围内的曲线图。

3.2. 调用链路查询

在**调用链路查询**页面,您可以通过Traceld精确查询调用链路详细情况,或结合多种条件筛选查询调用链路。也可以对多条调用链进行聚合分析。

查询调用链路

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏选择应用监控 > 调用链路查询,并在顶部菜单栏选择目标地域。

 在调用链路查询页面选择参数类型,在参数值中填入自定义标签,单击添加到查询条件。 可以使用Traceld参数进行精确查询。 参数类型说明

参数类型	描述
Traceld	输入Traceld。
接口名称	应用调用的接口名称,不支持模糊搜索。
客户端应用名	客户端的应用名称。
服务端应用名	服务端的应用名称。
耗时大于	调用的耗时大于指定毫秒数。
调用类型	选择调用类型。
是否异常调用	所有包含异常调用的链路。
仅含线程剖析快照	所有包含线程剖析快照的链路。
客户端IP	调用发起应用IP。
服务端IP	请求被调用的应用IP。
业务主键	搜索业务事件所使用的字段。
响应码	输入响应码。

4. 单击需要查看的TracelD名称,进入调用链路页面。

调用链路 业务轨迹									
应用条称	日忠产生时间	状态	1P1834E	调用类型	服务名称	1850212	方法线@	线程创新	时间数 (ms)
▼ arms-k8s-c	2020-11-10 19:17:3.562	٠		НТТРА.	All second s	Ŀ	Q		505ms
arms-k	2020-11-10 19:17:3.563	٠	10.000	нттрд.	(Bridge (Dec.))	Ł	Q		503ms

调用链路页面字段说明如下:

- 应用名称:所属应用名称。
- 日志产生时间:日志产生的时间。
- 状态: 红色表示该服务调用的本地调用链路中存在异常,绿色表示正常。

- IP地址:该应用的IP地址。
- 。调用类型:该次调用的调用类型,与即席查询的调用类型选项对应。
- 服务名称: 该次调用的服务接口名称。
- 时间轴: 各服务间调用链路的耗时, 以及相对于整条调用链路的耗时分布。

分析调用链路

在**调用链路查询**页面选中所有需要分析的调用链路,单击**分析选中调用链路**。 在**调用链分析**面板可以查看所有选中调用链路的Span名称、应用名、调用类型、请求数/请求比例、异常 数/异常比例、平均自身耗时和平均耗时。

调用链分析									
Span名称	应用名	调用类型	请习	找数 / 请求比例❷	异常	3数 / 异常比例 🝞	平均自身耗时 / 比例 2	平均耗时	操作
▼ /cart	frontend	ΗΤΤΡλΠ	22	/ 100.00%	0	/ 0.00%	16.2ms	18.2ms	统计分析
com.alibabacloud.hipstershop.cartserv iceapi.service.CartService@viewCart	cart	提供 DUBBO	18	/ 68.18%	0	/ 0.00%	1.7ms	1.7ms	统计分析
com.alibabacloud.hipstershop.cartserv iceapi.service.CartService@addItemTo Cart	cart	提供 DUBBO	5	/ 22.72%	0	/ 0.00%	3ms	3ms	统计分析
com.alibabacloud.hipstershop.cartservicea pi.service.CartService@viewCart	cart	提供 DUBBO	12	/ 100.00%	0	/ 0.00%	2.08ms	2.08ms	统计分析
▼ /	frontend	HTTP入口	8	/ 100.00%	1	/ 12.50%	270.6ms	281ms	统计分析

- 将鼠标放在Span名称上,可以查看包含该Span的TraceID。
- 单击应用名可以进入该应用的应用总览页面。
- 单击目标Span操作列的统计分析,可以查看该Span的详细信息,包括各接口的调用类型占比、消耗总时间、接口名称、请求次数、样本推荐和所有调用方法的名称、类型、调用次数、执行时间和时间百分比。



相关操作

在调用链路页面,单击指标监控列的折线图,可以查看不同时间段的请求数、响应时间和错误数。

在调用链路页面,单击方法栈列的 ④ 图标,进入方法栈对话框。

方法栈					\times
应用名称:	服务名称	a Carlos de		IP地址:	
Traceld:	产生时间	2021-04-01 19:06:3.048		耗时: 119ms	
方法栈					
调用方法	行号	扩展信息	时间轴(ms)		
▼ Tomcat Servlet Process			119ms		
org.apache.catalina.core.StandardHostValve.invoke(org.ap ▼ che.catalina.connector.Request request, org.apache.catalin connector.Response response)	a a. 134	参数: action=TraceAction&ever	118ms		
com.alibaba.arms.console.valve.account.impl.NormalA SAccountService.getLoginUser(javax.servlet.http.HttpS vletRequest request, javax.servlet.http.HttpServletResp se response)	er 27		11ms		
com.alibaba.arms.console.valve.client.ArmsAasClie t.getLoginUser(javax.servlet.http.HttpServletReques equest, javax.servlet.http.HttpServletResponse resp se)	r 51		11ms		

方法栈对话框字段说明如下:

- 调用方法:本地方法栈调用方法,展开后显示的是该方法的下一层调用。
- 行号:本地方法的代码所在行数。
- 扩展信息:
 - 参数:调用的输入参数等
 - SOL: 数据库调用的SOL语句等
 - 异常:抛错的信息等
- 时间轴:本地调用链路每次方法调用的时间分布。

3.3. Trace Explorer

ARMS应用监控的链路分析Trace Explorer功能是基于已存储的全量链路明细数据,自由组合筛选条件与聚合 维度进行实时分析,可以满足不同场景的自定义诊断需求。

功能入口

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏选择应用监控 > Trace Explorer, 然后在顶部菜单栏选择目标地域。

首次进入Trace Explorer页面需要授权ARMS操作SLS权限,以便数据投递到您名下的SLS进行二次开 发,但不会产生额外费用。

授权完成后会自动进行资源准备,预计等待2分钟即可完成。

?	说	明			
	0	开通Trace Explorer功能不能	会产生额外费用].	
	0	如果开通失败,请单击重试	或联系 <mark>技术支</mark>	持。	
Trac	e Ex	plorer			
\bigcirc	授权 授予 / SLS 进	2 RMS 操作 SLS 权限,以便数据投递到您名下的 行二次开发,且不会产生额外费用。	资源准备 〕 正在初始化当前区域的后端资源(不同区域需要单独初始	3 完成

- 3. 在Trace Explorer页面右上角的时间选择框设置需要查询的时间段。
- 4. 筛选链路。
 - 在左侧快捷筛选区域,通过状态、耗时、应用名称、接口名称和主机地址维度快速筛选链路。

正在初始化当前区域的后端资源(不同区域需要单独初始化),耗时约2分钟,请耐心等待。

筛选条件将会显示在页面右侧顶部文本框内。

• 单击右侧顶部文本框,在下拉弹窗中修改筛选条件或设置其他维度的筛选条件。

Trace Ex	plorer o			8	近30分钟			
1110 2965	三時間外間第四	statusCode=2	atusCode=2 O Q					
快播输选		常用査询 ~		2 墨莎李说说杨				
状態~	<u>沈政</u> / #6时	serviceName 应用名和	-					
□ 1.5%	зk	spanName BEDSR	r +	请否将以输入内容	~~~~			
		ip ±RUBL	-	请改组23款入小的				
耗时 ns ∨	8	duration 期时	BETWEEN	ns - ns	14:43:00 14	48.00 14:53:00		
0		海豹道(8) >						
uns	- 2m9s	+ 2109384		へ ##		٥		
应用名称 ~	次数 / 時时	1 Standardson T		**************************************	1	操作		

? 说明

- 单击文本框右侧的 。 图标可以保存当前筛选条件。
- 单击文本框右侧已保存视图可以查看已保存的筛选条件,单击目标的筛选条件可以快速 查看到对应筛选条件下的链路信息。
- 在右侧顶部文本框直接输入查询条件。查询语法说明,请参见Trace Explorer查询用法说明。

查询链路

筛选设置完成后, **Trace Explorer**页面将会显示筛选过滤后的链路查询信息,包括Span数和HTTP错误数的 柱状图,耗时百分位的时序曲线,以及Span列表。

Trace Explorer						最近30分钟	
查询 分析 三陸撤快速落选	statusCode=2 AND serviceName="apmZxTest"					8 Q C B	已保存视图 ~
快速筛选 状态 × <u>次数</u> / 形时 ビ 昇希 3k 本	spanik 120 16.35.00 90 e.count. 12	HTTP指误数 3 2			時时百分位 40s 30s		Î
₩ ₩bins> %	60 30 16.31.00 16.36.00 16.41.00 16.46.00 16.51.00 16.56.00 II count	1 17:01:00 17:05:00 16:32:00 16:3 500	37:00 16:42:00 16:47:00 16	.52:00 16:57:00 17:02:00	20s 10s 0ms 16:31:00 16:36:00 16:41:00 p99 p90 p90 p75 p50 avg_du	16:46:00 16:51:00 16:56:00 ration	17:01:00 17:05:00
0ns – 1m46s	搜索到3k坐span						۲
应用名称 ~ 次数 / 耗时	traceId	startTime 📲	serviceName	spanName	duration 4	ip	操作
Q 搜索serviceName字段的 🔽	Ob	10/26 17:03:55	apmZxTest	comaliy	Oms	11.165.71.108	童香
🖬 apmZxTest 3 🔺	Ob	10/26 17:03:55	apmZxTest	Recv To	0ms	11.165.71.108	25
	Obstantis to the termination of the second state	10/26 17:03:54	apmZxTest	comaliy	Oms	11.165.71.108	主委
	Observation of the second s	10/26 17:03:54	apmZxTest	Recv To	Oms	11.165.71.108	25
接口名称 × 次数 / 耗时	Observation	10/26 17:03:54	apmZxTest	comaliy	1ms	11.165.71.108	童香
Q 搜索spanName字段的修 🔽	Ob	10/26 17:03:54	apmZxTest	Recv To	1ms	11.165.71.108	25
com.aliyun.openservice1	Observed and a second se	10/26 17:03:44	apmZxTest	/demo/	2.3s	11.165.71.108	童老
Denu Tanic@tvMaTun 8	Oberenterenterenterenterenterenterenteren	10/26 17:03:35	apmZxTest	MySQL	29ms	11.165.71.108	25
主机地址 ~ 次数 / 耗时	Ob	10/26 17:03:34	apmZxTest	MySQL	36ms	11.165.71.108	童老
Q. 搜索p字段的值 %	Observation	10/26 17:03:34	apmZxTest	MySQL	27ms	11.165.71.108	±4
11.105.71.108 3 .	Ob	10/26 17:03:34	apmZxTest	MySQL	27ms	11.165.71.108	童老
-	Ob	10/26 17:03:33	apmZxTest	MySQL	34ms	11.165.71.108	24
٠	ob	10/26 17:03:33	apmZxTest	MySQL	37ms	11.100.71.100	童若

在Span列表区域,您可以执行以下操作:

- 单击目标Span右侧操作列的查看,可以查看完整的链路信息。更多信息,请参见链路详情。
- 单击右上角的 👩 图标可以增加或隐藏列表显示的参数。
- 将鼠标悬浮于Span参数上,单击右侧的了图标,可以将当前参数值添加为筛选条件。

擅亲到3k条span						۲
traceld	startTime 4	serviceName	spanName	duration 4	ip	操作
Ob	10/26 17:03:55	apmZxTest	com.aliyun.openservices.ons.a	Oms	11.165.71.108	查看
Ob	10/26 17:03:55	apmZxTest	Recv Topic@zxMqOne	0ms	11.165.71.108	宣告
Ob	10/26 17:03:54	apmZxTest	com.aliyun.openservices.ons.a	0ms	11.165.71.108	宣委
0b.	10/26 17:03:54	apmZxTest	Recv Topic@zxMqOne	0ms	11.165.71.108	童香
Ob	10/26 17:03:54	apmZxTest	com.aliyun.openservices.ons.a	1ms	11.165.71.108	宣香
Ob	10/26 17:03:54	apmZxTest	Recv Topic@zxMqOne	1ms	11.165.71.108	童香
Ob	10/26 17:03:44	apmZxTest	/demo/service	2.35	11.165.71.108	宣誓
Ob	10/26 17:03:35	apmZxTest	MySQL	29ms	11.165.71.108	2 6

分析链路

筛选设置完成后,在**Trace Explorer**页面左上角单击**分析**,在右侧数据展示区域右上角,设置对Span的分 析条件。



- count:显示Span被调用次数。
- group by: 按指定分组显示Span。
- limit to: 设置显示分析数据的Top 10、Top 5、Bottom 10或Bottom 5。

⑦ 说明 group by设置为no group时不支持设置limit to。

• 间隔: 按指定时间间隔分析Span。

设置完成后,可以查看筛选过滤后的Span被调用次数的时序图、具体统计数据、全链路聚合和全链路拓扑。

时序图



- 将鼠标悬浮于时序图上,可以查看对应分组下Span被调用的次数。
- 单击时序图底部的分组指标,可以隐藏或显示对应指标数据。

统计数据

> 文档版本: 20220713

搜索到34个span	
Name	count 🗄
Test	3328
consumer	72
KSender	40
producer	36

全链路聚合

Trace Explorer对查询到的Span可以按照各个维度进行分析,但这些分析是针对单个Span,并未在链路级别 深度分析。而全链路聚合功能支持通过指定条件查询分布式调用链路的Traceld(最多5000个),然后基于 这些Traceld查询对应的Span,并聚合这些Span得出最终结果,整个过程保证聚合的链路完整性。

⑦ 说明 由于全链路聚合是按照查询条件后聚合计算相应数据的,当您选择的条件比较多时,查询计算存在一定延迟,请耐心等待。

Trace Ex	plorer _③							最近30分钟
童词 分析	王章章快速等选	应用监控					×	Q Q 四 已保存視問 ~
快捷掷选		· 回 时序 · 翻 统计 · 翻 全域路限合 全域路拓扑						
状态 ~	次数 / 民时							
正常	65k	spanName	serviceName	③请求数 / 请求比例	③span数 / 请求倍数	③平均自身耗时 / 比例	平均耗时	②异常数 / 异常比例
日本	1k		arms	5 / 100.00%	5 / 1.00	0.00ms / 0.00%	2us	0 /0.00%
		No.	arms	1 / 20.00%	2 / 2.00	0.00ms / 0.00%	2us	0 /0.00%
利用~	8		arms	1 / 20.00%	1 / 1.00	0.00ms / 0.00%	4us	0 /0.00%
0	48.24	87,	arms	2 / 100.00%	2 / 1.00	0.00ms / 0.00%	23us	0 /0.00%
-	40.23	an and the second second second second second	arms	2 / 100.00%	4 / 2.00	0.00ms / 0.00%	9us	0 /0.00%
应用名称 ~	次数 / 托时	N	ams	2 / 100.00%	4 / 2.00	0.00ms / 0.00%	2us	0 /0.00%
Q. 搜索service?	Name字段 📡	M	arms	2 / 100.00%	2 / 1.00	0.00ms / 0.00%	4u5	0 /0.00%
~	66k	/	arms	2 / 100.00%	2 / 1.00	0.00ms / 0.00%	tus	0 /0.00%
		have a	arms	1 / 100.00%	1 / 1.00	0.00ms / 0.00%	fus	0 /0.00%
接口名称 ~	<u>次数</u> / 刊时							
Q. 提索spanNa	me学段的 📡							
Ciercia.	21k							
	13k							
	10k							
	- 8k - 4k							

参数	说明					
spanName	Span名称。					
serviceName	Span对应的应用名。					
请求数/请求比例	请求比例表示调用当前Span节点的请求比例数。 例如10%表示10%的请求会调用当前Span。 计算公式:请求比例=当前Span的请求数/总请求数*100%					
span数/请求倍数	请求倍数表示平均每个请求调用当前Span的次数。 例如1.5表示每个请求会调用当前Span 1.5次 。 计算公式:请求倍数=Span数/Span的请求数					
亚均白良耗时/比例	平均自身耗时表示不包括子Span的耗时。 例如,对于Span A和其子Span B, 其中A耗时为10 ms, B耗时为8 ms,那A 的自身耗时为2 ms。 计算公式:自身耗时=Span耗时-所有子Span耗时总和					
	↓ 注意 如果是异步调用,自身耗时即Span耗时,无需减去子Span 耗时。					
平均耗时	该Span的平均耗时。					

参数	说明
	异常比例表示出现异常的请求比例。 例如3%表示有3%的请求出现异常。 计算公式:异常比例=异常请求数/总请求数
异常数/异常比例	注意 异常请求数不等于异常数(Span调用异常的次数),当请求 倍数大于1时,一个异常请求可能对应多个异常数。

示例:	如下表所示,	Span A调用Span B和Span C,	各参数含义如下。
	20H 1 (24771)3 ()		

spanName		serviceNa me	请求数/请 求比例	span数/请 求倍数	平均自身 耗时/比例	平均耗时	异常数/异 常比例
A	-	demo	10/100.00 %	10/1.00	5.00ms/2 5.00%	20ms	2/20.00%
-	В	demo	4/40.00%	8/2.00	16.00ms/ 100.00%	16ms	2/50.00%
-	С	demo	1/10.00%	1/1.00	4.00ms/1 00.00%	4ms	1/100.00 %

对于入口Span, A的请求数/请求比例表示A的请求总数为10次,比例为100%。B的请求数/请求比例为 4/40.00%,表示只有4次请求调用了B,同理只有1次请求调用了C,对应的请求比例分别为40%和10%。其 余的请求可能因逻辑判断或者异常而未调用B和C。这里反映了请求的分布比例。

A的span数/请求倍数为10/1.00,表示每次请求只调用了一次A,但是对于B而言,4次请求有8个Span,每次请求调用了2次B。这里反应了一次请求中Span的分布比例。

A的平均自身耗时/比例为5.00ms/25.00%,表示A除了B和C之外的平均耗时为5ms,只占整体平均耗时的 25%。而子Span B和C因为没有子Span,所以自身耗时即整体耗时。这里反应了耗时的分布比例。

A的异常数/异常比例为2/20.00%,表示A发生了2次异常,占整体请求的20%。B的异常数/异常比例为 2/50.00%,因为每次请求调用了2次B,总的请求数是4,异常比例是50%,那么2次请求发生了异常。所以B 的分布可能是:一共有4次请求,其中有2次请求调用的4个Span B都是正常的,剩下2次请求中,首次Span B的调用都发生异常,然后重新调用成功。

⑦ 说明 如果需要查看具体的调用链详情,可以将鼠标悬浮于蓝色的Span名称上,在悬浮框中可以看 到推荐的调用链ID,单击Traceld即可查看。

全链路拓扑

全链路拓扑页签显示调用链聚合后的应用间拓扑。如下图所示,表示两个应用间存在调用关系,同时展示相应的请求数、错误数、响应时间等数据。

									最近30分钟		
riceName	-							×	۹ د	6	已保存視图
]时序	田 统计	目 全链路聚合	全链路拓扑								
						_ =	2日教 - 0				
					(二 道	读数:0 求数:0				
						user 响	应时间: Oms				
						1					
						1					
						1					
						i.					
					(误数: 0				
						· · · · · · · · · · · · · · · · · · ·					
					-						

链路详情

在链路详情面板可以查看链路下的所有接口、链路开始时间、总耗时、接口是否异常、以及每个接口的调用 耗时。

TraceID 0b 开始时间 2021-10-26 17:02:0	00 总耗时 1m46s		· 2
	Oms		1m46s
 ▼ 170 /http-call/service @ apmTestSender 	1m46s		
org.apache.http.protocol.HttpRequestExecutor.execut apmTestSender	36.2s		
→ 62 /demo/mysqlOne @ apmZxTest error	36.25		
 ◆ 62 /demo/mysqlTwo @ apmZxTest error 		35.45	
MySQL apmZxTest error		34ms	
MySQL apmZxTest error		36ms	
MySQL apmZxTest error		27ms	
MySQL apmZxTest error		29ms	
MySQL apmZxTest error		33ms	
MySQL apmZxTest error		30ms	

在链路详情面板,您可以执行以下操作:

- 将鼠标悬浮于接口名称上,可以查看接口对应的应用名、接口名、IP和开始时间。
- 单击目标接口右侧的 @ 图标, 可以查看目标接口下的方法栈信息。

	35.46	
方法栈		*
	0ms 22	1.8s
com.mysqljdbc.Statement.executeQuery(java.lang.Str	21.8s	
 com.mysql.jdbc.NonRegisteringDriver.connect(java.la 	21.75	
com.mysql.jdbc.Statement.executeQuery(java.lang.Str	21.5s	-
com.mysql.jdbc.Statement.executeQuery(java.lang.Str	l 34ms	-
com mysal idhc NonRegisteringDriverconnect/iava la		

 单击接口名称,可以在面板右侧查看对应接口的附加信息和指标详情。附加信息常见参数请参见Trace Explorer参数说明。JVM和Host详情请参见JVM监控和主机监控。

拉口叫加卢白

按口附加宿尽

应用名	apmZxTest	×
接口名	/demo/mysqlTwo	
IP	11.565.75.508	
开始时间	2021-10-26 17:02:36	
标签	error	
附加信息 指标		-
Attributes Resource	es Events	
callstack	1	
excep.ids	bination	
http.status_code	200	
net.peer.ip	11.165.71.108	
out.ids	arms	
ppid	b59	
source.ip	11.165.71.108	
tprof	1	
4		
安口指标详	情	
应用名	apmZxTest	>
接口名	/demo/mysqlTwo	
IP	11.165.75.108	
开始时间	2021-10-26 17:02:36	
标签	error	
附加信息 指标		4
JVM Host		

GC瞬时次数 32 24 16 16:46:00 16:51:00 16:56:00 17:01:00 17:06:00 17:11:00 17:16:00 - FullGC 次数 - YoungGC 次数 GC瞬时耗时 Ims Ims 16:46:00 16:51:00 16:56:00 17:01:00 - FullGC 耗时 - YoungGC 耗时 堆栈内存详情 10m '5m i0m :5m 0 16:56:00 16:51:00 16:56:00 17:01:00 17:06:00 17:11:00 17:16:00 - 年轻代Eden区 - 年轻代Survivor区 - 老年代 - 使用总和 JVM线程数 Trace 600

二次开发

链路数据保存在您的SLS中, Project名称为proj-xtrace-<encode>-<region-id>, logstore名称为logstoretracing。其中<region-id>是Trace Explorer对应的地域,例如cn-hangzhou。数据格式的含义,请参 见Trace Explorer参数说明。您可以基于已存储的全量链路明细数据进行二次开发,自由组合筛选条件与聚合 维度进行实时分析,可以满足不同场景的自定义诊断需求。更多使用方法,请参见通过TraceExplorer实时分析 链路数据。

3.4. ARMS OpenTelemetry Collector

本文介绍如何将应用数据接入ARMS OpenTelemetry Collector,并在ARMS控制台查看监控数据的上报流量、相关Trace和指标大盘。

前提条件

- 已开通ARMS,具体操作,请参见<mark>开通ARMS</mark>。
- 创建Kubernetes集群。各集群创建方法如下:
 - 。 创建Kubernetes专有版集群
 - 。 创建Kubernetes托管版集群
 - 创建注册集群并接入本地数据中心集群

背景信息

ARMS OpenTelemetry Collector是一套基于开源OpenTelemetry Collector开发的可观测数据(Traces、 Metrics、Logs)边缘侧统一采集与处理平台,具备安全、可靠、易用等特性,适合生产环境接入。

ARMS OpenTelemetry Collector服务端全托管(自动开通),客户端部署在用户集群内,支持边缘侧数据采 集、处理、缓存、查询与路由,可以大幅降低公网传输成本,统一多源异构数据格式,轻松实现多云或混合 云网络的数据融合与展示。

- 开箱即用:一键接入ARMS OpenTelemetry Collector后,即可自动获取Kubernetes容器资源层和网络性能数据,无需任何代码改造。
- 拥抱开源:兼容开源OpenTelemetry Collector。支持原生的OTLP、Jaeger、Zipkin等Receivers端口;可以自定义Pipeline,实现多重数据路由;支持开源Grafana大盘展示与探索。
- 异构端到端关联:无论是多云或混合云网络,还是不同的OpenTelemetry Agent/SDK、Jaeger、Zipkin等 异构链路框架,都可以通过ARMS OpenTelemetry Collector完成数据格式标准化,并自动上报至全托管的 服务端,实现异构数据的关联查询与统一展示。
- 无损统计:将Trace数据全量上报至ARMS OpenTelemetry Collector后,通过边缘计算自动生成Trace预 聚合指标,再对Trace进行采样上报,在降低数据传输和持久化存储成本的同时,还可以保证指标数据精 准统计,监控和告警效果不受采样率影响。
- 增强分析:通过ARMS OpenTelemetry Collector上报的Trace数据,支持Trace Explorer后聚合增强分析,实时定位流量不均、单机故障、慢SQL分布、灰度发布异常等经典线上故障,帮您开启链路追踪的"第三种玩法"。
- 高性能:通过时间窗口轮转的内存统计,数据压缩,连接优化等技术大幅降低了海量数据场景下的性能开销。
- 高可靠: 支持资源限额与自动限流保护, 保证高压场景下的集群稳定性。
- 灵活性: 支持动态配置下推, 可以实时调整流量开关, 调用链采样率, 接口过滤与收敛规则等参数。

ARMS OpenTelemetry Collector通过边缘侧的数据采集、处理、缓存与路由,可以实现多源异构数据的标准化;增强Traces、Metrics、Logs可观测数据间的关联性;支持无损统计,降低数据上报和持久化存储成本等。

场景一:全景监控数据一键采集与分析(容器环境)

ARMS OpenTelemetry Collector在容器服务ACK环境下,深度集成了阿里云Kubernetes监控组件与 Prometheus监控组件,一键安装完成后,会自动采集Kubernetes容器资源层和网络性能数据。结合用户上 报的应用层数据(只需修改Endpoint,无代码改造)与自动预聚合指标,全部上报至全托管的服务端数据中 心,再通过Grafana进行统一展示。最终实现覆盖应用、容器、网络、云组件的全景监控数据采集与分析。



场景二:多云/混合云网络,异构Tracing框架数据关联与统一展示

在多云/混合云架构下,不同集群或应用间的链路追踪技术选型可能存在差异,例如A采用了Jaeger, B采用 了Zipkin,C采用了Skywalking,不同链路追踪协议上报的数据格式互不兼容,无法串联,大幅降低了全链路 诊断效率。

通过ARMS OpenTelemetry Collector可以将不同来源的链路统一转化为OpenTelemetry Trace格式,并上报 至统一的服务端进行处理和存储,用户可以轻松实现跨网络或异构链路框架的联合数据查询与分析。



场景三:链路采样+无损统计,低成本实现应用监控告警精准统计

生产系统的日均调用量可以达到亿级别,调用链全量上报和存储的成本是个不小的开销,而且除了少数错慢 链路外,大部分调用链的查询价值并不高,对调用链执行采样存储是个不错的选择。

然而, 传统的链路采样会导致链路统计指标的准确性大幅下降, 例如一百万次真实调用经过10%采样后保留的十万次调用, 对其统计得到的结果会产生明显的"样本倾斜", 最终导致监控告警误报率过高, 基本处于不可用状态。

ARMS OpenTelemetry Collector支持Trace数据无损统计,对接收到的Trace数据自动进行本地预聚合,得 到精准的统计结果后再执行链路采样上报,这样在降低网络开销和持久化存储成本的同时,保证了应用监控 与告警指标的准确性。

此外, ARMS OpenTelemetry Collector还支持对Trace预聚合指标进行筛选、收敛与自定义打标,比如将同一类Trace调用收敛为一个接口,或者根据用户类型区分流量,如下图所示。

存現用報道面 OT Collectorを置 OT Collector集目455										
of Column RB: v d' SHER 922 (23) 0										
OT Collector 调用错设置	OT Colector 调明技设置									
是否采集调用链	最后车车间接									
调用链预聚合开关		说明:英讯此配置,OT Collector 将不再执行 Trace 转 Metrics 预聚合处理,指标数据可能不可见。								
调用延采样率	100 %	说明:调明疑上规采林事、取值观围 0.01% - 100%,符合深样率的磋器才会上接并存储。								
调用链流量用题	10000 奈 Trace / (時个 OT Collector 報時)	説用: OT Collector 諸等的处理的最大能求致,既以10000条r砂,大于该有值的同即指将不会采集。								
无效接口过滤 深州天	X推口运动									
匹配方式	匹配內容 ③	应用范围(应用名称时限)②	是否启用	80						
后级匹配	(pg.prg.js,jpeg	\$all_app		1658 BID						
发散接口收敛 2888	8月1日 秋秋									
电放正则 ③		应用添加(但用名称列集) ⑦	是否启用	80						
		没有除潮								
资源标签 ①										
标签名称	杨笙kary	标签 value (示例)	是否启用							
2.18	regionid	機關 OT Collector 所在区域自动积距,例如 cn-hangshou								
KaS集群ID	clusterid	根据 GT Collector 所在無其自动視聴其 K85 集群 ID								
环境	armsEnv	建取得印度 Attributes 中的 armsEnv 学校信, 例如 daily								
版本	armsWersion	提取调用结 Attributes 中的 armsWarsion 宇宙機、例約 v10.0								

ARMS Collector优势

适用场景

接入说明

ARMS OpenTelemetry Collector组件安装前,需先检查目标集群是否存在ARMS Addon Token,否则会出现 权限不足的报错。若存在ARMS Addon Token,ARMS会进行免密授权,组件安装完成后,即可使用ARMS OpenTelemetry Collector;若不存在ARMS Addon Token,则需要手动添加ARMS和链路追踪权限策略。

查看ARMS Addon Token和手动添加权限策略方法如下:

查看ARMS Addon Token > 手动添加ARMS和链路追踪权限策略 >

? 说明

- Kubernetes托管版集群:对于部分Kubernetes托管版集群,可能存在没有ARMS Addon Token 的情况,因此,对于Kubernetes托管版集群,建议首先检查ARMS Addon Token是否存在。若不存在,仍需进行手动授权。
- Kubernet es专有版集群:默认不支持ARMS Addon Token, 需进行手动授权。
- 注册集群:默认不支持ARMS Addon Token,需进行手动授权。由于注册集群没有Worker RAM 角色,无法通过Worker RAM角色手动添加ARMS和链接追踪的权限策略,因此,注册集群安 装Kubernet es监控组件ack-arms-cmonit or的操作,请参见注册集群安装组件。

安装方式

Kubernetes监控组件ack-arms-cmonitor中包含ARMS OpenTelemetry Collector容器,因此安装ack-arms-cmonitor即可接入ARMS OpenTelemetry Collector。

- 1. 安装Kubernetes监控组件ack-arms-cmonitor。

- ii. 在左侧导航栏中, 单击集群。
- iii. 在集群列表页面,单击目标集群名称或者目标集群右侧操作列下的详情。
- iv. 在左侧导航栏, 选择运维管理 > 组件管理。
- v. 在组件管理页面单击ack-arms-cmonitor区域的安装。
- vi. 在弹出的对话框中单击确定。

⑦ 说明 ack-arms-cmonitor默认安装在arms-prom命名空间下。

- 2. 检查ack-arms-cmonitor安装状态。
 - i. 在左侧导航栏选择工作负载 > 守护进程集。
 - ii. 在守护进程集页面切换命名空间为arms-prom。

如果cmonitor-agent(daemonset)中所有Pod都处于Running状态,则代表安装成功。

守护进程集 DaemonSet			使用镜像创建	使用YAML创建资源
请输入搜索内容 Q				刷新
□ 名称 标签 〒	容器 组数 量	镜像	创建时间	操作
	0/0	NATIONAL CONTRACTOR AND A CONTRACTOR AND	2022-02 17:21:46	-22 详情 编辑 监控 更多▼
cmonitor-agent otel-collecto app:cmonitor app.kubernet	r-app:otel-collector r-agent 1/1 tes.io/managed-by:Helm	regento de la formação desensionem las especimientes presentarios en especimientes presentarios en especimientes colonies en las especimientes colonies en las especimientes colonies en las especimientes de las entreses en especimientes de las especimientes en especimientes de las especimientes especimientes de las especimientes especimientes de las especimientes especimientes de las espectimientes especimientes de las espectimientes especimientes espectimientes de las espectimientes espectimientes de las espectimientes espectimien	2022-03 21:45:30	-08 详情 编辑 9 监控 更多▼
	1/1	Nageria da la Angelia d'Anna anteni Mangelia de Angelia d'Anteni Mangelia de la Angelia d'Anteni Mangelia d'Anteni	2022-02 17:21:46	-22 详情 编辑 5 监控 更多▼

上报应用数据

组件安装后默认配置的Receivers包括OTLP、Jaeger和Zipkin,端口与开源保持一致。

Receivers	protocols	port
	gRPC	4317
UT LP	http	4318
lagger	gRPC	14250
Jaeger	thrift_http	14268
Zipkin	无	9411

组件安装后会在arms-prom命名空间下默认创建名称为otel-collector-service的Service资源,用于将 Receivers中不同组件的相应端口进行暴露。Trace上报的Endpoint格式为 otel-collector-service.armsprom:<port> ,其中 otel-collector-service 是ARMS OpenTelemetry Collector的Service名 称, arms-prom 是其所在的命名空间。

示例:将生成的Traces通过OTLP的gRPC协议上报到ARMS OpenTelemetry Collector中。

```
func initProvider() func() {
    ctx := context.Background()
```

```
endpoint := "otel-collector-service.arms-prom:4317"
if !ok {
   log.Fatalf("Cannot init OpenTelemetry, exit")
   os.Exit(-1)
}
metricClient := otlpmetricgrpc.NewClient(
   otlpmetricgrpc.WithInsecure(),
   otlpmetricgrpc.WithEndpoint(endpoint))
metricExp, err := otlpmetric.New(ctx, metricClient)
handleErr(err, "Failed to create the collector metric exporter")
pusher := controller.New(
   processor.NewFactory(
        simple.NewWithExactDistribution(),
       metricExp,
   ),
    controller.WithExporter(metricExp),
    controller.WithCollectPeriod(2*time.Second),
)
global.SetMeterProvider(pusher)
err = pusher.Start(ctx)
handleErr(err, "Failed to start metric pusher")
serviceName, ok := common.GetClientServiceName()
if !ok {
    serviceName = common.ClientServiceName
}
headers := map[string]string{"Authentication": ""}
traceClient := otlptracegrpc.NewClient(
   otlptracegrpc.WithInsecure(),
   otlptracegrpc.WithEndpoint(otelAgentAddr),
    otlptracegrpc.WithHeaders(headers), // 鉴权信息
   otlptracegrpc.WithDialOption(grpc.WithBlock()))
log.Println("start to connect to server")
traceExp, err := otlptrace.New(ctx, traceClient)
handleErr(err, "Failed to create the collector trace exporter")
res, err := resource.New(ctx,
   resource.WithFromEnv(),
   resource.WithProcess(),
    resource.WithTelemetrySDK(),
   resource.WithHost(),
    resource.WithAttributes(
       // the service name used to display traces in backends
        semconv.ServiceNameKey.String(serviceName),
   ),
)
handleErr(err, "failed to create resource")
bsp := sdktrace.NewBatchSpanProcessor(traceExp)
tracerProvider := sdktrace.NewTracerProvider(
   sdktrace.WithSampler(sdktrace.AlwaysSample()),
   sdktrace.WithResource(res),
    sdktrace.WithSpanProcessor(bsp),
)
// set global propagator to tracecontext (the default is no-op).
otel.SetTextMapPropagator(propagation.TraceContext{})
otel.SetTracerProvider(tracerProvider)
```

```
log.Println("OTEL init success")
return func() {
    cxt, cancel := context.WithTimeout(ctx, time.Second)
    defer cancel()
    if err := traceExp.Shutdown(cxt); err != nil {
        otel.Handle(err)
    }
    // pushes any last exports to the receiver
    if err := pusher.Stop(cxt); err != nil {
        otel.Handle(err)
    }
}
```

默认Receivers配置说明

将应用数据上报到ARMS OpenTelemetry Collector

检查应用数据

查看ARMS OpenTelemetry Collector运行日志,确定应用数据已经上报到ARMS OpenTelemetry Collector。

- 1. 登录容器服务管理控制台。
- 2. 在左侧导航栏中,单击集群。
- 3. 在集群列表页面,单击目标集群名称或者目标集群右侧操作列下的详情。
- 4. 在左侧导航栏,选择工作负载 > 守护进程集。
- 5. 在守护进程集页面单击cmonitor-agent。
- 6. 在cmonitor-agent页面单击日志页签。
- 7. 选择容器为ot-collector。

查看ot-collector运行日志,如果发送的Metrics数量不为0,则代表已经有Trace成功上报,并自动转换为Metrics数据。



查看监控数据

1. 登录ARMS控制台。

- 2. 在左侧导航栏选择应用监控 > 全局配置。
- 3. 在**全局配置**页面单击OT Collector集群状态,然后在顶部菜单栏选择目标地域。 在OT Collector集群状态页签查看各集群的数据上报流量。

字储周期设置 OT Collector配置 OT Collector集群状态	
DT Collector 集群: 全部集群 〜 占	当前反域: 华北2(北京)
OT Collector集群名称:	
OT Collector健康状态: 4/4	
集群版本	
v1.0.0	4
数据上报 (最近5分钟均值)	
调用链Span数 59.75条/	9
週用链预聚合指标数 0.45条/	9
查看详情 ->	

1. 在ARMS控制台左侧导航栏选择**应用监控 > Trace Explorer**,然后在顶部菜单栏选择目标地域。 在**Trace Explorer**页面查看上报的Trace数据详情。更多信息,请参见Trace Explorer。

Trace E	xplorer ®						量近30分钟	
重调 分析	王勇嚴快獲得透	serviceName: "					× Q Q B	已保存视图 ~
快捷得选			1000000000			Sector 1		
状态 ~	次数 / 民时	spanax	HITE SERVER		,	45 877 14		
正常	587	32 74	16 750			10s	٨	
			500 250			58 2.59	$A \sim$	
飛时~	8	0 19:10:00 19:22:00 19:29:00 19:36:00 19:58:00	19:50:00 19:54:00 19:21:03		19:51:02	19:10:00 19:22:00 19:21	P.00 19:36:00 19:43:00	19:50:0019:54:00
0	0	Dount	count			p99 = p90 = p75 = p50 = avg	duration	
Ons	- 41.5s	機索到587条span						۲
应用名称 ~	次数 / 利利	traceld	startTime 1J	serviceName	spanName	duration 1	ip	操作
Q. 搜索servic	eName半段 😪	6	03/14 19:52:52	1000		39	192.168.2.23	22
_	587	c	03/14 19:52:39			38	192.168.2.23	童者
		0	03/14 19:52:14			2.9s	192.168.2.23	童者
		5	03/14 19:52:07			156ms	192.168.4.8	宝载
		£	03/14 19:52:07	1000		78ma	192.168.4.8	22
接口名称 ~	次数 / 長时	and and the second s	03/14 19:51:56	in the second se	and the design of	2.5s	192.168.2.23	童稚
Q. 搜索spanh	iame半段的 😪	3	03/14 19:51:35	1000		2.1s	192.168.2.23	宜看
	- 1	d'an anna an	03/14 19:51:16	inter a	and in contraction.	2.1s	192.168.2.23	童看
		(03/14 19:51:02	1000		2.5s	192.168.2.23	盘看
	<u> </u>	1	03/14 19:50:53	10.00	Aug. 10.00	29	192.168.0.100	28
		0	03/14 19:50:51	- Contract of the Contract of		2.7s	192.168.0.47	童者

- 1. 在ARMS控制台左侧导航栏选择Grafana服务 > 工作区管理。
- 2. 在工作区管理页面,单击Grafana服务共享版右侧的访问地址URL链接。
- 3. 在Grafana页面左侧导航栏选择
- 4. 在Manage页签通过关键字搜索名称为OpenTelemetry APM Dashboard的大盘,然后单击对应地域的大盘名称。

在OpenTelemetry APM Dashboard中查看监控指标。

<figure>RER I I I I I I I I I I I I I I I I I I</figure>	88 ot-i		/ OpenT	☆ ぷ		uhd+	월 🐵 🖵 🕐 Last 30	0 minutes ~ Q Q ~
	数据来源 All ~ 集群iD A	ll - 应用名 All - 主机	P All - 接口名称 All -					
CASURINA CASURINA <th< td=""><td>~应用总览 ◎ ◎</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></th<>	~应用总览 ◎ ◎							
	在线应用数 在线实	例数 在线接口数	全部	周用(当前)	错误调用(当前)		平均耗时(計	当前)
All Della (161) All Della (244)	11 *	1 4	60K cpm 40K cpm		50 cpm 40 cpm 30 cpm 20 cpm	m	2.0 s 1.5 s 1.0 s	
9.52 Mil 4.02 Mil 1171 s 172 s <th172 s<="" th=""> 172 s 172 s</th172>	调用总量(当前)	调用总量(-24H)	0.com		10 cpm		500 ms	
Try T	3.52 Mil	4.02 Mil	17:15 17:20 1	7:25 17:30 17:35 17:40	17:15 17:20 17:25 17:30	17:35 17:40	17:15 17:20 17:25	17:30 17:35 17:40
1.17 k 1.57 k 0.00 / 00	错误总量(当前)	错误总量 (-24H)	全部	期用 (-24H) ① timeshift -24	· 错误调用 (-24H)	🕑 timeshift -24h	平均耗时(-	24H) () timeshift -24h
Picker (138) Picker (241) Picker (241) <th< td=""><td>1.17 К</td><td>1.57 K</td><td>60K cpm</td><td></td><td>100 cpm 75 cpm</td><td></td><td>2.0 \$ 1.5 \$</td><td></td></th<>	1.17 К	1.57 K	60K cpm		100 cpm 75 cpm		2.0 \$ 1.5 \$	
627.1 2 ms 626.41 ms 0 mm 1/3 m 1/2	平均耗时(当前)	平均耗时 (-24H)	20K cpm		50 cpm		1.0 s 500 ms	
HTP B45425 HTP B4525	627.12 ms	626.41 ms	17:15 17:20 1	7:25 17:30 17:35 17:40	17:15 17:20 17:25 17:30	17:35 17:40	17:15 17:20 17:25	17:30 17:35 17:40
HTP Bdt A2 HTP Bdt A2 60/4 60/4 60/4 60/4 60/4 60/4 60/4 60/4 60/4 60/4 60/4 60/4 60/4 60/4 60/4 60/4 60/4 60/4 60/4 60/4 60/4 60/4 60/4 60/4 60/4 60/4 60/4 60/4 60/4 60/4	~HTTP 服务端总览							
60% 60% <td></td> <td>HTTP 組件类型</td> <td></td> <td>HTTP</td> <td>响应码</td> <td></td> <td>HTTP 请求类型</td> <td></td>		HTTP 組件类型		HTTP	响应码		HTTP 请求类型	
1715 1720 1725 1735 1736 1736 1735 1736 1736 1736 1736 1736 1736 1736 1736 1736 1736 1736 1736 1736 1736 1736 1736 1736 <	60 K 50 K 40 K 20 K 10 K		60 K 50 K 40 K 30 K 20 K 10 K			30 K 25 K 20 K 15 K 10 K 5 K		
(c)	17:15 17:20	17:25 17:30 17:35	17:40	17:15 17:20 17:25	17:30 17:35 17:40	17:15 17:20	17:25 17:30	17:35 17:40
Best Part (bit and bit	- (componentType='None')			(httpStatusCode="2xx")		- (httpMethod='GET')	 (httpMethod="POST") 	
<u>初発称 彼ら称 はた</u>				HTTP 服务端接口	状态(最近一分钟)			
<u>ත්තියෙකක</u> එක්කි විද්යා කර	应用名称	19	四名称				全部调用 ↓	平均耗时
otlo-server // hello 20.5K com 0.017 ms	otlp-server	n	rello				21.9K cpm	0.017 ms
	otlp-server	n	nello				20.5K cpm	0.017 ms

查看数据上报流量

查看上报的Trace数据

查看监控指标大盘 相关文档

- 开始监控Java应用
- •
- 通过OpenTelemetry Java SDK进行手工埋点
- OpenTelemetry接入说明
- ARMS OpenTelemetry Collector指标说明
- ARMS OpenTelemetry Collector配置说明

3.5. 应用总览

应用总览页面显示应用的关键指标、上下游依赖组件以及3D拓扑图,帮助您掌握应用的总体健康情况。

功能入口

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏中选择应用监控 > 应用列表,并在顶部菜单栏选择目标地域。
- 在应用列表中选择您想查看的应用,进入应用总览页面。
 您可以在应用总览页面顶部选择概览分析、拓扑图和3D拓扑页签查看相应信息。

概览分析

概览分析页签上展示以下关键指标:

- 选定时间内的总请求量、平均响应时间、错误数、实例数、Insights、Full GC次数、慢SQL次数、异常次数 和慢调用次数,以及这些指标和上周、上一天的同比升降幅度。
- 应用相关事件: 应用相关的事件, 比如0-1报警, 应用监控报警, K8s集群事件等。
- 应用提供服务: 应用提供服务的请求量和平均响应时间的时序曲线。

⑦ 说明 如需查看平均响应时间各分位数曲线,请先在自定义配置页面的高级设置区域开启分位 数统计开关。具体操作,请参见自定义配置。

- 应用依赖服务:应用依赖服务的请求量、平均响应和应用实例数的时序曲线,以及HTTP-状态码统计。
- 系统信息: CPU、MEM和负载的时序曲线。
- 慢调用: 慢调用的时序曲线和调用详情。
- 统计分析: 接口慢调用分析和异常类型分析。



应用拓扑

在**拓扑图**页签上,您可以通过拓扑图更加直观地看到应用的上下游组件以及与它们的调用关系,从而更快速 地找出应用的瓶颈。

I	✓ □ ⑦ 应用健康概定	概览分析	拓扑图	3D拓扑			最	近15分钟	Ē
				鐐	实例健康: 🕚				
						2 严重: 0			
					调用类型	调用次数	平均响应	如间/次	错误率
						32	0.6ms		1.0%
	19.73/00/27	# HITP			调用 MYSQL	134	1.0ms		0.1%
	+#19 1274.	.19ms			НТТР入□	296	1.3s		0.0%
	TOMOAT				调用 HTTP	268	789.8ms		0%
1					实例IP (共计: 2)	运行时	ĸ	进程号	JVM版本
	8.93次/分钟 MYSQL 1 ,	7.87次/分钟 UNKNOWN			172.20.0.88	31天34	▶时16分	1	1.8.0_191
	平均 0.96ms	平均 789.76ms			172.20.1.19	31天3/	10时16分	1	1.8.0_191
				RESET					
	请求数 / 每分钟 ● HTTP入□	响应时间 / 每分钟		• НТТРДД	错误率 / 每分钟	钟			• НТТРДП
	24 18 12 6 0 11-19 16:44 11-19 16:52 11-19 16:56	2.4s 1.8s 1.2s 600ms 0ms 11-19 16:44 11-19 16:48 1	1-19 16:52	11-19 16:56	2 1.5 1 0.5 0 11-19 16:44 1	1-19 16:48 1	1-19 16:52	11-19 16:	

3D拓扑

在3D拓扑页签上,立体地展示了应用、服务和主机的健康状况,以及应用的上下游依赖关系。借助3D拓扑 图,您可以快速定位诱发故障的服务、被故障影响的应用和关联的主机等,全方位地诊断故障根源,从而快 速排除故障。3D拓扑详细介绍,请参见应用监控3D拓扑图。



3.6. 应用详情

3.6.1. 概览

本文说明如何查看应用概览,从而了解应用拓扑、请求数、响应时间、错误数、HTTP状态码等信息。

前提条件

接入应用监控

功能入口

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏,选择应用监控 > 应用列表。
- 3. 在顶部菜单栏,选择地域。
- 4. 在应用列表页面,单击应用名称。
- 5. 在左侧导航栏,单击应用详情。
- 6. 在**应用详情**页面,选择应用实例,设置时间段,单击概览页签。

arms-kas-demo											4/1	
0	响后时间 / 请求数 / 错误数 / 异常数 📑	概章 JVM监控②	主机监控⑦ SQ	L调用分析 NoSQL调用分析	异常分析	错误分析 上	游应用 🚥	下游应用 🚥	接口快照			
应用分组 全部	✓ 請職入 Q											
arms-k8s-demo								USER				
• 172.2	20.10s / 61.9K / 51.5K / Ö							1.17205919 HTTP				
• 172.2								₩10 30134.28ms				
cn-hangzhou-f ecs	sn2ne.4xlarge							172.20				
												+ RESET
												•
		请求数 / 每天				Ŀ	∠⊡ ¥	响应时间 / 每天 л	由曲线突遣水深虚分析 (专家版)	7用)		∠ □ ★
		8000		Γ			1	30s				
		4000						22.58				
		2000		/				7.5s				
		0	01.07.00.00	01.13.0200	01 10 00.00		01.35.00.00	0ms	01.07.00.00	01.12.00.00	01 10 00-00	01.05.00.00
		01-0100.00	01-07-00.00	• HTTP入口	01-17-00.00		01-23 00.00	01-0100.00	01-07 00.00	 НТТРДД 	01-17 00.00	01-23 00.00
		错误数 / 每天 点击曲	1978-1972-055 (†281	(可用)		Ŀ	⊻ ⊡ ¥	HTTP-状态码统计				k o
		0003		Γ			1	6000				_
		4500						4500				
		1500						1500				
		0	\backslash	/			<u>``</u>		\checkmark			
		01-01 00:00	01-07 00:00	01-13 00:00 ● HTTP入口	01-19 00:00		01-25 00:00	01-02 00:00	01-08 00:00	01-14 00:00 • 2XX • 3XX • 4XX • 5XX	01-20.00:00	88
												-

应用拓扑

应用拓扑区域显示该应用在指定时间段的内部服务的调用关系拓扑图。

	200
USER	
2.96Queries/Minute HTTP	
On Average 24110.98ms	
ams-K8s-d	
	3
	SET
	3

- 1. (可选)在应用拓扑区域,您可以执行以下操作:
 - 单击途图标,设置应用拓扑图的显示设置。

⑦ 说明 设置会被浏览器存储,下次依然生效。

- 单击加号图标或向上滑动鼠标滚轮,放大应用拓扑图。
- 单击减号图标或向下滑动鼠标滚轮,缩小应用拓扑图。
- 单击RESET图标,将应用拓扑图恢复至默认大小。

请求数

请求数区域显示该应用在指定时间段的请求数时序曲线。



- 1. (可选)在请求数区域,您可以执行以下操作:
 - 将光标移到统计图上, 查看统计情况。
 - 使用光标选中一段时间,查看指定时间段的统计情况。
 - 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。
 - 单击 图标, 查看该指标的API详情。
 - 单击 資图标,为该指标创建报警。具体操作,请参见创建报警。

响应时间

响应时间区域显示该应用在指定时间段的响应时间时序曲线。



- 1. (可选) 在响应时间区域, 您可以执行以下操作:
 - 将光标移到统计图上,查看统计情况。
 - 使用光标选中一段时间,查看指定时间段的统计情况。
 - 单击曲线突增点,进行深度分析。

? 说明 仅专家版支持该功能。

- 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。
- 单击 国标,查看该指标的API详情。
- 单击 資图标,为该指标创建报警。具体操作,请参见创建报警。

错误数

错误数区域显示该应用在指定时间段的错误数时序曲线。



- 1. (可选)在错误数区域,您可以执行以下操作:
 - 将光标移到统计图上,查看统计情况。
 - 使用光标选中一段时间,查看指定时间段的统计情况。
 - 单击曲线突增点,进行深度分析。

⑦ 说明 仅专家版支持该功能。

- 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。
- 单击园图标,查看该指标的API详情。
- 单击 資图标,为该指标创建报警。具体操作,请参见创建报警。

HTTP状态码

HTTP状态码区域显示该应用在指定时间段的HTTP状态码时序曲线。



1. (可选)在HTTP-状态码统计区域,您可以执行以下操作:

- 将光标移到统计图上, 查看统计情况。
- 使用光标选中一段时间,查看指定时间段的统计情况。
- 单击图例,隐藏或显示数据。
- 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。
- 单击 図标, 查看该指标的API详情。

3.6.2. JVM监控

JVM监控功能用于监控重要的JVM指标,包括堆内存指标、非堆内存指标、直接缓冲区指标、内存映射缓冲区 指标、GC(Garbage Collection)累计详情和JVM线程数等。本文介绍JVM监控功能和查看JVM监控指标的操 作步骤。

功能入口

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏选择应用管理 > 应用列表,并在顶部菜单栏选择目标地域。
- 3. 在应用列表中单击您想查看的应用。
- 4. 在左侧导航栏单击应用详情。
- 5. 在应用详情页面选择您想查看的实例,并在页面右侧单击JVM监控页签。



查看JVM监控指标

JVM监控页签内展示了GC瞬时次数、GC瞬时耗时、堆内存详情、元空间详情、非堆内存、直接缓冲区和JVM 线程数的时序曲线。

- 单击GC瞬时次数/每分钟和GC瞬时耗时/每分钟面板右上角的瞬时值和累计值按钮,可以切换查看GC瞬时次数和GC瞬时耗时的时序曲线。
- 单击各监控面板上的指标名称(例如FullGC次数),可打开或关闭该指标在图表中的可见性。

⑦ 说明 每个图表必须至少有一个指标设为可见,这意味着当图表中只有一个指标时,您无法关闭 该指标的可见性。

单击堆内存详情/每分钟、元空间详情/每分钟、非堆内存/每分钟、直接缓冲区/每分钟和JVM线程数/每分钟的右上角的查看API按钮,可看该监控指标的API详情。

功能介绍

JVM监控功能可监控以下指标:

- GC(垃圾收集)瞬时和累计详情
 - FullGC次数
 - YoungGC次数
 - FullGC耗时

- YoungGC耗时
- 堆内存详情
 - 堆内存总和
 - 堆内存老年代字节数
 - 堆内存年轻代Survivor区字节数
 - 堆内存年轻代Eden区字节数
 - 已提交内存字节数
- 元空间
 - 元空间字节数
- 非堆内存
 - 非堆内存提交字节数
 - 非堆内存初始字节数
 - 非堆内存最大字节数
- 直接缓冲区
 - DirectBuffer总大小(字节)
 - DirectBuffer使用大小(字节)
- JVM线程数
 - 线程总数量
 - 死锁线程数量
 - 新建线程数量
 - o 阻塞线程数量
 - 可运行线程数量
 - 终结线程数量
 - 限时等待线程数量
 - 等待中线程数量

3.6.3. 池化监控

您可以通过池化监控功能监控具体应用所使用的线程池的各项指标,包括核心线程数量、当前线程数量、最 大线程数量、活动线程数量、提交任务数量和任务队列容量。

前提条件

接入应用监控

↓ 注意 仅专家版支持池化监控功能。

开启池化监控

- 1. 登录ARMS控制台, 在左侧导航栏选择应用监控 > 应用列表。
- 2. 在顶部菜单栏,选择地域,然后单击目标应用名称。
- 3. 在左侧导航栏单击应用设置。
- 4. 在自定义配置页签下的高级设置区域内, 打开线程池监控的开关。

○ 注意 池化监控支持Tomcat/Dubbo/HSF/Druid/HikariCp/SchedulerX等框架的线程池指标监控,您需要将Agent升级至最新版本。详细信息,请参见升级探针。应用重启后,池化监控功能生效。

查看池化监控指标

开启池化监控功能后,您可以在**应用详情 > 池化监控**下查看各项池化监控指标。**池化监控**页签包含以下指标的时序曲线:核心线程数量、当前线程数量、最大线程数量、活动线程数量、提交任务数量和任务队列容量。

\sim	(Tomcat)	
	●核心线程数量 ●当前线程数量 ●最大线程数量	● 活动线程数量
200		0.03
150		
100		
50		0.0075
0	04-18 00:00 04-19 00:00 04-20 00:00	0
	● 提交任务数量	● 任务队列容量
000000		0.0012
500000		0.0009
000000		
500000		0.0003
0	04-18 00:00 04-19 00:00 04-20 00:00	
监控指标	ī	说明
核心线程	数量	线程池内核心线程数量。
当前线程	数量	线程池内当前存在的线程数量。
最大线程	数量	线程池可容纳的最大线程数量。
活动线程	数量	当前正在处理任务的线程数量。
提交任务	数量	已提交到线程池内的任务数量。
任务队列	容量	线程池的阻塞队列大小。

⑦ 说明 当线程池内当前线程数少于核心线程数时,即使存在空闲线程,在提交新任务时也会创建新的线程执行该任务。当线程池内线程数超过核心线程数量,但未超过最大线程数量,且任务队列容量已满时,提交的新任务会通过创建新的线程来执行。线程池内线程数不会超过最大线程数量。

在**池化监控**页签下,您可以执行以下操作:

- 将光标移到统计图上,查看统计情况。
- 单击各监控面板上的指标名称(例如核心线程数量),打开或关闭该指标在图表中的可见性。

⑦ 说明 每个图表必须至少有一个指标设为可见,因此当图表中只有一个指标时,您无法关闭该指标的可见性。

3.6.4. 主机监控

主机监控功能用于监控CPU、内存、Disk(磁盘)、Load(负载)、网络流量和网络数据包的各项指标。本 文介绍主机监控功能和查看主机监控指标的操作步骤。

功能入口

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏选择应用监控 > 应用列表,并在顶部菜单栏选择目标地域。
- 3. 在应用列表页面选择您想查看的应用。
- 4. 在左侧导航栏单击应用详情。
- 5. 在**应用详情**页面选择您想查看的节点,并在页面右侧单击**主机监控**页签。

arms ~	④ JVM监控		最近15分钟 🗎
⑦ 响应时间 / 请求款	(/ 惜決数 / 异菜数 三)	電流 JVM监控① 主机监控① SQU範用分析 NoSQU範用分析 异常分析 描读分析 上際应用 💶 下卧应用 💶 接口快照	
应用分组 全都 ╰ 讷端入	Q	CPU 点击曲线完缩点深度分析(电影振可用)	∠ □ 常 堂
arms		 ・ 単和 ・ 単純CPU便用本 ・ 単純IO売成的CPU便用本 ・ ・ ・	
• (1)(0)(0)	1.4s / 150 / 8 / 27		
cn-hangzhou-f ecs.sn2ne4xlarge		2.6%	
•	1.1s / 152 / 8 / 27	126	
		0% 11-24 14:51 11-24 14:55 11-24 14:59 11-24 15:03	
		物理内存	ビロ業業
		● 邑和 ● 系统的空洞内存 ● 系统的已经使用的内存 ● 系统的PageCache型的内存数 ● 系统的BurierCache的内存数	
		74.50	
		37.36	
		18.46	
		11504 (NG)1 11504 (NG)2 11504 (NG)2 11504 (NG)2	
		Dick	医同常常
		 ・他和 ・融血団肉数 ・融血団肉数 ・融血団肉数 ・融血団肉数 ・ ・ ・	
		558.80	
		19.100	

查看主机监控指标

主机监控页签展示了CPU、内存、Disk(磁盘)、Load(负载)、网络流量和网络数据包的时序曲线。

• 单击各监控面板上的指标名称(例如系统CPU使用率),可打开或关闭该指标在图表中的可见性。

⑦ 说明 每个图表必须至少有一个指标设为可见,这意味着当图表中只有一个指标时,您无法关闭 该指标的可见性。

- 单击右上角的折线图按钮, 可选择区间查看和对比查看。
- 单击右上角的查看API按钮,可查看该监控指标的API详情。
- 单击监控面板右上角的一和一图标,可以查看已有报警的报警点和创建新的报警。创建报警的方法,请
 参见创建报警。

功能介绍

主机监控功能可监控以下指标:

- CPU
 - CPU使用率总和
 - 系统CPU使用率
 - 用户CPU使用率
 - 。 等待IO完成的CPU使用率
- 物理内存
 - 。 系统总内存
 - 。 系统空闲内存
 - 系统已使用内存
 - 系统PageCache中的内存
 - 系统BufferCache中的内存
- Disk (磁盘)
 - 系统磁盘总字节数
 - 系统磁盘空闲字节数
 - 系统磁盘使用字节数
- Load (负载)
 系统负载数
- 网络流量
 - 网络接收的字节数
 - 。 网络发送的字节数
- 网络数据包
 - 。 每分钟网络接收的报文数
 - 。 每分钟网络发送的报文数
 - 。 每分钟网络接收的错误数
 - 。 每分钟网络丢弃的报文数

3.6.5. Pod监控

本文说明如何查看Pod监控,从而了解应用的Pod情况,包括CPU、物理内存、网络流量、网络数据包等信息。

前提条件

接入应用监控

⑦ 说明 仅部署在Pod中的应用支持查看Pod监控。

功能入口

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏,选择应用监控 > 应用列表。
- 3. 在顶部菜单栏,选择地域。

应用实时监控服务 ARMS

- 4. 在应用列表页面,单击应用名称。
- 5. 在左侧导航栏,单击应用详情。
- 6. 在应用详情页面,选择Pod实例,设置时间段,单击Pod监控页签。

IDU-Consumer ^③ MM論控				¥ī⊒	1个月 自
⑦ 响应时间 / 请求数 / 错误数 / 异常数 書	↓ 概范 JVM监控⑦ 主机	监控 Pod监控 SQL调用分析 NoSQ	调用分析 异常分析 错误分析 上游应用 🏎	下游应用 1000 接口快照 日志	
请输入 Q	CPU		●累计CPU使用量 ● CPU配额		
idu-consumer- 11.6ms / 982 / 190 / cn-hangshourg ecc.c5.starge	0 0.8				
idu-consumer- " 0.7ms / 139 / 13 / cn-hangzhou-g ecs.c5.xlarge	0.3				
	0	01-14 00:00	01-16 00:00	01-18 00:00	
<	物理内存		 内存使用量 ○内存配額 		
	457.8M 343.3M				
	228.9M				
	0	01-14 00:00	01-16 00:00	01-18 00	:00
	网络流量(Byte)	。网络海你动空节数。网络岩洋动空节数	网络数据包(个)	洋的煤立数 。 网络银发手车的煤立数 。 网络将洋的港港	か (1):838(かか):第三章の
	2000		6 		
	1000		3		
	500		1.5		88

CPU

CPU区域显示该应用在指定时间段的Pod的CPU情况,包括以下指标:

- 累计CPU使用量
- CPU配额

CPU		累计CPU使用量 ● CPU配额	
1			
0.8			
0.5			
0.3			
0	01-14 00:00	01-16 00:00	01-18 00:00

1. (可选)在CPU区域,您可以执行以下操作:

- 将光标移到统计图上, 查看统计情况。
- 使用光标选中一段时间,查看指定时间段的统计情况。
- 单击图例,隐藏或显示数据。

物理内存

物理内存区域显示该应用在指定时间段的Pod的物理内存情况,包括以下指标:

- 内存使用量
- 内存配额

物理	內存	● 内存使用量 ● 内存配额	
457.8M			
343.3M			
228.9M			
114.4M			
0			
	01-14 00:00	01-16 00:00	01-18 00:00

- 1. (可选)在物理内存区域,您可以执行以下操作:
 - 将光标移到统计图上,查看统计情况。
 - 使用光标选中一段时间,查看指定时间段的统计情况。
 - 单击图例,隐藏或显示数据。

网络流量

网络流量区域显示该应用在指定时间段的Pod的网络流量情况,包括以下指标:

- 网络接收的字节数
- 网络发送的字节数

网络流量 (Byt	e) • 网维	接收的字节数 🏾 网络发送的	的字节数	
2000				
1500				
1000				
500				
0				
	01-14 00:00	01-16 00:00	01-18 00:00	

- 1. (可选)在网络流量区域,您可以执行以下操作:
 - 将光标移到统计图上, 查看统计情况。
 - 使用光标选中一段时间,查看指定时间段的统计情况。
 - 单击图例,隐藏或显示数据。

网络数据包

网络数据包区域显示该应用在指定时间段的Pod的网络数据包情况,包括以下指标:

- 网络发送丢弃的报文数
- 网络发送的报文数
- 网络接收丢弃的报文数
- 网络发送的错误数
- 网络接收的错误数

网络数据包(个) • 网络发送丢弃的报文数	• 网络发送的报文数	● 网络接收丢弃的报文数	● 网络发送的错误数	● 网络接收的错误数
6				
4.5				
3				
1.5				
0				
01	1-14 00:00	01-16 00:00	01-18 00:00	

- 1. (可选)在网络数据包区域,您可以执行以下操作:
 - 将光标移到统计图上,查看统计情况。

- 使用光标选中一段时间,查看指定时间段的统计情况。
- 单击图例,隐藏或显示数据。

3.6.6. SQL调用分析

本文说明如何查看SQL调用分析,从而了解应用的SQL调用情况。

前提条件

接入应用监控

功能入口

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏,选择应用监控 > 应用列表。
- 3. 在顶部菜单栏,选择地域。
- 4. 在应用列表页面,单击应用名称。
- 5. 在左侧导航栏,单击应用详情。
- 6. 在应用详情页面,选择应用实例,设置时间段,单击SQL调用分析页签。

arms-k8s-demo-subcompone Y Q ③ JVM监控					7	кл 🖿	
⑦ 响应时间 / 请求数 / 错误数 / 异常数 副	概览 JVM监控② 主机监控③ SQL调用分析 No	SQL调用分析 异常分析	错误分析 上游应用 💶 下游应用 💶 接口快照				
应用分组 全部 ∨ 请输入 Q	SQL调用统计 / 每天						
arms-k8s-demo-subcomponent 172.						800	
en-hangzhou-i ecs.g6.xlarge						400	
	01-01 00:00 01-14 00:00						
	所属应用	数据库类型	SQL语句	平均耗时非	调用次数小	操作	
<	arms-k8s-demo-subcomponent	MySQL	select * from 'user_notExist' where id = 0#	1.4ms	19.1K	调用统计 接口快照	

SQL调用统计

SQL调用统计区域显示该应用在指定时间段的SQL调用时序曲线。



- 1. (可选)在SQL调用统计页签下,您可以执行以下操作:
 - 将光标移到统计图上,查看统计情况。
 - 使用光标选中一段时间,查看指定时间段的统计情况。
 - 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。
 - 单击 图标, 查看该指标的API详情。

SQL语句列表

SQL语句列表显示该应用在指定时间段的所有SQL语句的列表。

所属应用	数据库类型	SQL语句	平均耗时 小	调用次数 小	操作
arms-k8s-demo-subcomponent	MySQL	select * from `user_notExist` where id = 0#	1.4ms	19.2K	调用统计 接口快照

- 1. (可选)在SQL语句列表,您可以执行以下操作:
 - 。在SQL语句的操作列,单击调用统计,查看该SQL语句的SQL调用时序曲线。
 - 在SQL语句的操作列,单击接口快照,查看该SQL语句调用的接口的快照。
 接口快照说明,请参见接口快照。

3.6.7. NoSQL调用分析

本文说明如何查看NoSQL调用分析,从而了解应用的NoSQL调用情况。

前提条件

接入应用监控

功能入口

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏,选择应用监控 > 应用列表。
- 3. 在顶部菜单栏,选择地域。
- 4. 在应用列表页面,单击应用名称。
- 5. 在左侧导航栏,单击应用详情。
- 6. 在**应用详情**页面,选择应用实例,设置时间段,单击NoSQL调用分析页签。

Spring-Redis >> 口③ JVM監控					i	近三个月 🛗
⑦ 响应时间 / 请求数 / 错误数 / 异常数 三	概范 JVM监控⑦ 主机监控⑦ SQL调用分析	NoSQL调用分析 异常分析 错误分析	上游应用 💶 下游应用 💶 接口快照			
应用分组 全部 ∨ 请输入 Q	调用统计 / 每天					
Spring-Redis						00
• 30.225 768.5ms / 42 / 0 / 0	0.6	1				60
• 30.225 0ms / 0 / 0 / 0	0.4					40
• 30.225 0ms / 0 / 0 / 0	0.2					20
• 30.225 Oms / 0 / 0 / 0	11-01 00:00	11-23 00:00	12-15 00:00 01-06 00:00			01-29 00:00
• 30.225 Oms / 0 / 0 / 0						
	所属应用	NoSQL类型	操作命令	平均耗时小	调用次数 1	操作
<	Spring-Redis	Redis	DEL	3.9ms	7	调用统计 接口快照
	Spring-Redis	Redis	RPUSH	0.7ms	21	调用统计 接口快服
	Spring-Redis	Redis	LLEN	0.3ms	7	调用统计 接口快服
	Spring-Redis	Redis	UNDEX	0.2ms	21	调用统计 接口快服
	Spring-Redis	Redis	EVAL	0.1ms	14	调用统计 接口快服

NoSQL调用统计

NoSQL调用统计区域显示该应用在指定时间段的NoSQL调用时序曲线。

调用统计 / 每天				
0.8				80
0.6				60
0.4				40
0.2				20
0	11 00 00 0	10 15 00-00	01.07.00.00	0
11-01 00:00	11-23 00:00	12-15 00:00	01-06 00:00	01-29 00:00

1. (可选)在NoSQL调用统计页签下,您可以执行以下操作:

- 将光标移到统计图上, 查看统计情况。
- 使用光标选中一段时间,查看指定时间段的统计情况。
- 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。
○ 单击 图标,查看该指标的API详情。

操作命令列表

操作命令列表显示该应用在指定时间段的所有NoSQL调用的操作命令。

所属应用	NoSQL类型	操作命令	平均耗时小	调用次数小	操作
Spring-Redis	Redis	DEL	3.9ms	7	调用统计 接口快照
Spring-Redis	Redis	RPUSH	0.7ms	21	调用统计 接口快照
Spring-Redis	Redis	LLEN	0.3ms	7	调用统计 接口快照
Spring-Redis	Redis	LINDEX	0.2ms	21	调用统计 接口快照
Spring-Redis	Redis	EVAL	0.1ms	14	调用统计 接口快照

- 1. (可选)在操作命令列表,您可以执行以下操作:
 - 在操作命令右侧操作列,单击调用统计,查看该操作命令的NoSQL调用统计。
 - 在操作命令右侧操作列,单击接口快照,查看该操作命令的接口的快照。
 接口快照说明,请参见接口快照。

3.6.8. 异常分析

本文说明如何查看异常分析,从而了解应用的异常情况。

前提条件

接入应用监控

功能入口

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏,选择应用监控 > 应用列表。
- 3. 在顶部菜单栏,选择地域。
- 4. 在应用列表页面,单击应用名称。
- 5. 在左侧导航栏,单击应用详情。
- 6. 在**应用详情**页面,选择应用实例,设置时间段,单击**异常分析**页签。

arms-k8s-demo V ①⑦ 从M监控			本月 前
⑦ 响应时间 / 请求数 / 错误数 / 异常数 副	概览 JVM监控	③ 主机监控 SQL调用分析 NoSQL调用分析 异常分析 错误分析 上游应用 💶 下游应用 💶 接口使照	
应用分组 全部 ∨ 请输入 Q	异常统计 / 每天		2
arms-k8s-demo	8000		
172.2 1.5min / 227.7K / 384K / 0 0 0 0 0 0 0	6000		
• 172.2 1.5min / 190.2K / 38.4K / 0	2000		
172.2 1.3min / 152.7K / 38.4K / 0 cn-hangzhou-i ecs.g6.starge	0 01-01 00:00	01-14 00:00	01-27 00:00
	错误数 个	异常壊核 (從宗:如果要忽略與常荣笑思,講到応用设置>高度设置T-态段设置T-态界常过滤字段进行设置。)	操作
K	113.4K	com alibeba druid pool. Seriomaerianiimeoriineoriine art com alibeba druid pool. Bruidbarafource gerKonnerianiinermal (Bruidbarafource java 1107)	调用统计 接口快照 译情
		at com alitera, grun por, brilantabource, genomectionurrec (brunnetabource, jewa, lies)	
	25.6K	jern lag Eneption at em alikaka arm dem TartEneptimisodeController, hereEneptimiStartEneptimisodeController.jere:20)	调用统计 接口快服 译情
		as zum reflast. GeneratedBerhodskosennosk innoka Unincom Source) -	

异常统计

异常统计区域显示该应用在指定时间段的异常的堆积柱状图和异常列表。



- 1. (可选)在异常统计区域,您可以执行以下操作:
 - 将光标移到统计图上,查看统计情况。
 - 使用光标选中一段时间,查看指定时间段的统计情况。
 - 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。
 - 单击云图标,查看该指标的API详情。

异常列表

异常列表显示该应用在指定时间段的所有异常。

1. (可选) 在异常列表, 您可以执行以下操作:



⑦ 说明 如果要过滤异常,在应用设置 > 自定义设置 > 高级义设置的异常过滤文本框设置。

- 在异常的操作列,单击调用统计查看该异常的堆积柱状图。
- 在异常的操作列,单击接口快照查看该异常的接口的快照。
 接口快照说明,请参见接口快照。
- 在异常的操作列,单击详情查看该异常的详细信息。

3.6.9. 错误分析

本文说明如何查看错误分析,从而了解应用的错误情况。

前提条件

接入应用监控

功能入口

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏,选择应用监控 > 应用列表。
- 3. 在顶部菜单栏,选择地域。
- 4. 在应用列表页面,单击应用名称。
- 5. 在左侧导航栏,单击应用详情。
- 6. 在**应用详情**页面,选择应用实例,设置时间段,单击错误分析页签。

arms-k8s-demo Y C JVMER						本月	8
⑦ 响应时间 / 请求数 / 错误数 / 异常数 围	概算 JVM监控② 主机监控③	SQL调用分析 NoSQL调用分析 异常分析 错误	分析 上游应用 💶 下游的	应用 🚥 接日	□快照		
ammskis demo 388A Q armskis demo 200% / 61.8K / 51.3K / 0 (minangetros) Exactive Allarge 172	朝田朝文 (南大 4000 1500 8 0-1-01 00:00 01-07 00:00	01-13 00:00 • NTTPA_	€ € HTP	00:00	01	-20 000 01-14 000 01-20 000 • 200 • 200 • 200 • 900	
	产生时间收	接口名称	所属应用	耗时よ	状态	Traceld	操作
	2021-01-01 00:00:03	/demo/queryException/12	arms-k8s-demo	3ms	٠	ac140058160943	查看日志
<	2021-01-01 00:00:03	/demo/invokeConponent	arms-k8s-demo	6ms	•	ac140058160943	査看日志
	2021-01-01 00:00:03	/demo/invokeConponent	arms-k8s-demo	5ms	•	ac140058160943	査看日志
	2021-01-01 00:00:06	/demo/queryException/12	arms-k8s-demo	1ms	•	ac140113160943	查看日志
	2021-01-01 00:00:06	/demo/invokeConponent	arms-k8s-demo	6ms	•	ac140113160943	查看日志
	2021-01-01 00:00:06	/demo/invokeConponent	arms-k8s-demo	6ms	•	ac140113160943	查看日志
	2021-01-01 00:00:06	/demo/invokeConponent	arms-k8s-demo	6ms	•	ac14011316094304069175688d0001	查看日志
	2021-01-01 00:01:06	/demo/invokeConponent	arms-k8s-demo	4ms	٠	ac14011316094304669185689d0001	查看日志

错误数





- 1. (可选)在错误数区域,您可以执行以下操作:
 - 将光标移到统计图上,查看统计情况。
 - 使用光标选中一段时间,查看指定时间段的统计情况。
 - 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。
 - 单击 图标,查看该指标的API详情。

HTTP状态码

HTTP-状态码统计区域显示该应用在指定时间段的HTTP状态码时序曲线。



- 1. (可选)在HTTP-状态码统计区域,您可以执行以下操作:
 - 将光标移到统计图上,查看统计情况。
 - 使用光标选中一段时间,查看指定时间段的统计情况。

- 单击图例,隐藏或显示数据。
- 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。
- 单击 国标, 查看该指标的API详情。

错误列表

错误列表显示该应用在指定时间段的所有错误。

产生时间 🖡	接口名称	所属应用	耗时↓	状态	Traceld	操作
2021-01-28 16:06:06	/demo/queryException/12	arms-k8s-demo	1ms	•	ac1401131611821	查看日志
2021-01-28 16:06:06	/demo/invokeConponent	arms-k8s-demo	4ms	•	ac1401131611821	查看日志
2021-01-28 16:06:06	/demo/invokeConponent	arms-k8s-demo	4ms	•	ac1401131611821	查看日志
2021-01-28 16:06:06	/demo/invokeConponent	arms-k8s-demo	4ms	•	ac1401131611821	查看日志
2021-01-28 16:07:06	/demo/invokeConponent	arms-k8s-demo	5ms	•	ac1401131611821	查看日志
2021-01-28 16:07:35	/demo/queryNotExistDB/11	arms-k8s-demo	2min	•	ac1401131611821	查看日志
2021-01-28 16:08:06	/demo/queryException/12	arms-k8s-demo	1ms	•	ac1401131611821	查看日志
2021-01-28 16:08:06	/demo/invokeConponent	arms-k8s-demo	4ms	•	ac1401131611821	查看日志
2021-01-28 16:08:06	/demo/invokeConponent	arms-k8s-demo	4ms	•	ac1401131611821	查看日志

1. (可选)在错误列表,您可以执行以下操作:

○ 在错误右侧的Traceld列,单击Traceld名称查看该错误的调用链。

• 在错误右侧的操作列,单击查看日志查看该错误的日志。

3.6.10. 上游应用

对于某个应用而言,上游应用是指向该应用发送数据的应用。本文说明如何查看上游应用情况,包括响应时间、请求数、错误数等信息。

前提条件

接入应用监控

功能入口

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏,选择应用监控 > 应用列表。
- 3. 在顶部菜单栏,选择地域。
- 4. 在应用列表页面,单击应用名称。
- 5. 在左侧导航栏,单击应用详情。
- 6. 在**应用详情**页面,选择应用实例,设置时间段,单击上游应用页签。

arms-k8s-demo-subcompone V			班三个月 111
⑦ 响应时间 / 请求数 / 错误数 / 异常数 副	概范 JVM监控② 主机监控③ SQL调用分析 NoSQL调用分析	异常分析 错误分析 上游应用 💶 下游应用 💶 接口快照	
应用分组 全部 ∨ 清输入 Q	全部折動/展开 96歳时间 > 三 輸入应用名称或接口名称	Q 搜索结果数量: 1	
arms-k8s-demo-subcomponent	arms-k8s-demo-subcomponent		请求数: 223939/响应时间: 1.3s/错误数: 22394 💙
• 172. 1.3s / 447.9K / 22.4K / 0	响应时间	请求数	描误数
cn-hangzhou-b ecs.n1.xlarge	1.68	8000	800
• 172.2 1.3s / 446.8K / 22.3K / 0	1.28	4000	003
cn-hangzhou-b ecs.n1.xlarge	800ms	4000	400
	400ms	2000	200
	0ms	0	o
	10-31 00:00 11-23 00:00 12-16 00:00 01-08 00:00 • N的应应时间	10-3100:00 11-2300:00 12-1600:00 01-0800:00 ● 请求数	10-31 00:00 11-23 00:00 12-16 00:00 01-08 00:00 ● 1部英歌

响应时间

响应时间区域显示该应用的上游应用在指定时间段的响应时间时序曲线。

响应时间				
1.6s				
1.2s				
800ms				
400ms				
0ms				
10-31 00:00	11-23 00:00	12-16 00:00 • 响应时间	01-08 00:00	

1. 在响应时间区域,您可以执行以下操作:

○ 将光标移到统计图上, 查看统计情况。

○ 使用光标选中一段时间,查看指定时间段的统计情况。

请求数

请求数区域显示该应用的上游应用在指定时间段的请求数时序曲线。

请求数				
8000				
6000	~			
4000				
2000				
0				
10-31 00:00	11-23 00:00	12-16 00:00 ■ 请求数	01-08 00:00	

- 1. 在请求数区域,您可以执行以下操作:
 - 将光标移到统计图上, 查看统计情况。
 - 使用光标选中一段时间,查看指定时间段的统计情况。

错误数

错误数区域显示该应用的上游应用在指定时间段的错误数时序曲线。



- 1. 在错误数区域, 您可以执行以下操作:
 - 将光标移到统计图上, 查看统计情况。
 - 使用光标选中一段时间,查看指定时间段的统计情况。

3.6.11. 调用链查看

本文说明如何查看调用链,从而了解应用的所有接口的被调用情况,包括产生时间、耗时、状态等信息。

前提条件

接入应用监控

功能入口

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏,选择应用监控 > 应用列表。
- 3. 在顶部菜单栏,选择地域。
- 4. 在应用列表页面,单击应用名称。
- 5. 在左侧导航栏,单击应用详情。
- 6. 在**应用详情**页面,选择应用实例,设置时间段,单击调用链查询页签。

ams-kis-demo V QO AMBR					近三个月	*	
⑦ 响应时间 / 请求数 / 错误数 / 异常数 副	⑦ 彩脸时间/ 读柔微 / 编微微 / 编微微 / 编微微 / 编微微 / 编微微 / 编微微 文 24 编曲的⑦ SQL调用分析 PSSQL调用分析 PSS分析 验证分析 上激组用 400 下激组用 400 正微出映照						
应用分组 全部 ∨ 请输入 Q	输入接口名称搜索	Q					
arms-k8s-demo	产生时间《	按口名称	所属应用	和时业	状态 よ	Traceld	操作
172.2 5.8s / 694.8K / 111.8K / 0 cn-hangzhou-f ecs.sn2ne-Atlarge	2021-01-03 16:33:47	/demo/queryNotExistDB/11	arms-k8s-demo	2.0006mi n	•	ac1401131609	查看日志
172. 5.45 / 7503K / 166.2K / 0 cn-hangzhou-f ecs.sn2ne.Adarge	2021-01-20 18:19:00	/dema/queryNotExistDB/11	arms-k8s-demo	2.0004mi n	•	ac1401131611	查看日志
	2021-01-05 06:39:53	/demo/queryNotExistDB/11	arms-k8s-demo	2.0004mi n	•	ac1401131609	查看日志
	2021-01-04 03:25:48	/demo/queryNotExistDB/11	arms-k8s-demo	2.0004ml n	•	ac1401131609	查看日志
	2021-01-20 19:19:00	/demo/queryNotExistDB/11	arms-k8s-demo	2.0004mi n	•	ac1401131611	查看日志
	2021-01-18 08:08:50	/dema/queryNatExistDB/11	arms-k8s-demo	2.0004mi n	•	ac1401131610	查看日志

接口快照

接口快照页签下显示该应用在指定时间段内被调用的所有接口的列表。

输入接口名称搜索	Q					
/ ⁶⁴ 生时间 小	接口名称	所属应用	耗时↓	状态 ♪	Traceld	操作
2021-01-03 16:33:47	/demo/queryNotExistDB/11	arms-k8s-demo	2.0006mi n	•	ac140113160966	查看日志
2021-01-20 18:19:00	/demo/queryNotExistDB/11	arms-k8s-demo	2.0004mi n	•	ac140113161113	查看日志
2021-01-05 06:39:53	/demo/queryNotExistDB/11	arms-k8s-demo	2.0004mi n	•	ac140113160979	查看日志
2021-01-04 03:25:48	/demo/queryNotExistDB/11	arms-k8s-demo	2.0004mi n	•	ac14011316097(查看日志
2021-01-20 19:19:00	/demo/queryNotExistDB/11	arms-k8s-demo	2.0004mi n	•	ac140113161114	查看日志
2021-01-18 08:08:50	/demo/queryNotExistDB/11	arms-k8s-demo	2.0004mi n	•	ac140113161092	查看日志
2021-01-25 13:55:21	/demo/queryNotExistDB/11	arms-k8s-demo	2.0004mi n	•	ac14011316115	查看日志
2021-01-27 21:37:31	/demo/queryNotExistDB/11	arms-k8s-demo	2.0004mi n	•	ac140113161175	查看日志

- 1. (可选)在接口快照页签下,您可以执行以下操作:
 - 在搜索框, 输入接口名称, 然后单击⊂图标, 查看指定接口的快照。
 - 在接口的Traceld列,单击Traceld名称,查看该接口的调用链路。
 - 在接口的操作列,单击查看日志,查看该接口的日志。

3.6.12. 日志

本文说明如何查看应用的Pod日志,从而了解Pod情况。

前提条件

接入应用监控

⑦ 说明 仅部署在Pod中的应用支持查看Pod日志。

功能入口

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏,选择应用监控 > 应用列表。
- 3. 在顶部菜单栏,选择地域。
- 4. 在应用列表页面,单击应用名称。
- 5. 在左侧导航栏,单击应用详情。
- 6. 在**应用详情**页面,选择Pod实例,设置时间段,单击日志页签。

checkout ど 日 小小監控	最近5分钟
⑦ 响应时间 / 请求数 / 错误数 / 异常数 副	構度 JAMEEN ⑦ 主机监控 Rod监控 SQL调用分析 NoSQL调用分析 异常分析 错误分析 上游应用 400 按口快照 日志
应用分组 全部 ∨ 请输入 Q	最多型示量研制网络5000条日志
checkout	E to
checkout-group 15.3ms / 166 / 0 / 0 continuents	
checkout-gro 14.04ms / 137 / 0 / 0 cn-sharephale ecs.r4.2xlarge checkout-g 13 9ms / 168 / 0 / 0	613 3021-47-28 3021 100 1
cn-shanghai-a esin4.2klarge	(0) 201-4-9 8 4:1:30 201 70 1 - (d=404907030) c + a c list coeffs ign [ListFiber : set AssesStrutger;] (0) 201-4-9 8 4:1:80 71 201 - (d=404907030) c + a c list coeffs ign [ListFiber : set AssesStrutger;] (0) 201-4-9 8 4:1:80 71 201 - (d=404907030) c + a c list coeffs ign [ListFiber : set AssesStrutger;] (0) 201-4-9 8 4:1:80 71 201 - (d=404907030) c + a c list coeffs ign [ListFiber : set AssesStrutger;] (0) 201-4-9 8 4:1:80 71 201 - (d=404907030) c + a c list coeffs ign [ListFiber : set AssesStrutger;] (0) 201-4-9 8 4:1:80 71 201 - (d=404907030) c + a c list coeffs ign [ListFiber : set AssesStrutger;] (0) 201-4-9 8 4:1:80 71 201 - (d=404907030) c + a c list coeffs ign [ListFiber : set AssesStrutger;] (0) 201-4-9 8 4:1:80 71 201 - (d=404907030) c + a c list coeffs ign [ListFiber : set AssesStrutger;] (0) 201-4-9 8 4:1:80 71 201 - (d=404907030) c + a c list coeffs ign [ListFiber : set AssesStrutger;]
	 2014-0-28 34.01 2019 100 1 = [d-4036703303] a. a. disc and i gail (distribute : jet desp54r04071) 2014-0-28 34.01 201 2014 1 = [000+0+0+00] (m correcting (a particle) - Nails of the validate constitution on spale (j phc constraints) [M-60764] (b operations allowed after constitution (dest). 2014-0-28 34.02 (11 300 201 - [000+0+0+00] (m correcting (a particle) - Nails of the validate constitution on spale (j phc constraints) [M-60764] (b operations allowed after constitution (dest). 2014-0-28 34.02 (11 300 201 - [000+0+0+00] (m correcting (a particle) - Nails of the validate constitution correcting (j phc constraints) [M-60764] (b operations) [M-60764] (b op
	301 402 80 40 41 1000 11 1000 11 1000 11 1000 10 10000 10000 1000 1000 1000 10000 10000 1000 10000 100000 10000 1000 100000 1000
	(c) 2017-0-2 W (d. 1.1 KD) = 0 (d. 0.1 KD) (d. 1.1
	000 2021-01-29 09:47:14 347 1990 1 [39-53:0954763562] c.s.n.client.config.inpl.ClientWarker ; prt dampedirosylCongl

日志

日志页签下显示该应用的Pod实例的最新时间的日志。

```
⑦ 说明 最多显示最新时间的5000条日志。
```

最多显示量	最新时间的5000条日志				
				S1	= 0
4972	2021-01-29 08:38:15.723	INFO 1 [dB-63d954763b50] c. a. n. client. config. impl. ClientWorker	: get changedGroupKeys:[]		▲
4973	2021-01-29 08:38:45.231	INFO 1 [dB-63d954763b50] c. a. n. client. config. impl. ClientWorker	: get changedGroupKeys:[]		
4974	2021-01-29 08:39:14.738	INFO 1 [d8-63d954763b50] c. a. n. client. config. impl. ClientWorker	: get changedGroupKeys:[]		
4975	2021-01-29 08:39:44.879	INFO 1 [dB-63d954763b50] c. a. n. client. config. impl. ClientWorker	: get changedGroupKeys:[]		
4976	2021-01-29 08:40:14.529	INFO 1 [d8-63d954763b50] c. a. n. client. config. impl. ClientWorker	: get changedGroupKeys:[]		
4977	2021-01-29 08:40:44.037	INFO 1 [dB-63d954763b50] c. a. n. client. config. impl. ClientWorker	: get changedGroupKeys:[]		
4978	2021-01-29 08:41:13.670	INFO 1 [d8-63d954763b50] c. a. n. client. config. impl. ClientWorker	: get changedGroupKeys:[]		
4979	2021-01-29 08:41:43.312	INFO 1 [d8-63d954763b50] c. a. n. client. config. impl. ClientWorker	: get changedGroupKeys:[]		
4980	2021-01-29 08:42:12.842	INFO 1 [dB-63d954763b50] c. a. n. client. config. impl. ClientWorker	: get changedGroupKeys:[]		
4981	2021-01-29 08:42:42.352	INFO 1 — [d8-63d954763b50] c.s.n.client.config.impl.ClientWorker	: get changedGroupKeys:[]		
4982	2021-01-29 08:43:11.859	INFO 1 [dB-63d954763b50] c. a. n. client. config. impl. ClientWorker	: get changedGroupKeys:[]		
4983	2021-01-29 08:43:41.368	INFO 1 — [d8-63d954763b50] c. a. n. client. config. impl. ClientWorker	: get changedGroupKeys:[]		
4984	2021-01-29 08:44:10.879	INFO 1 [d8-63d954763b50] c. a. n. client. config. impl. ClientWorker	: get changedGroupKeys:[]		
4985	2021-01-29 08:44:40.387	INFO 1 [d8-63d954763b50] c. a. n. client. config. impl. ClientWorker	: get changedGroupKeys:[]		
4986	2021-01-29 08:45:09.898	INFO 1 [d8-63d954763b50] c. a. n. client. config. impl. ClientWorker	: get changedGroupKeys:[]		
4987	2021-01-29 08:45:39.411	INFO 1 [d8-63d954763b50] c. a. n. client. config. impl. ClientWorker	: get changedGroupKeys:[]		
4988	2021-01-29 08:46:08.920	INFO 1 [d8-63d954763b50] c. a. n. client. config. impl. ClientWorker	: get changedGroupKeys:[]		
4989	2021-01-29 08:46:38.428	INFO 1 [d8-63d954763b50] c. a. n. client. config. impl. ClientWorker	: get changedGroupKeys:[]		
4990	2021-01-29 08:47:07.936	INFO 1 — [d8-63d954763b50] c.s.n.client.config.impl.ClientWorker	: get changedGroupKeys:[]		
4991	2021-01-29 08:47:37.446	INFO 1 [d8-63d954763b50] c. a. n. client. config. impl. ClientWorker	: get changedGroupKeys:[]		
4992	2021-01-29 08:48:06.954	INFO 1 — [d8-63d954763b50] c. a. n. client. config. impl. ClientWorker	: get changedGroupKeys:[]		
4993	2021-01-29 08:48:36.464	INFO 1 [d8-63d954763b50] c. a. n. client. config. impl. ClientWorker	: get changedGroupKeys:[]		
4994	2021-01-29 08:49:05.973	INFO 1 [d8-63d954763b50] c. a. n. client. config. impl. ClientWorker	: get changedGroupKeys:[]		
4995	2021-01-29 08:49:35.480	INFO 1 [dB-63d954763b50] c. a. n. client. config. impl. ClientWorker	: get changedGroupKeys:[]		
4996	2021-01-29 08:50:04.994	INFO 1 [d8-63d954763b50] c. a. n. client. config. impl. ClientWorker	: get changedGroupKeys:[]		
4997	2021-01-29 08:50:34.512	INFO 1 [dB-63d954763b50] c. a. n. client. config. impl. ClientWorker	: get changedGroupKeys:[]		
4998	2021-01-29 08:51:04.023	INFO 1 [d8-63d954763b50] c. a. n. client. config. impl. ClientWorker	: get changedGroupKeys:[]		
4999	2021-01-29 08:51:33.554	INFO 1 [d8-63d954763b50] c. a. n. client. config. impl. ClientWorker	: get changedGroupKeys:[]		
5000	2021-01-29 08:52:03.069	INFO 1 [d8-63d954763b50] c. a. n. client. config. impl. ClientWorker	: get changedGroupKeys:[]		·

1. 在日志页签下的搜索框,您可以输入关键字,单击 图标,筛选日志。

3.6.13. 内存快照

JVM监控可以直观展示指定时间段内的多项内存指标,虽然图表能体现出内存使用量过大的情况,但无法显示具体信息,因此不能帮助您排查问题产生的原因。此时您可以创建内存快照,通过详细的日志查看内存占用的详细信息,帮助您排查内存泄漏和内存浪费等内存问题。

创建内存快照

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏选择应用监控 > 应用列表,并在顶部菜单栏选择目标地域。
- 3. 在应用列表页面单击目标应用名称。
- 4. 在左侧导航栏单击应用详情,在页面右侧单击JVM监控页签。
- 5. 在JVM监控页签右上角,单击创建内存快照。

⑦ 说明 如果单击创建内存快照时,上一个快照任务仍在运行,则系统会弹出错误消息。请您 耐心等待上一个快照任务运行完毕。目前仅支持为Linux系统新建内存快照。

6. 在创建内存快照对话框中选择一个IP,并单击保存。

警告 快照任务的运行时间从几分钟到半小时不等。应用进程在转储期间会停止响应,请谨慎
使用。

创建内存快照⑦ 内存快照 >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>	(
* IP:	
获取和分析全部内存快照需要1~20分钟,请稍后查看结果。 创建快照会导致Java进程停顿,请谨慎操作。 Memory Dump功能不支持Alphine系统。 2.7.0及以上版本,使用新版本内存分析功能需要联系 ARMS服务号,钉钉号:arms160804。	
保存	

⑦ 说明 如果在应用详情页面左侧已选择目标实例,则IP字段会默认选中该实例的IP地址。

查看内存快照详情

1. 在JVM监控页签右上角,单击历史快照。

在**快照任务数量**面板展示了任务执行状态:绿色表示快照任务执行成功,蓝色表示快照任务执行中, 红色表示快照任务执行失败。

快照任务名称的信息依次为:

- 内存分析状态
- 由IP地址和时间戳组成的快照任务ⅠD

快照创建时间



- 2. 单击开始分析,弹出提示对话框,单击确认。
- 3. 单击分析结果,打开内存分析页面查看内存分析详情,帮助您查找内存泄漏信息,用于减少内存消耗。
 - 单击概况页签,查看堆使用大小、类数量、对象数量、类加载器数量和根对象数量等统计信息,以及 以环形图显示的内存占用情况。

< 概况	泄露报表	GC 根对象	支配树	类视图	不可达类视图	重复类视图	类加载器视图	堆外内有 >
堆使用大	·/\\		38.6 N	ИB				
类数量			13.54	k				
对象数量	t		668.6	k				
类加载器	数量		166					
根对象数	量		3.285	k				
创建时间]		2020-	04-20 17:59:4	11			
操作系统	位数		64 bit					
Biggest Obje 堆使用大小:	acts by Retained	SIZE						~

 单击泄露报表页签,可以查看消耗过多内存的可疑对象。单击页面下方的Problem Suspect查看可 疑对象的具体实例、内存占用的情况以及类加载信息。



○ 单击GC根对象页签,查看按根类型和Java类类型分类的所有根对象(被静态变量或线程栈等GC根引用的对象)。

< 概況	泄露报表	GC 根对象	支配树	类视图	不可达类视图	重复类视图	类加载器视图	堆外内有 >
Class Name								
> 👸 Sys	tem Class							
> 💡 Th	read							
> 👸 JN	Global							
∨ <mark>o</mark> e Bu	sy Monitor							
~ 0	java.util.TaskQueu	e						
~ [) java.util.TaskQu	ieue @ 0x803186b8						
~	class> class	s java.util.TaskQueu	e @ 0x8003aa5	8 System Class	;			
	> 🔂 <class> 🤉</class>	class java.lang.Class	@ 0x80168350	System Class				
	> 🚺 <classloa< td=""><td>ader> java.lang.Clas</td><td>sLoader @ 0x0</td><td><system class="" l<="" td=""><td>oader></td><td></td><td></td><td></td></system></td></classloa<>	ader> java.lang.Clas	sLoader @ 0x0	<system class="" l<="" td=""><td>oader></td><td></td><td></td><td></td></system>	oader>			
	> 🔂 <super></super>	class java.lang.Obje	ct @ 0x8026c4	d0 System Clas	5			
	Σ 3/3							
>	queue java.u	util.TimerTask[128] @	0x803186d0					
	∑ 2/2							
> [java.util.TaskQu	ueue @ 0x81382c98						
> [java.util.TaskQu	ieue @ 0x81383658						
Σ	3/3							
> 0	java.lang.ref.Refer	enceQueue\$Lock						
> 0	org.apache.derby.	impl.services.daemo	on.BasicDaemo	n				
> 0	java.lang.ref.Refer	ence\$Lock						
Σ4	/ 4							

单击**支配树**页签,查看堆中对象之间的支配关系,可以识别出占用了大量内存的对象和它们之间的依赖关系。

概况	泄露报表	GC 根对象	支配树	类视图	不可达类视图	重复类视图	类加载器视图	堆外内存视图	系统属性	线程信息	OQL	
						Object	Class					
Class Na	me						Shallow Heap	Retain	ed Heap	Perc	entage	
> 🔓	class java.lang.re	f.Finalizer @ 0x8002	27b68 System (Class			16	20440)344	37.	35%	
> 0	org.springframe	work.boot.loader.La	unchedURLClas	sLoader @ 0x80)59d420		80	25452	296	4.6	5%	
> 🔎	class java.beans.	ThreadGroupConte	xt @ 0x8096077	78 System Class			8	23960	064	4.3	8%	
> 🔎	class sun security	y.provider.X509Facto	ory @ 0x82629	580 System Cla	55		24	13722	188	2.5	1%	
> 🗅	com.navercorp.p	inpoint.bootstrap.ir	nterceptor.regis	try.DefaultInter	eptorRegistr		24	12304	148	2.2	5%	
> 🗅	org.springframe	work.beans.factory.s	support.Default	ListableBeanFac	tory @ 0x80d		208	12274	148	2.2	4%	

• 单击**类视图**页签,查看各个类型的堆使用情况和对应实例的数量。

概况	泄露报表	GC 根对象	支配树	类视图	不可达类视图	重复类视图	类加载器视图	堆外内存视图	系统属性	线程信息	OQL	
Class N	ame							Objects	Shallow	Heap	Retained I	Heap
🕒 cha	ar[]							91159	126779	20	>= 1267	7920
G byt	te[]							8157	616801	6	>= 61680	016
G jav	a.lang.String							83899	201357	6	>= 10194	4800
G jav	a.lang.reflect.Meth	od						17517	154149	6	>= 24653	328
G jav	a.util.concurrent.C	oncurrentHashMap!	\$Node					46327	148246	4	>= 62173	384
G int	0							8302	932040		>= 93204	40
G jav	a.util.HashMap\$N	ode[]						9608	798424		>= 5256	120
🕒 jav	a.util.LinkedHashN	lap\$Entry						19338	773520		>= 21114	416

• 单击不可达类视图页签,查看堆中未被引用的对象的大小及类型情况。

概況	泄露报表	GC 根对象	支配树	类视图	不可达类视图	重复类视图	类加载器视图	堆外内存视图	系统属性	线程信息	OQL	***
Class Na	me							Objects		Shallow Heap		
G char	0							948174		110867488		
G byte	20							462932		38622904		
G java	.lang.Object[]							126454		31227728		
G int[]								39009		22313496		
G java	lang.String							694689		16672536		
G java	.util.HashMap\$N	ode						191176		6117632		
G byte	:00							129427		3779064		

• 单击重复类视图页签, 查看被多个类加载器加载的类型情况。

概况	泄露报表	GC 根对象	支配树	类视图	不可达类视图	重复类视图	类加载器视图	堆外内存视图	系统属性	线程信息	OQL	***
Class N	ame / Class Loade	r					ClassLoader	Counts	Defined Classes	Num	of Instances	
> 0	java.lang.invoke.l	.ambdaForm\$DMH					54					
> G	java.lang.invoke.l	.ambdaForm\$MH					32					
> 0	java.lang.invoke.l	.ambdaForm\$BMH					6					
> G	com.navercorp.pi	inpoint.plugin.jdbc.i	mysql.MySqlCo	nstants			3					
> G	com.google.com	mon.base.Function					2					
> G	com.google.com	mon.base.Predicate					2					
> 0	com.google.com	mon.cache.Cache					2					

单击**类加载器视图**页签,查看应用当前使用到的所有类加载,以及每个类加载器加载类的情况。例如加载了哪些类,这些类有多少实例等。

概況	泄露报表	GC 根对象	支配树	类视图	不可达类视图	重复类视图	类加载器视图	堆外内存视图	系统属性	线程信息	OQL	•••
Class N	ame							Defined Classes		No. of Insta	nces	
> 6	org.springframew	ork.boot.loader.Lau	inchedURLClass	Loader @ 0x80	59d420			7768		72905		
> 8	<system class="" loa<="" td=""><td>der></td><th></th><td></td><th></th><td></td><td></td><td>3529</td><td></td><td>991784</td><td></td><td></td></system>	der>						3529		991784		
> 6	com.navercorp.pi	npoint.bootstrap.cla	assloader.Parall	elCapablePinpo	intURLClassLoader @ 0	x800d8fa8		2422		46619		
> 6	com.navercorp.pi	npoint.common.plu	gin.PluginLoad	erClassLoader (0x80341058			84		0		
> 6	sun.misc.Launche	r\$AppClassLoader (@ 0x800d9008					66		1934		
> 6	com.navercorp.pi	npoint.common.plu	gin.PluginLoad	erClassLoader (0x804336a8			36		29		
> 6	sun.misc.Launche	r\$ExtClassLoader @	0x800d9068					35		2		
> 6	com.navercorp.pi	npoint.common.plu	gin.PluginLoad	erClassLoader @	0x80435870			19		19		

○ 单击**堆外内存视图**页签,查看应用目前使用的所有java.nio.DirectByteBuffer对象以及对应的堆外内存信息,用于排查由堆外内存导致物理内存消耗过多的问题。

概況	泄露报表	GC 根对象	支配树	类视图	不可达类视图	重复类视图	类加载器视图	堆外内存视图	系统属性	线程信息	OQL	•••
Label								position	limit		capacity	
🗋 java	.nio.DirectByteBu	ffer @ 0x6cc1daae0						0	65536		65536	
🗋 java	.nio.DirectByteBu	ffer @ 0x6cc042658						0	65536		65536	
🗋 java	.nio.DirectByteBu	ffer @ 0x6cc009740						0	65536		65536	
🗋 java	.nio.DirectByteBu	ffer @ 0x6ccac5c90						0	16384		16384	
🗋 java	.nio.DirectByteBu	ffer @ 0x6cd019790)					0	3871		4283	
🗋 java	.nio.DirectByteBu	ffer @ 0x6cd06efb0						0	1872		1875	
🗋 java	.nio.DirectByteBu	ffer @ 0x6cd03c0a0						0	358		1875	
🗋 java	.nio.DirectByteBu	ffer @ 0x6cd4b3b9	В					16	16		1024	

• 单击**系统属性**页签,查看系统参数和环境变量。

概況	泄露报表	GC 根对象	支配树	类视图	不可达类视图	重复类视图	类加载器视图	堆外内存视图	系统属性	线程信息	OQL	
Key					Value					输入关键	字搜索	
awt.to	olkit				sun.awt.X11.XToolki	t						
arms.a	igent.args											
java.sp	ecification.vers	ion			1.8							
file.en	coding.pkg				sun.io							
sun.cp	u.isalist											
sun.jn	u.encoding				UTF-8							
java.cl	ass.path				/home/admin/sprin	gboot-undertow-	1.0.0-SNAPSHOT.jar;	/home/admin/.opt/A	rmsAgent/arms	-bootstrap-1.7	0-SNAPSHO	OT.jar
java.v	n.vendor				Oracle Corporation							
sun.ar	ch.data.model				64							
arms.a	igent.env				ACSK8S							
arms.v	rersion				1.7.0-SNAPSHOT							
java.ve	endor.url				http://java.oracle.co	im/						

单击线程信息页签,查看线程的相关信息,例如线程名、堆占用情况、调用栈信息、局部变量等。通过该视图您可以分析线程过多、死锁、调用栈过深等问题。

概况	泄露报表	GC 根对象	支配树	类视图	不可达类视图	重复类视图	美加載器视图	堆外内存	视图 系统属	性 线程信息	OQL	***	
Thread	/ Stack				Name			Shallow Heap	Retained Heap	Context Class Loade			Is Daemon
> @	org.xnio.nio.Wor	kerThread @ 0x6cc	b73470		XNIO-2 I/O-13			152	68120	org.springframewor	k.boot.loader.La	unchedURLClassLoader @ 0x6cc504	false
> @	org.xnio.nio.Wor	kerThread @ 0x6cc	ca38c0		XNIO-2 I/O-1			152	67984	org.springframewor	k.boot.loader.La	unchedURLClassLoader @ 0x6cc504	false
> @	org.xnio.nio.Wor	kerThread @ 0x6cc	c25e00		XNIO-2 I/O-5			152	67984	org.springframewor	k.boot.loader.La	unchedURLClassLoader @ 0x6cc504	false
> @	org.xnio.nio.Wor	kerThread @ 0x6cc	c26e88		XNIO-2 I/O-3			152	67952	org.springframewor	k.boot.loader.La	unchedURLClassLoader @ 0x6cc504	false
> @	org.xnio.nio.Wor	kerThread @ 0x6cc	bd55f8		XNIO-2 I/O-7			152	67952	org.springframewor	k.boot.loader.La	unchedURLClassLoader @ 0x6cc504	false
> @	org.xnio.nio.Wor	kerThread @ 0x6cc	bd4570		XNIO-2 I/O-9			152	67952	org.springframewor	k.boot.loader.La	unchedURLClassLoader @ 0x6cc504	false
> @	org.xnio.nio.Wor	kerThread @ 0x6cc	b745a8		XNIO-2 I/O-11			152	67952	org.springframewor	k.boot.loader.La	unchedURLClassLoader @ 0x6cc504	false
> @	org.xnio.nio.Wor	kerThread @ 0x6cc	b44130		XNIO-2 I/O-15			152	67952	org.springframewor	k.boot.loader.La	unchedURLClassLoader @ 0x6cc504	false
> @	org.xnio.nio.Wor	kerThread @ 0x6cc	86dde0		XNIO-1 I/O-1			152	67168	org.springframewor	k.boot.loader.La	unchedURLClassLoader @ 0x6cc504	true

○ 单击OQL页签,查询堆中的各种信息,例如所有长度大于2000的字符串。

概况	泄露报表	GC 根对象	支配树	类视图	不可达类视图	重复类视图	类加载器视图	堆外内存视图	系统属性	线程信息	OQL	•••
🖉 sele	ect * from java.l	ang.String s wher	e s.value.@lei	ngth > 2000							(8)	Q
												OQL Help
Class Na	ime									Shallow Heap	Retained	Heap
	java.lang.String	@ 0xe2446aa0 /usr/	lib/jvm/java-1.8	.0-openjdk-1.8.	0.65-3.b17.1.alios7.x86_	64/jre/bin/java -cp /ł	nome/admin/ArmsAger	it/lib/eagleeye-common	-1.2.5.1-ARMS	24	11992	*
	java.lang.String	@ 0x816450d8 \u22	ec\u22ed\u22e	e\u22ef\u22f0\u	u22f1\u22f2\u22f3\u22f	4\u22f5\u22f6\u22f7	\u22f8\u22f9\u22fa\u2	2fb\u22fc\u22fd\u22fe\i	u22ff\u2300\u2	24	8232	
	java.lang.String	@ 0x816450c0 \u6a	bd\u6abe\u6ab	f\u6ac0\u6ac1\ı	ибас2\ибас3\ибас4\иба	ic5\u6ac6\u6ac7\u6a	c8\u6ac9\u6aca\u6acb\	ubacc\ubacd\ubace\uba	cf\u6ad0\u6ad	24	8232	
	java.lang.String	@ 0x816450a8 \ucd	cb\ucbcd\ucaca	\uccd3\ue5cc\u	e5cb\uc4e6\ude9a\ude	9b\ud1a1\ud1b7\ue	5cd\ude9c\ue5d0\ude9	ld\ucdb8\ud6f0\ue5cf\u	b5dd\ude9e\u	24	8232	

3.7. 定时任务

定时任务可以在固定的间隔时间执行指定的程序或者指令,应用监控的定时任务功能可以帮助您了解定时任务的详细情况,包括概览、SQL调用分析、NoSQL调用分析、异常分析、错误分析、链路下游和调用链查询。

⑦ 说明 ARMS应用监控目前仅支持展示XXL-JOB和SchedulerX类型的定时任务。

功能入口

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏,选择应用监控 > 应用列表。
- 3. 在顶部菜单栏,选择地域。
- 4. 在应用列表页面,单击应用名称。
- 在左侧导航栏,单击定时任务。
 在定时任务页面左侧列表显示了当前应用下的所有定时任务。

⑦ 运行时间 / 运行次数 / 异常数 目.	概选 SQL调用分析 NoSQL调用分析 异常分析 错误分析 链路下路 🚥 调用链查询	
请输入 Q		錼
• SchedulerX 300 1.10min / 103 / 0		
• XXL-JOB GLUE 11.7ms / 347 / 0	TAPIL	
• XXL-JOB BEAN 1.2ms / 347 / 347		
• XXL-JOB GLUE 0.9ms / 347 / 0		
• XXL-JOB BEAN 0.3ms / 347 / 0	1.44次/小时 MYSQL	
	平均1.46ms	
	运行次数 / 3小时 ▲ ビ 国 筆 平均时间 / 3小时 点击曲线兜击点滚空分析 120 13min 13min 13min 1min 1min 205 000 000 000 000 000 000 000 000 000	2 K 5 *
	07-27 00.00 07-28 00.00 07-29 00.00	:00

6. 单击目标定时任务,可以查看对应的定时任务详情。

↓ 注意 定时任务功能要求Agent版本升级至2.7.1.3及以上。升级Agent的具体操作,请参见升级探
 针。

查看概览信息

概览页签可以查看目标定时任务的详细调用拓扑,以及运行次数、平均时间、运行异常次数和调度延迟时间 的时序曲线。



查看SQL和NoSQL调用分析

SQL调用分析页签和**NoSQL调用分析**页签展示了左侧选中的定时任务所发起的SQL和NoSQL请求列表。借助此页签,您可以找出是哪一个SQL或NoSQL造成某个服务过慢。

概览	SQL调用分析	NoSQL调用	分析 异	常分析	错误分析	链路下游 🚥	调用链查询						
so	QL调用统计												
1	2 15 16 0 0 07-27 15.30 07-27 15.40 07-27 15.40 07-27 15.40 07-27 15.40												
改据库表	请选择数据库表	✓ 操作名:	请选择操作	馆 💙									
所属应	涌	数	国车类型	5	iQL语句				慢5QL执行次数♪	平均耗时 🖡	调用次数 🖡	返回大小	操作
	test-shanghai		ysql		SELECT user, I	host FROM user wł	here user = "root";		0	1.2ms	13	32.00 B	调用统计 调用链查询

- 1. (可选)在SQL或NoSQL语句列表,您可以执行以下操作:
 - 单击操作列调用统计,查看该SQL或NoSQL语句的调用时序曲线。
 - 单击操作列调用链查询,查看该SQL或NoSQL语句的调用链。更多信息,请参见调用链查询。

查看异常分析

异常分析页签展示了左侧选中定时任务的代码段内所造成的Java异常。



- 1. (可选) 在异常列表, 您可以执行以下操作:
 - 单击操作列调用统计,查看该异常的时序曲线。
 - 单击操作列调用链查询,查看该异常的调用链。更多信息,请参见调用链查询。

查看错误分析

错误分析页签展示了目标定时任务产生错误的统计数据。

概览	SQL调用分析	NoSQL调用分析	异常分析	错误分析	链路下游 💷	调用链查询	1					
错误数	女/每分钟											~ 🗠 🖂
3 -												
2.25 -												
1.5												
0.75												
0 -												
07-27 1	15:30			07-27 15:34			07-27 1 • XXL-JOB	15:38			07-27 15:42	
产生时间	4	接口	名称			į	新属应用		耗时♪	状态	Traceld	操作
2021-07-	27 15:31:03	BEAN	i(Class)-demoClas	sslobHandler		1	-test-shanghai		Oms	٠	WINDOW NO. INC.	宣誓日志
2021-07-	27 15:31:23	BEAN	I(Class)-demoCla	ssJobHandler			-test-shanghai		1ms	•		宣誓日志
2021-07-	27 15:31:43	BEAN	I(Class)-demoCla	ssJobHandler			-test-shanghai		1ms	٠		查看日志
2021-07-	27 15:32:03	BEAN	I(Class)-demoCla	ssJobHandler			-test-shanghai		1ms	٠		宣看日志

- 1. (可选)在错误列表,您可以执行以下操作:
 - 在错误右侧的Traceld列,单击Traceld名称查看该错误的调用链。更多信息,请参见调用链路查询。
 - 在错误右侧的操作列,单击查看日志查看该错误的日志。

查看链路下游的调用情况

链路下游页签列出了应用下游(被应用调用的一方)的接口及其调用性能指标,包括响应时间、请求数和错误数。

全部折疊/展开 响应时间 > 目 输入应用名称或接口名称	Q 搜索结果数量: 1	
springboot-server		请求数: 388386 / 响应时间: 2.006s / 错误数: 388386 🗸 🗸
响应时间	请求数	错误数
2.4s	4800	4800
1.85	3600	3600
1.2s	2400	2400
600ms	1200	1200
0ms	0	0 10-31 00:00 11-23 00:00 12-16 00:00 01-08 00:00 ● 错误数

在链路下游页签上,可按需执行以下操作:

- 在页签顶部单击**全部折叠/展开**,即可折叠或展开下方的所有接口。
- 在页签顶部的搜索框内输入应用名称或接口(Span)名称的关键字,并单击搜索图标,即可筛选出符合 条件的接口。
- 单击接口信息所在的折叠面板,或者单击行末的上箭头或下箭头,即可展开或折叠该接口的性能指标信息。

调用链查询

调用链查询页签展示了目标定时任务的调用链详情,以及根据该任务生成的总调用链次数和调用链响应时间 两个图表。



- 在总调用链次数图表中(图示中①),超过500毫秒被定义为慢调用链。慢调用链受应用设置中阈值影响,但不包含因为异步产生的调用本地API的子调用链。
- 在调用链响应时间图表中(图示中②),其展示的数据是基于接口调用链数据的二次统计,并受采样率影响。
- 在调用链详情表格中(图示中③),单击Traceld可以查看调用链路和业务轨迹。单击查看日志可以查看 定时任务的调用日志。

3.8. 接口调用

本功能用于监控应用下的接口调用详情,包括SQL调用分析、NoSQL调用分析、异常分析、错误分析、链路 上下游和调用链查询。

功能入口

按照以下步骤进入应用接口调用监控功能。

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏选择应用监控 > 应用列表,并在顶部菜单栏选择目标地域。
- 3. 在应用列表页面选择您想查看的应用。
- 4. 在左侧导航栏中单击接口调用。

支持的框架

本功能模块可自动发现和监控以下Web框架和RPC框架中提供的接口:

- Tomcat 7+
- Jetty 8+
- Resin 3.0+
- Undertow 1.3+
- WebLogic 11.0+
- SpringBoot 1.3.0+
- HSF 2.0+
- Dubbo 2.5+

查看接口概览

在概览页签上可以查看目标接口的详细调用拓扑,以及请求数、响应时间、错误数和HTTP-状态码统计的时序曲线。

	(AP) #3	観2:0 裁2:1558	
	/tb	(到)时: 18.08ms	
for the second se	0	QC	
青求数 / 每分钟	~⊻⊡ ≋	响应时间 / 每分钟 点击曲线突端点深度分析	2 K 🗆 1
120		24ms	
90		18ms	
0		12ms	
30		6ms	
0 15-06 11:17 05-06 11:11 05-06 11:15 05-06 11:19 ● HTPA⊡		0ms 05-06 11:07 05-06 11:11 05-06 11:15 ・HTTP入口	05-06 11:19
當県数 / 每分钟 点击曲线突遣点深度分析	~⊻⊡ *	HTTP-状态码统计	~ []
2		120	
		90	
1		60	
		30	
		0	(
.o-usii.uz US-Usii:11 US-Usii:15 05-06 11:19		up-up ii.u/ Up-U6 II:II Up-U6 II:I5	03-06 (1117

查看SQL和NoSQL调用分析

在SQL调用分析页签和NoSQL调用分析展示了左侧选中服务的代码段内所发起的SQL和NoSQL请求列表。 借助此页签,您可以找出是哪一个SQL或NoSQL造成某个服务过慢。您还可以单击某个SQL或NoSQL中的调 用链查询来查看一个SQL或NoSQL执行逻辑所处的完整代码链路。



查看异常分析

在**异常分析**页签展示了左侧选中服务的代码段内所抛出的Java异常。您还可以单击某个异常中的**调用链查 询**来查看一个异常堆栈所处的完整代码链路。



查看错误分析

在错误分析页签展示了应用的错误和HTTP状态码统计数据。您还可以单击需要查看的Traceld,即可在新页 面查看调用链路相关信息。

攝現数 / 毎分钟 2 1		~ <u>~</u>	F H	TTP-状态 20 90 60 30		
05-06 11:23 05-06 11:27	05-06 11:31 0 • HTTP入口	05-06 11:35	05	-06 11:23	05-06 11:27 05-06 11:31 05-06 11:35 • 200 • 2XX • 3XX • 4XX • 5XX	
产生时间小	接口名称	所屬应用	耗时 小	状态	Traceld	操作
2022-05-06 11:25:35	/tb/i	Rentered	23ms	٠	ea	查看日志
2022-05-06 11:25:37	/tb/i	Rentered	15ms	٠	23	查看日志
2022-05-06 11:29:06	/tb/	Rentered	27ms	٠	ea	查看日志
2022-05-06 11:29:16	/tb/	Rentwood	21ms	٠	ea	查看日志
2022-05-06 11:33:05	/tb/i	Rentered	19ms	٠	ea	查看日志
2022-05-06 11:33:10	/tb/i	Former	19ms	٠	ea	查看日志

查看链路上游和链路下游的接口调用情况

链路上游和链路下游页签分别列出了应用上游(调用应用的一方)和应用下游(被应用调用的一方)的接口及其调用性能指标,包括响应时间、请求数和错误数。

全部折叠/展开 响应时间 ✓ 三Ⅰ 输入应用名称或被口名称	Q 搜索结果数量: 2	
Cloth /product/		请求数: 1507 / 响应时间: 2.7ms / 错误数: 6 🗸 🗸
间也这时间	请求数	错误数
3.2ms	120 90	2
1.6ms 0.8ms	60 30	0.5
0ms 05-06 11:25 05-06 11:29 05-06 11:33 05-06 11:37 ● 網应對問	0 05-06 11:25 05-06 11:29 05-06 11:33 05-06 11:37 ● 清水政	0

在链路上游和链路下游页签上,可按需执行以下操作:

- 在页签顶部单击**全部折叠/展开**,即可折叠或展开下方的所有接口。
- 在页签顶部的搜索框内输入应用名称或接口(Span)名称的关键字,并单击搜索图标,即可筛选出符合 条件的接口。
- 单击接口信息所在的折叠面板,或者单击行末的上箭头或下箭头,即可展开或折叠该接口的性能指标信息。

查看调用链查询

调用链查询页签展示了该服务接口中的参数详情,以及根据该参数详情生成的调用链次数和调用链响应时间 两个图表。



- 总调用链次数图表显示了总调用链次数和慢调用链次数。调用时间超过500毫秒则被定义为慢调用链,受应用设置中阈值影响,不含因为异步产生的子调用链(调用本地API)。
- 调用链响应时间图表数据基于调用链查询数据(采样明细)二次统计,受采样率影响。
- 在调用链列表中,单击Traceld可以查看调用链路和业务轨迹。单击查看日志可以查看该接口的调用日志。

3.9. 事件中心

事件中心将云产品所生成的事件数据进行统一管理、存储、分析和展示,已接入EDAS的变更事件、ARMS的 报警事件、0-1事件(如死锁、OOM和应用启动等)、MSE的微服务管控事件和K8s集群事件。当您的应用使 用了相关的产品,对应的事件会自动接入事件中心进行统一的分析展示,方便您查看与分析。

事件模型

事件中心的一个事件主要由以下参数来定义:

参数	是否必须	描述
source	是	事件来源
type	是	事件类型
level	是	事件等级
time	是	事件发生时间
data	是	事件体(一般为JSON格式)
PID	否	Pod ID
IP	否	IP地址
ClusterId	否	集群ID
PodName	否	Pod名称

进入事件中心

1. 登录ARMS控制台。

2. 在左侧导航栏选择应用监控 > 事件中心,并在顶部菜单栏选择目标地域。

事件中心模块介绍

事件中心主要包含典型事件和四个页签(分别是普通视图,拓扑视图、集群视图和订阅规则)。

- 典型事件: 展示系统预置的典型事件的数量。更多信息, 请参见典型事件。
- 普通视图将与当前应用关联的所有事件进行简单的多维度分析与展示。更多信息,请参见普通视图。
- 拓扑视图将当前应用关联的事件和与当前应用的资源拓扑图进行结合展示。更多信息。请参见拓扑视图。
- 集群视图将K8s集群关联的所有事件进行简单的多维度分析与展示。更多信息。请参见集群视图。
- 订阅规则以列表形式展示了当前您创建的订阅规则。更多信息,请参见订阅规则。

典型事件

典型事件:展示系统预置的典型事件类型在最近30分钟(可在右上角调整时间范围)内发生的次数。在事件 下方单击**订阅**,可以编辑该事件的订阅规则。订阅规则的操作,请参见订阅规则。

应用变更失败 ⑦	应用扩缩容 ⑦	应用扩缩容到达上下限②	离群摘除 ⑦	pod启动失败 ②	鏡像拉取失败 ②	POD被驱逐 ⑦	POD OOM (?)	K8S集群资源不足 ⑦	K8S节点OOM ②	K8S节点重启 ⑦	K8S节点FD不足 ②
0	0	0	0	1225	0	0	11	0	0	0	0
次數(7天)::1	次数(7天)::54	次数(7天)::0	次数(7天)::0	次数(7天)::2749	次数(7天)::0	次數(7天)::0	次数(7天)::19	次数(7天)::0	次数(7天)::0	次数(7天)::0	次数(7天)::0
次数(30天)::1	次数(30天)::60	次数(30天)::0	次数(30天)::0	次数(30天)=12110	次数(30天)::0	次数(30天)=0	次数(30天)=19	次数(30天)::0	次数(30天)=0	次数(30天)::0	次数(30天)::0
订阅	订阅	រោល	订问	已订阅	订间	订阅	订问	订问	订问	订问	订问

普通视图

普通视图可以按照您指定的检索条件进行搜索,搜索的结果会以四个视图进行展示,分别是:最近两周事件热力图、不同来源事件占比、不同来源事件数量走势和事件详情。



- 最近两周事件热力图展示近两周内满足过滤条件的事件发生次数按小时统计的热力分布。颜色越深, 说明该小时内发生的事件数量越多。
- 不同来源事件占比展示不同来源事件的数量占比。
- 不同来源事件数量走势展示不同来源事件在选定时段内的走势。
- **事件详情**展示当前所有事件的详情列表。

普通视图查看说明:

- 单击最近两周事件热力图中的热力方块,可查看该小时内的事件详情。
- 单击不同来源事件数量走势中的立柱,可在事件详情区域查看对应时段内的所有事件列表。
- 单击事件详情右侧的查看,可以查看事件的具体内容。
- 单击事件详情右侧的订阅,可以订阅指定事件。更多信息,请参见订阅规则。

拓扑视图

拓扑视图首先会绘制出该应用的资源拓扑,包含该应用使用的ECS,该应用部署的实例Pod,该应用使用的 所有RDS和Redis等中间件资源,以及该应用挂载的SLB和NAT等。之后会将获取到的关联事件、操作审计、 云监控事件关联到对应的拓扑节点上,单击相应节点,会在左上角的事件详情区域展示与该节点关联的事 件:

- 应用侧相关事件:即事件中心的事件。
- **云资源侧相关事件**:即云监控的相关事件。
- 相关操作审计:即来自于操作审计的审计记录。

该视图可以帮助您在应用出现故障时,快速排查关联的资源各自发生了什么问题。例如在大型企业中,由于 某个员工的误操作,重启了生产环境的RDS,导致线上业务故障,利用该视图,可以快速的发现应用访问的 RDS出现了重启操作。

应用监控·控制台功能

普通视图 拓扑视图 集群视图 订阅规则	
×	
事件详情 当前选择节点arms-console-hz-edas,点击其他节点切换	APELICATION
> 应用侧相关事件: 60	
> 云资源则相关事件:0	
>	(HOST) (HOST)
	APPLINSTANCE APPLINSTANCE APPLINSTANCE

集群视图

集群视图可以按照您指定的检索条件进行搜索,搜索的结果会以四个视图进行展示,分别是:不同集群事件数量占比、不同集群事件数量走势、关键事件一览、关键资源一览和事件详情。



- **不同集群事件数量占比**:展示不同集群事件的数量占比。
- **不同集群事件数量走势**:展示不同集群事件在选定时段内的走势。
- 关键事件一览: 展示指定集群下系统预置的典型事件类型在一段时间内发生的次数。
- 关键资源一览:展示指定集群下关键资源的详情列表。
- **事件详情**:展示指定集群下所有事件的详情列表。

集群视图查看说明:

- 单击不同集群事件数量走势中的立柱,可在事件详情区域查看对应时段内的所有事件列表。
- 单击查看集群相关事件,可在普通视图页签查看所有集群相关事件详情。
- 单击**关键事件一览**中的事件名称,可在**普通视图**页签查看对应事件详情。
- 单击关键资源一览中的查看生命周期相关事件,可在事件详情区域查看对应时段内的资源对应的事件列表。
- 单击事件详情右侧的查看,可以查看事件的具体内容。
- 单击事件详情右侧的订阅,可以订阅指定事件。更多信息,请参见订阅规则。

订阅规则

订阅规则页签展示您当前所有的订阅规则,您可以在该页面启用、禁用或者修改规则。

注意 您只能修改您自己在控制台创建的规则,无法修改系统内部自动创建的规则。

订阅规则是一条表示您订阅满足指定条件的事件,并将该事件发送到指定Webhook的规则。创建订阅规则有两种方式:

- 方式一: 在**订阅规则**页签右上角单击新建订阅规则。
- 方式二:在普通视图页签的事件列表区域单击事件的订阅。
- 1. 单击订阅规则页签, 在页面右上角单击新建订阅规则。
- 2. 在创建订阅规则面板的填写基本信息页面输入规则名称和规则描述,然后单击下一步。
- 3. 在选择事件模式页面设置事件规则参数,然后单击下一步。

参数	描述
事件来源	在下拉列表中选择事件来源。
事件类型	在下拉列表中选择事件类型。
事件等级	在下拉列表中选择事件等级。
事件关键字	在文本框中输入事件关键字。
显示高级过滤选项	默认关闭,开启后可设置 关联应用ID、主机IP、集群 ID和POD名过滤选项。
自定义过滤条件	自定义过滤条件一般用来指定事件体JSON的某个字段 需要满足的条件,根节点是data,以 . 的形式下钻 事件体JSON的某个字段。请输入自定义的过滤条件, 最多可以设置6条过滤条件。
选择有效字段	以data.x.y的形式选择有效字段,选择完成后,请给选 择的字段输入一个别名,该别名可用于在填写 Webhook信息的Post请求体中以占位符的形式出现。 最多可以设置6条有效字段。
消息通知模板	消息通知模板会作为当指定消息发生时通知给您的内容。如果通知对象为钉钉机器人Webhook,请注意在 消息模板中包含创建钉钉机器人Webhook时的关键 字。

4. 在选择联系人页面选择联系人,然后单击提交。

如果选择联系人的列表中没有联系人信息,在右侧单击新建联系人,创建联系人后再在列表中选择。

1. 在普通视图页签的事件详情区域,单击事件详情右侧的订阅。

以该种方式创建的订阅规则,会根据您选择的事件来自动选择事件来源、事件类型和事件等级三个过滤 条件。

 2. 单击事件JSON文件中某个字段 Value的形式来选择自定义过滤条件和有效字段。每次单击会自动生成一 个过滤条件和选择字段,同时您还可以手动修改或者删除字段来调整规则,完成事件模式设置后单击下 一步。 在选择联系人页面选择联系人,然后单击提交。
 如果选择联系人的列表中没有联系人信息,在右侧单击新建联系人,创建联系人后再在列表中选择。

新建订阅规则

在普通视图中订阅事件

3.10. 数据库调用

本文说明如何查看数据库调用,从而了解应用的数据库调用情况,包括概况、SQL调用、异常、调用来源、 调用链列表等信息。

前提条件

接入应用监控

功能入口

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏,选择应用监控 > 应用列表。
- 3. 在顶部菜单栏,选择地域。
- 4. 在应用列表页面,单击应用名称。
- 5. 在左侧导航栏,单击数据库调用。
- 6. 在显示的页面,选择数据库,设置时间段。

۵ <u>×</u> ۵		電近15分钟 曽
响应时间 / 请求数 / 异常数 ☴↓	概览 SQL调用分析 异常分析 调用来源 💶 调用链查询	
新有美型 ン 清暗入 Q	医伤脑明	
wyca, mysgl- 1.1ms / 1548 / 0	■ 2015 0 第2月10 0 和目目で 0 和目目 1 日本 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	
<	mysql- Ø Q Q C	

- 7. 完成设置后,您可以执行以下操作:
 - 单击概览页签, 查看应用的数据库调用概况。
 - 单击SQL调用分析页签,查看应用的SQL调用。
 - 单击**异常分析**页签, 查看应用的数据库调用异常。
 - 单击调用来源页签,查看应用的数据库调用来源。
 - 单击调用链查询页签, 查看应用的调用链列表。

概览

概览页签下显示数据库的调用关系拓扑、请求数、响应时间、错误数、返回大小和性能一览。

既览 SQL调用	分析 异常分析 🕴	周用来源 www 调用链到	登询						
				1					
				11 -					
				68	t: 0				
			(t: 6188				
			mysql-						
			Ø	Q	Q C				
清求数			2141		临床时间 古主曲线家	网络古空度公析			z k a i
2/0					0.0mg	08/00/08/20101			
240				_	0.8005			~	
180					U.6ms				
120					0.4ms	-	\sim	-	
60					0.2ms				
0	05 01 40 05	07.01.00.01	07.01.00.07		Oms	05 01 40 05	05 01 10 01	AF 44 40 AF	
U5-U6 10:23	U5-U6 10:27	• 调用 MYSQL	U5-U6 10:35		05-06 10:23	05-06 10:27	• 適用 MYSQL	05-06 10:35	
错误数 点击曲线的	郑增点深度分析		~ [~ [- *	返回大小				2 kr 6
2					2				
1					1				
0				_	0				

1. (可选) 在概览页签下, 您可以执行以下操作:

○ 单击 🕘 图标或向上滑动鼠标滚轮,放大应用拓扑图。

○ 单击 🕘 图标或向下滑动鼠标滚轮,缩小应用拓扑图。

○ 单击 🥑 图标,将拓扑图恢复至默认大小。

- 单击 🔗 图标,可以将拓扑图调整为适应页面大小。
- 将光标移到统计图上, 查看统计情况。
- 使用光标选中一段时间,查看指定时间段的统计情况。
- 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。
- 单击 图标,查看该指标的API详情。
- 单击 資图标,为该指标创建报警。具体操作,请参见创建报警。

SQL调用分析

SQL调用分析页签下显示数据库的SQL调用数柱状图、响应时间时序曲线、SQL语句列表等信息。

SQL调用统计 / 每分钟					
2 1.5 1 0.5 0 05-06 11:07	05-06 11:12	05-06 11:1	7		05-06 11:22
数据库表: 请选择数据库表 ∨ 操作名: 请选择操作名	遺库表 諸选择数据年表 > 提作名: 请选择操作名 >				
所屬应用 数据库类型	SQL语句	平均耗时小	调用次数 ♪	返回大小	撮作
frontend MySQL SELECT * FROM user_base_info			1558	- 0	调用统计 调用链查询

- 1. (可选)在SQL调用分析页签下,您可以执行以下操作:
 - 将光标移到统计图上, 查看统计情况。
 - 使用光标选中一段时间,查看指定时间段的统计情况。
 - 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。
 - 单击 图标, 查看该指标的API详情。
 - 在SQL语句的操作列,单击调用统计,查看该SQL语句的SQL调用统计。
 - 在SQL语句的操作列,单击调用链查询,查看该SQL语句的调用链列表。

异常分析

异常分析页签下显示数据库的异常情况。



1. (可选) 在异常分析页签下,您可以执行以下操作:

⑦ 说明 如果要过滤异常,在应用设置>自定义设置>高级义设置的异常过滤文本框设置。

- 将光标移到统计图上, 查看统计情况。
- 使用光标选中一段时间,查看指定时间段的统计情况。
- 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。
- 单击 国标, 查看该指标的API详情。
- 在异常的操作列,单击调用统计,查看该异常的统计情况。

• 在异常的操作列,单击详情,查看该异常的详情。

调用来源

调用来源页签下显示数据库的调用来源情况。

全部折疊/展开 响应时间 >> 目 输入应用名称或接口名称	Q 搜索结果数量: 6	
● arms-k8s-demo /demo/queryNotExistDB/11 @查看详情		请求数: 59079 / 响应时间: 3.2ms / 错误数: 59079 🗸 🗸
响应时间	请求数	错误数
4ms 3ms ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~	1600	1600
2ms	800	008
Ims	400	400
Cms 11-0100:00 11-24 00:00 12-17 00:00 01-09 00:00 ● 49(应答)问	0 11-01 00:00 11-24 00:00 12-17 00:00 01-09 00:00 ● 请求政	0 11-0100:00 11-2400:00 12-1700:00 01-0900:00 ● 错误数

- 1. (可选)在调用来源页签下,您可以执行以下操作:
 - 在搜索框, 输入应用或接口名称, 单击 Q图标, 查看应用或接口的情况。
 - 在调用来源应用的接口右侧,单击查看详情,查看该接口的调用链列表。
 - 将光标移到统计图上, 查看统计情况。
 - 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。

调用链查询

调用链查询页签下显示数据库的所有调用链列表。

概览 SQL调用分析 异常分析	调用来源《》 调用链查询					
● 新版调用链查询分析 Trace Explorer 已上线	 立即体验					
输入接口名称搜索	~					
产生时间 ↓	接口名称	所属应用	耗时↓	状态↓	Traceld	操作
2022-05-06 10:43:16	/tb/	Transmission of Con-	142ms	•	eal	查看日志
2022-05-06 10:43:16	/tb/	Torneral Inc.	140ms	•	eal	查看日志
2022-05-06 10:43:16	/tb/	Transmitt .	140ms	•	ea1	查看日志
2022-05-06 10:43:16	/tb/	Number of Concession, Name of Street or Street	139ms	•	eal	查看日志
2022-05-06 10:43:16	/tb/	Number of Concession, Name of Street	139ms	•	eal	查看日志
2022-05-06 10:39:46	/tb/	Number of Concession, Name of Street or Street of Street	135ms	•	eal	查看日志
2022-05-06 10:39:46	/tb/	Transmittened.	135ms	•	eal	查看日志
2022-05-06 10:39:46	/tb/	Providence	135ms	•	ea1	查看日志

- 1. (可选)在调用链查询页签下,您可以执行以下操作:
 - 在搜索框,输入接口名称,然后单击⊂图标,查看指定接口的调用链列表。
 - 在Traceld列,单击Traceld名称,查看调用链路详情。
 - 在操作列,单击查看日志,查看该调用链的日志信息。

3.11. 外部调用

当您需要定位应用外部调用缓慢或出错的问题时,可以使用ARMS应用监控的外部调用功能实现问题定位。

功能入口

1. 登录ARMS控制台。

- 2. 在左侧导航栏选择应用监控 > 应用列表,并在顶部菜单栏选择目标地域。
- 3. 在应用列表页面单击目标应用的名称。
- 4. 在左侧导航栏中单击**外部调用**。

外部调用页面的左侧列表展示了应用的所有外部调用。您可以按照响应时间、请求数、错误数或异常数 对该列表排序。

概览

在左侧列表选中其中一个外部调用,可在概览页签上查看该外部调用的请求数、响应时间、错误数以及 HTTP-状态码的时序曲线。

峭应时间 / 请求数 / 错误数 / 异常数 三	概第 调用来源 MET 接口快照 MET		
调用类型 全部 V 清编入 Q	请求数 / 每分钟	ビ 同 🐩 响应时间 / 每分钟 点击曲线突造点深度分析 (专家版可用)	🗉 厳
arms 12s / 146 / 0 / 0 www. 2295ms / 118 / 0 / 0		23 1.58 18 500ms	/
	0	0mg	
	福洪数 / 每分钟 点击曲线完准点采取分析(令家族可用) 2	ビ 回 第 HTTP-状态研究計 24 18 12 6 0 0 10 10 10 10 10 10 10 10	
	11-23 17:30 11-23 17:30 11-23 17:36 ● 調用 HTTP	• 200 • 20X • 3XX • 4XX • 5XX	

调用来源

在左侧列表选中其中一个外部调用,可在**调用来源**页签上查看该外部调用所有接口的响应时间、请求数和错 误数的时序曲线。

响应时间 / 请求数 / 错误数 / 异常数 目	概范 调用来源 🚾 接口快照 🚾		
週用类型 全部 ∨ 清協入 Q	全部折叠/展开 魄应时间 Y 王 输入应用名称或接口名称	Q 提索结果数量: 2	
arms- 1.2s / 146 / 0 / 0	● arms		请求数: 30 / 响应时间: 233.3ms / 错误数: 0 💙
www.229.5ms / 118 / 0 / 0	响应时间	请求数	错误数
	300ms	2	2
	150ms	1	1
	75ms	0.5	
<	0ms 11-25 17:46 11-25 17:50 11-25 17:54 11-25 17:58 • 印应时间	0 11-25 17:56 11-25 17:50 11-25 17:54 11-25 17:58 ● 请求数	0 11-25 17:46 11-25 17:50 11-25 17:54 11-25 17:58 ● 错误数
	● arms- Q查看详情		请求数: 90 / 响应时间: 227.5ms / 错误数: 0 🗸 🗸
	响应时间	请求数	错误数
	300ms	6	2
	225ms	4.5	
	150ms	3	1
	75ms	1.5	
	0ms 11-25 17:46 11-25 17:50 11-25 17:54 11-25 17:58 ● 網应时间	0 11-25 17:46 11-25 17:50 11-25 17:54 11-25 17:58 ● 请求款	0 11-25 17:46 11-25 17:50 11-25 17:54 11-25 17:58 ● 错误数

在调用来源页签上,可按需执行以下操作:

- 在页签顶部单击**全部折叠/展开**,即可折叠或展开下方的所有接口。
- 在页签顶部的搜索框内输入应用名称或接口(Span)名称的关键字,并单击搜索图标,即可筛选出符合 条件的接口。
- 单击接口信息所在的折叠面板,或者单击行末的上箭头或下箭头,即可展开或折叠该接口的性能指标信息。

接口快照

在左侧列表选中其中一个外部调用,可在**接口快照**页签上查看该外部调用的参数详情,单击Traceld可以查 看调用链路和业务轨迹。

3.12. MQ监控

ARMS应用监控的MQ监控可展示消息队列RocketMQ版的Topic发布和订阅消息的情况。

功能入口

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏选择应用监控 > 应用列表,并在顶部菜单栏选择目标地域。
- 3. 在应用列表页面单击目标应用名称。
- 4. 在左侧导航栏中单击MQ监控。
- 5. 在页面右侧单击查询结果的链接。

<		apmZxTest ►	2018-11-14 10:17 至 2018-11-29 10:32	
应用总览		通过消息队列(MQ)发送消息的情况		
应用详情		1 http://	请求数: 7391 / 响应时间: 36.524928774928778ms / 错误数: 3696 /	^
接口调用 数据库调用		2 http://	请求数: 7391 / 响应时问: 36.76760889712697ms / 错误数: 3695 /	^
MQ监控	NEW	通过消息队列(MQ)接收消息的情况 结果数:2		
监控方法自定义		1 http://	请求数: 14741 / 响应时问: 0.25ms / 错误数: 13161 /	^
应用设置	-	• 2 http://	请求数: 30533 / 响应时间: 0.15ms / 错误数: 29762 /	^
				咨询
				· 建

功能介绍

• 在概览页签中以拓扑图展示应用与MQ数据源之间的消息发布和订阅关系。



• 在发布端统计页签展示消息发布的统计数据,包括请求数、响应时间和错误数。

apmZxTest + http://				2018-11-14 14:10 至 2018-12-03 1-	1:25
概览 发布端统计 订阅端统计 接口快照					返回总览
请求数 10K 8K 6K 4K 2K 0 11-16 00:00 11-19 00:00 11-22 00:00	■会部計詞 35.5ms 36.2ms 36.5ms 36.5ms 36.5ms 35.0ms 35.7ms 45.7	1-16 00.00 11-19 00.00	11-22 00:00	研究数 5K 4K 3K 2K 1K 0 11-16 00:00 11-19 00:00	答 建 议 11-22 00:00
应用 IP	调用次数	平均耗时	错误数	操作	
apmZxTest	69040	37.15	34522	接口快照	

• 在**订阅端统计**页签展示消息订阅的统计数据,包括消息请求数、响应时间和错误数。



在接口快照页签提供关于消息发布和订阅的接口快照,您可以通过Traceld链接查看完整调用链以及诊断问题原因。

apmZxT	est • http://	and in internet.	ni yan ami'n					2018-11-	-14 14:10 至 2018-12-03 14:25	
概览	发布端统计	订阅端统计	接口快照	1					返回总览	Â
共100条记	录。									
产生时间	1 1	安口名称				所属应用	耗时 ≑	状态 🛊	TraceId	咨
2018-11-	14 14:10:31 9	Gend Topic@zxMq1	Гмо			apmZxTest	109 (ms)	٠	0a65592f15421758314034559d02f4	前 · 建
2018-11-	14 14:10:31 9	Gend Topic@zxMq(Dne			apmZxTest	91 (ms)	٠	0a65592f15421758314024558d02f4	议
2018-11-	14 14:10:11 9	Send Topic@zxMq(Dne			apmZxTest	70 (ms)	٠	0a65592f15421758114184536d02f4	
2018-11-	14 14:10:11 5	Send Topic@zxMq1	Гwo			apmZxTest	70 (ms)	٠	0a65592f15421758114174532d02f4	
2018-11-	14 14:10:51 5	Send Topic@zxMq1	Гwo			apmZxTest	62 (ms)	٠	0a65592f15421758514044573d02f4	
2018-11-	14 14:11:11 5	Gend Topic@zxMq1	Гwo			apmZxTest	59 (ms)	•	0a65592f15421758714044591d02f4	-

3.13. 主动剖析

ARMS应用监控的主动剖析功能可以根据您的实际需求,手动触发线程剖析监听任务。

前提条件

接入应用监控

⑦ 说明 该功能需要升级探针版本至2.7.1.1及以上,如何升级探针,请参见升级探针。

新增主动剖析任务

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏,选择应用监控 > 应用列表。
- 3. 在顶部菜单栏,选择地域。
- 4. 在应用列表页面,单击应用名称。
- 5. 在左侧导航栏, 单击主动剖析。
- 6. 在新增任务区域设置所有参数,然后单击完成。

新增任务				
*接口名称	请选择接口名称			
* 监控持续时间	🕽 5min 🔿 10min 🔿 15min			
* 剖析开始阈值 (ms)	请输入剖析开始阈值 (50ms-3	:0s)		
* 线程剖析频率 (ms)	请输入线程剖析频率 (5ms-1s	请输入线程剖析频率 (5ms-1s)		
* 最大采样数	请输入最大采样数 (1~100)			
完成查看历史任务				
参数		描述		
~~~		ли ~_		
接口名称		接口名称。		

参数	描述
监控持续时间	一次监控任务有效的生命周期。选项: o 5min o 10min o 15min
剖析开始阈值(ms)	超过剖析阈值的慢调用才会被监听。例如:一个持续 时间为1.4s的请求,如果剖析开始阈值设置为 100ms,那么只有100ms~1400ms的时间段会被监 听。取值范围:50ms~30000ms。
线程剖析频率(ms)	线程剖析过程中,接口快照的打印频率。取值越小, 精度越高,相应的性能开销越大,建议频率设置为 100ms。取值范围:5ms~1000ms。
最大采样数	最多监听的慢调用数量。取值范围:1~100。

## 查看线程剖析信息

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏,选择应用监控 > 应用列表。
- 3. 在顶部菜单栏,选择地域。
- 4. 在应用列表页面,单击应用名称。
- 5. 在左侧导航栏,单击主动剖析。
- 6. 在新增任务区域单击查看历史任务。
- 7. 在页面右上角选择需要查看的时间段,并在在任务列表区域单击需要查看的主动剖析任务。

在任务列表区域显示所选时间段内所有的主动剖析任务。

	r a							近三天	÷
注意: 该功能需要升级探针版本至 2.7.1.1	及以上								
任务列表	十 新增任务	线程剖析信息						共查询到6	条
实例总数 / 监听完成 / 当	监听执行 / 异常终止 〓	TraceID	产生日志时间♪	接口名称	响应码⑦	IP	耗时止	详情	
请输入	Q	NAME OF TAXABLE PARTY.	2021-02-23 19:30:07	/demo/mysqlOne	03		15s	线程剖析	
/demo/mysqlOne 2021-02-23 19:29:55	4 / 2 / 2 / 0		2021-02-23 19:29:30	/demo/mysqlOne	03	1000	15s	线程制析	
/demo/userDefinedTwo	4 / 2 / 2 / 0		2021-02-23 19:28:20	/demo/mysqlOne	03	1.00	15s	規程創析	
2021-02-23 19:02:45		NUMBER OF STREET, OPPOSITE	2021-02-23 19:27:43	/demo/mysqlOne	03		15s	线程剖析	
2021-02-23 17:54:55	4/2/0/2	NAME OF TAXABLE PARTY.	2021-02-23 19:26:33	/demo/mysqlOne	03	1.00	15s	线程剖析	
< /test/send 2021-02-23 17:49:54	2 / 0 / 2 / 0		2021-02-23 19:25:56	/demo/mysqlOne	03		15s	线程制析	

在右侧线程剖析信息区域显示当前选中接口的剖析信息,包括:TraceID、产生日志时间、接口名称、 响应码、IP和耗时。

- 单击TraceID,可以查看调用链路详情。更多信息,请参见调用链路查询。
- 单击**详情**列的线程剖析,查看目标链路的线程剖析详情。

 $\times$ 

线程剖析	
机器IP: 监听耗时: 4785ms <mark>说明: 只包含一个子方法时不缩进。</mark> 釘 java.lang.Thread.run():745 [4785ms] org.apache.tomcat.util.thread5.TaskThr java.util.concurrent.ThreadPoolExecuto ▼显示已隐藏的59个条目	实际耗时:4825ms 服务名称:/demo/oracleOne 色表示真正消耗CPU的方法。格式:类名.方法名.行号[监听耗时] ead\$WrappingRunnable.run():61 [4785ms] r\$Worker.run():617 [4785ms]
java.lang.reflect.Method.invoke():497 sun.reflect.DelegatingMethodAccessor sun.reflect.GeneratedMethodAccessor com.primeton.arms.demo.service4.Zipi ~ com.primeton.arms.demo.database com.primeton.arms.demo.database java.sql.DriverManager.getConnect java.sql.DriverManager.getConnect oracle.jdb.cdriver.rdacDriver.strensio oracle.jdb.cdriver.rdacDriverstrensio oracle.jdb.cdriver.rdacDriverstrensio oracle.jdb.cdriver.rdacDriverstrensio	4785ms]         ImpLinvoke():-1 [4785ms]         Jinvoke():-1 [4785ms]         vinBraveController.oracleOne():194 [4785ms]         .OracleUtil.oracle1005qlffset():33 [1100ms]         .OracleUtil.oracleWint25ldCondition():57 [1100ms]         .OracleUtil.oracleExecute():71 [1100ms]         on():247 [1100ms]         ext():801 [1100ms]         .oracleUtil.oracleS[1100ms]         .oracleUtil.oracleS[1100ms]         .oracleUtil.oracleS[1100ms]         .oraclit2():438 [1100ms]

# 查看线程剖析任务详情

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏,选择应用监控 > 应用列表。
- 3. 在顶部菜单栏,选择地域。
- 4. 在应用列表页面,单击应用名称。
- 5. 在左侧导航栏,单击主动剖析。
- 6. 在新增任务区域单击查看历史任务。
- 7. 单击任务列表区域目标剖析任务右下角的详情。

在日志页签可以查看该任务的详细信息。

任务详情			×	
任务日志				
实例总数 8	监听完成 4	监听执行 4	异常终止 0	
接口名称	/demo/mysqlOne			
监控时间	2021-02-25 17:13:05			
监控持续时间	300000			
剖析开始阈值 (ms)	1000			
线程剖析频率 (ms)	100			
最大采样数	3			

8. 单击日志页签。

在日志页签可以查看该任务的日志信息。

任务详情	×	
任务日志		
实例总数: 8		
2021-02-25 17:13:05	实例: a2n80plglh@8e0988db8768a49_1614079494147 实例: WATCH_FINISHED	
2021-02-25 17:12:30	实例: a2n80plglh@8e0988db8768a49_1614079494147 实例: WATCH_FINISHED	
2021-02-25 17:08:01	字例: a2n80plglh@8e0988db8768a49_1614079494147 实例: START_WATCH	
2021-02-25 17:08:01	实例: a2n80plglh@8e0988db8768a49_1614079494147 实例: START_WATCH	

# 相关文档

• 使用线程剖析诊断代码层面的问题

# 3.14. 应用诊断

# 3.14.1. 实时诊断

实时诊断功能适用于在短时间内密切监控应用性能和定位问题原因的场景。本文介绍实时诊断功能的使用方法。

# 背景信息

当您需要密切监控一小段时间内的应用性能时,例如发布应用或者对应用进行压测时,可以使用实时诊断功能。开启实时诊断后,本功能会持续监控应用5分钟,并在这5分钟内全量上报调用链数据。接下来,您就能以出现性能问题的调用链路为起点,通过方法栈瀑布图和线程剖析等功能定位问题原因。

### 快速入门

请查看以下交互式文档,了解ARMS应用监控的实时诊断功能。为获得最佳体验,请在新窗口中打开查看。

# 功能入口

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏选择应用监控 > 应用列表,并在顶部菜单栏选择目标地域。
- 3. 在应用列表页面单击目标应用的名称。
- 4. 在左侧导航栏选择应用诊断 > 实时诊断。

### 开启和终止实时诊断

首次进入**实时诊断**页面时,默认自动开启实时诊断。其他情况下,如需开启实时诊断,请单击页面右上角的**开启实时诊断**。

实时诊断将于自动开启5分钟后自动终止。如需提前终止实时诊断,请单击页面右上角的暂停定时刷新。

### 查看实时监控数据

在**实时请求分布**和**请求数/耗时分布**区域,您可以查看截至当前时间点捕捉到的最后1000次请求统计数据。

+				× ノ 捜索		剩余连接时间为76秒。	暂停定时刷新	
武助请求分布     105     6 Enor ● Success					请求数/耗时分布 ■ Error ■ Success			
15 • • • • • • •	•••••	10						
100ms	• • • • •	• • • • •		• • • • •	1			
1ms 12-18 13:50:59	12-18 13:52:25	12	-18 13:53:48	12-18 13:55:13	0 0-0.5s	0.5-1s 1-3	38	
调用链列表 接口聚合								
TraceId	产生时间小	接口名称	IP	耗时小	状态码 ♪	小常只出続	*	
and the second sec	2020-12-18 13:50:59			1014ms	500	是		
	2020-12-18 13:51:03	10.000		72ms	200	否		
	2020-12-18 13:51:03	1000	1000	1010ms	500	是		
1	2020-12-18 13:51:07	(and		2ms	500	是		
Includes and the local second	2020-12-18 13:51:07	100	100.00	8ms	500	足		
	2020-12-18 13:51:11		1100.00	1011ms	500	是		
Inclusion of the local sectors and	2020-12-18 13:51:12			2ms	<b>6</b> 500	是	<b></b>	
Includes and Additional and	2020-12-18 13:51:15	lana internet interne	1000	1042ms	<b>6</b> 500	足	8	

在**实时请求分布**区域的图表中,用鼠标框选一段时间区间,即可将所选时间区间设为数据可视时间区间, 即图表中仅显示该时间区间内的数据。此后,单击图表右上角的重置即可恢复为默认视图。
+				X レ 捜索		剩余连接时间为 274 秒。	暂停定时刷新
实时请求分布 10s		<ul> <li>Error</li> <li>Success</li> </ul>		重置	请求数/耗时分布 100	Error Success	
1s •	• • • •	• •	•••••	• •	10		
10ms	• • •		• • •	•	1		
ims 12-18 13:53:01	12-18 13:53:34	12	-18 13:54:04	12-18 13:54:27	0-0.5s	0.5-1s 1-3s	
调用链列表 按口聚合							
Traceld	产生时间。1	接口名称	IP	耗时 小	状态码 ♪	小 常呆出態	
1000 (0000 0000 000	2020-12-18 13:53:01	100	1.100	8ms	500	是	Â
Index of the second	2020-12-18 13:53:01	-144	1000000	3ms	500	是	
NAME AND TAXABLE	2020-12-18 13:53:01	Jane 1	1-40000	68ms	0 200	否	
table opportunities	2020-12-18 13:53:01	-105	1.000	987ms	500	뷴	
	2020-12-18 13:53:05	-	To an	10ms	500	是	
book opposition and	2020-12-18 13:53:10		1.000	9ms	500	是	
	2020-12-18 13:53:10	-inc	To make and	2ms	<b>5</b> 00	是	
0ba5df6b16082707941625244d16d3	2020-12-18 13:53:14	/oome	11.165.223.107	1099ms	500	是	88

### 筛选监控数据

您可以按照接口名称和IP筛选页面上显示的请求监控数据。

- 1. 在实时请求分布区域上方单击+图标。
- 2. 在下拉框中选择一个API或IP,并单击**搜索**。 仅选中接口的请求监控数据会显示在页面上。

### 查看调用链信息

在调用链列表和接口聚合页签上,您可以查看相应时间区间内捕捉到的全部调用链信息。单击一个 Traceld,即可进入调用链路页面,并借助本地方法栈瀑布图和线程剖析等功能定位问题原因。

+				表鉄 しみ X		ł	刷余连接时间为 97 秒。	暂停定时刷新
编码	请求分布	• Error •	Success		请求数/耗时分布	📕 Error 📕 Suc	cess	
103	••••••••••••••••••••••••••••••••••••••	••• * * • • * <del>*</del> * • • • • • * *		••••	1000		-	
100m: 10m:	· · · · · · · · · · · · · · ·	• • • • • • • • •	• • • • • • • • • • • • • • • • • • •		10			
1ms 12-1	• 13:50:59	12-18 13:54:03	12-18 13:58:21	12-18 14:01:25	0 -0.5s	0.5-1s	1-3s	>5s
调用	链列表 接口聚合							
	接口名称	请求数小	平均耗时业	错误率(%) 🖡		4 (%) 本常程		
+	- Anna	60	1269.7ms	100%		100%		
+	danal in the second	51	2.9ms	100%		100%		
+		46	10.8ms	100%		100%		
+	dan dan	41	14.4ms	100%		100%		
+		38	73.5ms	0%		0%		

### 相关文档

- 调用链路查询
- 使用线程剖析诊断代码层面的问题

# 3.14.2. 启动分析

ARMS应用监控支持对部署在ACK集群下的Dubbo应用进行启动分析,统计应用下各Dubbo实例启动耗时, 单位为毫秒(ms),并支持从容器启动、应用初始化、服务注册和正常流量进入四个维度进行数据展示分 析。

## 功能入口

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏选择应用监控 > 应用列表,并在顶部菜单栏选择目标地域。
- 3. 在应用列表页面单击目标应用的名称。
- 4. 在左侧导航栏选择**应用诊断 > 启动分析**。

在启动分析页面左侧列表显示了当前应用下的所有Dubbo实例。

# 容器启动

容器启动页签显示了Dubbo实例启动过程中容器启动的过程和耗时。

容器启动(22s)	应用初始化(5s) 服务注册(2s) 正常流量进入(0s)	
类型	名称	耗时(ms)
容器启动	Created container one-pilot-initcontainer	Ims
容器启动	Started container one-pilot-initcontainer	1ms
容器启动	Successfully pulled image "registry-vpc.cn-hangzhou.aliyuncs.com/arm	1ms
容器启动	Created container gracefulshutdown	Ims
容器启动	Successfully pulled image "registry-vpc.cn-hangzhou.aliyuncs.com/hz	1ms
容器启动	Started container gracefulshutdown	Ims
容器启动	Pulling image "registry-vpc.cn-hangzhou.aliyuncs.com/hz-demo-test/du	Ims
容器启动	Pulling image "registry-vpc.cn-hangzhou.aliyuncs.com/arms-docker-re	Ims
容器启动	Started container dubbo-test	1ms
容器启动	Container image "registry-vpc.cn-hangzhou.aliyuncs.com/edas_unified/	Ims
容器启动	Created container dubbo-test	Ims

# 应用初始化

应用初始化页签显示了Dubbo实例启动过程中各服务初始化耗时。

容器启动(22s)	应用初始化(5s) 服务注册(2s) 正常流量进入(0s)				
类型	名称	耗时(ms)			
应用初始化	org.spi	87ms			
应用初始化	org.spi	4ms			
应用初始化	org.spi	15ms			
应用初始化	org.api	133ms			
应用初始化	service	24ms			
应用初始化	dubbo-	5ms			
应用初始化	org.api	Oms			
应用初始化	dubbo!	Oms			
应用初始化	dubbo(	Oms			
应用初始化	dubbol	2ms			
应用初始化	proper	24ms			
应用初始化	org.spi	Oms			
应用初始化	referer	55ms			

# 注册服务

**注册服务**页签显示了Dubbo实例启动过程中各服务注册耗时。

容器启动(21s)	应用初始化(5s) 服务注册(2s) 正常流量进入(0s)	
类型	名称	耗时(ms)
服务注册	export service com.alibaba	948ms
服务注册	export service com.alibaba	249ms
服务注册	export service org.apache.	67ms

# 正常流量进入

**正常流量进入**页签显示了Dubbo实例启动过程中正常流量进入的时间段和耗时。

容器启动(21s)	应用初始化(5s) 服务注册(2s)	正常流量进入(0s)	
类型	名称		耗时(ms)
正常流量进入	Dubbo Apllication Started	开始时间: 03-24 17:41:34.727 结束时间: 03-24 17:41:34.727	Oms

# 3.14.3. 异常分析

本文说明如何查看异常分析,从而了解应用的异常情况。

## 前提条件

接入应用监控

### 功能入口

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏,选择应用监控 > 应用列表。
- 3. 在顶部菜单栏,选择地域。
- 4. 在应用列表页面,单击应用名称。
- 5. 在左侧导航栏,选择应用诊断 > 异常分析。
- 6. 在右侧页面,设置时间范围,然后单击要查看的异常。

spring-data-redis[EDAS]								最近15分钟 1	<b></b>
异常选择	异常名称 ⊒↓	异常次数 异常名称:NestedServietExcept	tion						
请输入	Q	120							_
全部		90							
RedisConnectionFailureException	1920	60							
NestedServletException 2	1920	0							
		03-29 17:08 0	3-29 17:11 03-2	● 今日 ● 昨日	17		03-29 17:20		33-29 17:23
		异常名称	异常接口	摘要	Count 🕸	占比	图表		
<		NestedServletException	/api/cache	org.springframework.web.util.NestedServl etXxxeption at org.springframework.web.s ervlet.FrameworkServlet.processRequestFir ameworkServlet.java.1014) at org.springfra meworkServlet.Servlet.frameworkServlet.do Get(FrameworkServlet.iava.898) Caused b	1920	100.00%		$\bigvee$	_

## 异常次数

异常次数区域显示该应用在指定时间范围抛出该异常的次数。

异常次数 异常名称:NullPointe	erException				1
1/00					
1200					
800					
400					
0					
03-24 17:33	03-24 17:36	03-24 17:39 · 숙日 · 태	03-24 17:42 作日	03-24 17:45	03-24 17:48

- 1. 在异常次数区域下,您可以执行以下操作:
  - 将光标移到统计图上, 查看统计情况。

• 使用光标选中一段时间,查看指定时间段的统计情况。

# 异常接口列表

异常接口列表显示抛出该异常的接口的情况,包括异常接口名称、异常信息摘要、异常次数、占比等。

异常名称	异常接口	摘要	Count 사	占比	图表
NullPointerException	/registerPromCluster	java.lang.NullPointerException	17.4K	99.93%	<u> </u>
NullPointerException	/updateCLusterRolePolicy	java.lang.NullPointerException at com.alibaba. arms.core.service.k8s.impl.K8SClientProviderl mpl.isClientFromVpc(K8SClientProviderImpl.ja va:334)	7	0.04%	
NullPointerException	/createApp	java.lang.NullPointerException at com.alibaba. arms.core.service.trace.common.TraceDiamon dService.publishToDiamond(TraceDiamondSer vice.java:210)	4	0.02%	•
NullPointerException	/openVPromCluster	java.lang.NullPointerException at com.alibaba. arms.core.service.prometheus.PrometheusCo nfigService.listPrometheusRules(PrometheusC onfigService.java?91)	1	0.01%	•

### 1. 在异常接口列表,您可以执行以下操作:

### • 在**异常名称**列,单击异常名称,查看该接口的异常详情。



■ 异常详情面板的异常堆栈页签下显示该接口的异常堆栈信息。

异常名称:	RuntimeException				
异常消息:		1600			
已带拉口.		1200			
开币按口:	**	800			
摘要:	java.lang.RuntimeException at com.alibaba.arms.core.service.pr ometheus.grafana.GrafanaConfigService.refreshViewByOrgld(G	400			
	rafanaConfigService.java:286)	0	03-24 18:48	03-24 18:53	03-
异常次数:	18424			n+ 🗆	
异常堆栈	关联Trace		• 今日 •	IFD.	
ava.lang.Ru at c at c 02) at c	ntimeException om. alibaba. arms. core. service. prometheus. grafana. GrafanaCor om. alibaba. arms. core. service. prometheus. grafana. GrafanaCor om. alibaba. arms. core. service. prometheus. PrometheusServiceJ	ifigService, refre ifigService, insta ifigService, creat impl. addExporter(	shViewByOrgId(Grafan 11BaseK8sEnv(Grafane eGrafanaExporterDash PrometheusServiceImp	aConfigService.java:2 ConfigService.java:16 board(GrafanaConfigSe 1.java:1050)	86) 9) rvice.jæ
ava.lang.Ru at c at c 02) at c at c at c	ntimeException om. alibaba. arms. core. service. prometheus. grafana. GrafanaCor om. alibaba. arms. core. service. prometheus. grafana. GrafanaCor om. alibaba. arms. core. service. prometheus. PrometheusServicel om. alibaba. arms. core. service. prometheus. PrometheusServicel om. alibaba. arms. core. service. prometheus. PrometheusServicel om. alibaba. arms. core. service. prometheus. PrometheusServicel	figService.refre figService.insta figService.creat mpl.addExporter( mpl.initKBSAndNo mpl.installPilot	shViewByOrgId(Grafan eGrafanaExporterDash PrometheusServiceImp deInstance(Prometheu ForCluster(Prometheu	aConfigService.java:2 ConfigService.java:16 board(GrafanaConfigSe 1.java:1050) LServiceImpl.java:679 LServiceImpl.java:531	86) 9) rvice. jav )
ava.lang.Ru at c at c 02) at c at c at c	ntimeException om. alibaba. arms. core. service. prometheus. grafana. GrafanaCor om. alibaba. arms. core. service. prometheus. grafana. GrafanaCor om. alibaba. arms. core. service. prometheus. PrometheusServiceJ om. alibaba. arms. core. service. prometheus. PrometheusServiceJ om. alibaba. arms. core. service. prometheus. PrometheusServiceJ un. reflect. NativeMethodAccessorImpl. invokeO(Native Method)	ifigService.refre ifigService.insta ifigService.oreat impl.addExporter( impl.initK8SAndNo impl.installPilot	shViewByOrgId(Grafan 11BaseK8sEnv(Grafane eGrafanaExporterDash PrometheusServiceImp deInstance(Prometheu ForCluster(Prometheu	aConfigService.java:2 ConfigService.java:16 board(GrafanaConfigSe 1.java:1050) 1.ServiceImpl.java:679 1.ServiceImpl.java:531	86) 9) rvice.jav )
ava.lang.Ru at o at c 02) at c at c at c at s at s	ntimeException om. alibaba. arms. core. service. prometheus. grafana. GrafanaCor om. alibaba. arms. core. service. prometheus. grafana. GrafanaCor om. alibaba. arms. core. service. prometheus. PrometheusServicel om. alibaba. arms. core. service. prometheus. PrometheusServicel om. alibaba. arms. core. service. prometheus. PrometheusServicel un. reflect. NativeMethodAccessorImpl. invokeO(Native MethodAcc	figService.refre figService.insta figService.creat mpl.addExporter( mpl.initKBSAndNo mpl.installPilot essorImpl.java:6	shViewByOrgId(Grafan eGrafanaExporterDash PrometheusServiceImp deInstance(Prometheu ForCluster(Prometheu 2)	aConfigService.java:2 ConfigService.java:16 doard(GrafanaConfigSe 1.java:1050) uSServiceImpl.java:679 uSServiceImpl.java:531	86) 9) rvice.jav ) )
ava.lang.Ru at c at c 02) at c at c at s at s	ntimeException om. alibaba. arms. core. service. prometheus. grafana. GrafanaCor om. alibaba. arms. core. service. prometheus. grafana. GrafanaCor om. alibaba. arms. core. service. prometheus. PrometheusServiceJ om. alibaba. arms. core. service. prometheus. PrometheusServiceJ om. alibaba. arms. core. service. prometheus. PrometheusServiceJ un. reflect. NativeMethodAccessorImpl. invoke0(Native Method) un. reflect. NativeMethodAccessorImpl. invoke(NativeMethodAcc un. reflect. DelegatingMethodAccessorImpl. invoke(NativeMethodAcc	ifigService.refre ifigService.insta ifigService.oreat impl.addExporter( impl.initK8SAndWo impl.installPilot essorImpl.java:6 lethodAccessorImp	shViewByOrgId(Grafan 11BaseK8sEnv(Grafana eGrafanaExporterDash PrometheusServiceImp deInstance(Prometheu ForCluster(Prometheu 2) 1. java:43)	aConfigService.java:2 ConfigService.java:16 board(GrafanaConfigSe 1.java:1050) 1:ServiceImpl.java:679	86) 9) rvice.jæ ) )

异常名称:	ClientException			
已過湖白。			4000 —	
异常消息:			3000 🧮	
异常接口:			2000 —	
摘要:	com.aliyuncs.exceptions.Cli	ientException at com.aliyuncs.Default	t 1000 —	$\sim$
	AcsClient.parseAcsRespons	e(DefaultAcsClient.java:235)	0	
异常次数:	53511			9:10 03-24 19:15 03-24 19:20 03-24
				● 今日 ● 昨日
异常堆栈	关联Trace			
产生时间↓		接口名称	耗时↓	Traceld
2021-03-24	19:11:00	/api/version.json	75ms	1cc9005116165842599342062991bc
2021-03-24	19:11:01	/api/version.json	132ms	2a25dd1816165842612995423991bc
2021-03-24	19:11:01	/api/version.json	131ms	27e68f2516165842616404230991bc
2021-03-24	19:11:01	/api/version.json	91ms	7aa2194b161658426175018832bf5a
2021-03-24	19:11:02	/api/xtrace.json	36ms	7aa2194b161658426210518912bf5a
2021-03-24	19:11:02	/api/version.json	89ms	bfd887a416165842621954412991bc

您可以在**Traceld**列单击Trace ID, 查看该异常调用的链路。 如何分析调用链路,请参见分析调用链路。

• 在**异常接口**列,单击接口名称,查看该接口的调用情况。

接口调用说明,请参见接口调用。

- 将光标移到统计图上, 查看统计情况。
- 使用光标选中一段时间,查看指定时间段的统计情况。

# 3.14.4. 线程分析

线程分析功能提供线程粒度的CPU耗时和每类线程数量的统计,并且每5分钟记录一次线程的方法栈并聚合, 可真实还原代码执行过程,帮助您快速定位线程问题。当发现集群的CPU使用率过高,或者出现大量慢方法 时,可以通过线程分析功能找到消耗CPU最多的线程或方法。

### 功能入口

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏选择应用监控 > 应用列表,并在顶部菜单栏选择目标地域。
- 3. 在应用列表页面单击目标应用的名称。
- 4. 在左侧导航栏中选择应用诊断 > 线程分析。

### 进行线程分析

**线程分析**页面的左侧列表展示了应用的全部线程,您可以根据CPU耗时统计快速发现异常线程。选中某一异 常线程后,再根据右侧的CPU耗时和线程数曲线图分析CPU耗时与线程数变化,例如分析每分钟的线程总数 是否过多。



您还可以单击异常线程的**方法栈**,查看指定时间内的真实运行方法栈,例如查看处于BLOCKED状态的线程对 应的方法,从而优化指定代码段,以便降低CPU使用率。

方法栈	Scheduler_*_Worker-* X
ALL: 365194	UNNABLE: 986 BLOCKED: 87 WAITING: 1498 TIMED_WAITING: 362617
	1/87(次) - 占比9310%
at java.lang.Ob	st wait (Native Method)
ar jaranangio i	.awaitAvailable (BasicResourcePool.java:1315)
	)l.prelimCheckoutResource (BasicResourcePool.java:557)
41.000	ol.checkoutResource (BasicResourcePool.java:477)
41.000	tionPool.checkoutPooledConnection (C3P0PooledConnectionPool.java:525)
at 1000 m	dDataSource.getConnection (AbstractPoolBackedDataSource.java:128)
	tConnection (PoolingConnectionProvider.java:247)
	onnection (DBConnectionManager.java:108)
	irt.detConnection (JobStoreSupport.iava:775)
	.EnhanceJobStoreSupport.getNonManagedTXConnection (EnhanceJobStoreSup
port.ja	17)
	Support.executeInNonManagedTXLock (JobStoreSupport.java:3777)
	Support.retryExecuteInNonManagedTXLock (JobStoreSupport.java:3742)
	reSupport.triggeredJobComplete (JobStoreSupport.java:3039)
	itifyJobStoreJobComplete (QuartzScheduler.java:1804)
	lobRunShell.java:269)
	sl\$WorkerThread.run (SimpleThreadPool.java:573)
	/87(次) - 占比2 30%
[DECONCED]	checkinStatement (CooCooStatementCache iava:220)
a con acta	sheckinStatement (NewPooledConnection java:220)
	ament close (NewProvyDrenardStatement iava:1807)
	sestatement (Stri IDR/Teplarate java-3287)
	Sected Transmic (Stell DRSChelanate Java 2016)
	ggered.jobComplete (JobStoreSupport.java;3116)
	45 executeVoid (JobStoreSupport java:3043)
	VoidTransactionCallback.execute (JobStoreSupport.iava:3703)
	\$VoidTransactionCallback.execute (JobStoreSupport.java:3701)
	t.executeInNonManagedTXLock (JobStoreSupport.java:3787)
	vrt.retryExecuteInNonManagedTXLock (JobStoreSupport.java:3742)
	ort.triggeredJobComplete (JobStoreSupport.java:3039)
	StoreJobComplete (QuartzScheduler.java:1804)
	Shell.java:269)
	kerThread.run (SimpleThreadPool.java:573)
[BLOCKED]	87(次) - 占比1.15%

⑦ 说明 如果单击方法栈后,显示无数据,则需要单击左侧导航栏的应用设置,在自定义配置页 签线程设置区域查看线程诊断方法栈开关是否开启。如未开启,则无法记录方法栈信息;如已开启,则每5分钟采集一次方法栈信息。

# 相关文档

• 使用线程剖析诊断代码层面的问题

# 3.14.5. 日志分析(直接采集)

当应用出现业务异常问题时,您可以通过分析业务日志,精准定位业务异常。日志分析功能支持分析日志服务SLS或直接采集的日志,本文介绍如何开通日志分析功能并分析直接采集的日志。

### 前提条件

ARMS Agent版本为v2.7.1.4或以上。登录ARMS控制台,在应用监控 > Agent列表页面的Java版本发布说 明页签获取2.7.1.4版本的Agent安装包。

### 日志源说明

日志分析功能支持分析的日志源包括:日志服务SLS的日志、直接采集的日志。本文仅介绍如何查询并分析 直接采集的日志,日志服务SLS的日志分析详情,请参见日志分析(日志服务SLS)。

● 日志分析(直接采集):通过ARMS探针采集日志框架的输出并直接推送到ARMS的日志分析中心,通过一 键开启后无需其他操作即可在ARMS控制台查询分析应用日志。

⑦ 说明 日志分析(直接采集)功能需要将探针升级到v2.7.1.4及以上版本。

日志分析(日志服务SLS):
 您需要将应用的日志采集到日志服务SLS,并在ARMS应用配置中配置相应的Project和Logstore,ARMS会内嵌日志服务的页面方便您进行日志分析。

## 功能开通说明 开通日志采集功能

- 1. 在ARMS控制台左侧导航栏,选择应用监控 > 应用列表。
- 2. 在顶部菜单栏,选择地域,然后单击应用名称。
- 3. 在左侧导航栏中单击应用设置,并在右侧单击自定义配置页签。
- 在自定义配置页签的应用日志关联配置区域,选择日志源为默认日志,打开日志框架自动采集开关, 选择需要采集的日志级别,并设置单条日志显示的最大长度和日志上报限制。

⑦ 说明 日志分析功能不会采集日志级别低于应用中配置的日志输出级别的日志。例如应用中配置的日志输出级别为Warn,即使在ARMS控制台配置采集了Debug级别的日志,也不会生效。如果想不重启应用修改应用日志输出级别,您可以使用ARMS的Arthas诊断中的Arthas Shell功能动态修改日志框架的日志输出级别,更多信息,请参见Arthas Shell。

应用日志关联配置 ?				
选择日志源 日志框架自动采集		) 日志服务 控制是否采集日	志框架打印日志,目前支持log	4j2.logback框架, Agent 2.7.1.4以上版本生效
采集日志级别	🗹 Debug	Info	Vam Vam	Error
单条日志最大长度	512	单条	日志最大长度范围 0 ~ 2000 字	<del>7</del>
每秒上报日志最大条数	5000	上限	5000, 超过限制,按照ERROR>\	WARN>INFO>DEBUG优先级保存日志
每秒上报日志最字符数(单	20	上限	制50,最大占用流量为50*10000	000°4(B)≈200(MB), 其中4为utf-8骗码单字符最大字节数
位百万)				

5. 在自定义配置页签左下角单击保存。

### 日志采集原理

- 1. 通过ARMS探针拦截日志框架的日志打印方法,获取日志内容。
- 2. 为日志增加部分上下文(例如traceld、spanName、threadName)。
- 3. 将日志推送至ARMS日志分析中心存储。
- 4. 在ARMS控制台的日志分析页面,查询该条日志。

⑦ 说明 整个日志采集链路时延大约为10s。

### 功能额外开销

• CPU额外开销: 0.01核

- 内存额外开销: 20 M以内
- 带宽额外开销: 最大为每秒配置的日志上报条数上限×配置最长日志字符数(个)

## 单条日志结构

一条日志包含日志附带的Label以及日志自身解析出的字段。

### 日志附带的Label字段

字段	说明
majorVersion	探针主版本号。
minorVersion	探针子版本号。
serverlp	探针所在主机IP。
userld	应用所属用户ID。
appld	应用ID。
job	日志来源,固定为arms-agent。
logType	日志类型,固定为userLog。

### 日志解析出的字段

字段	说明
level	日志级别。
log	包括格式化时间和日志原文。时间和日志中间使用短划线 (-)连接。
loggerName	Logger名称。
parentAppId	本条日志关联的调用链中上游应用ID。
spanName	本条日志关联的SpanName。
parentSpanName	本条日志关联的上游SpanName。
threadName	线程名称。
traceld	关联的TraceID。
ts	日志上报时间。

# 常用查询语句示例

下面以appld为a2n80plglh@89f2dd21b561bdc的应用为例说明。

### 关键字查询

例如搜索包含关键字error的日志,查询语句为:

{job="arms-agent", appId="a2n80plglh@89f2dd21b561bdc"} |= "error"

例如搜索traceld为eaac105afb16540713955671006d0009的日志,查询语句为:

```
{job="arms-agent", appId="a2n80plglh@89f2dd2lb561bdc"} |= "eaac105afb16540713955671006d0009"
```

# 多条件查询

例如搜索同时包含关键字error和Exception的日志,查询语句为:

{job="arms-agent", appId="a2n80plglh089f2dd21b561bdc"} |= "error" |= "Exception"

### 分析日志

#### 例如想要分析不同spanName error日志输出的数量变化趋势,查询语句为:

```
sum(count_over_time({job="arms-agent", appId="a2n80plglh@89f2dd21b561bdc"} | json [1m])) by
(spanName)
```

### 查询结果如下:



同理可以查询不同loggerName、threadName、level的日志输出数量变化。

# 3.14.6. 日志分析(日志服务SLS)

当应用出现业务异常问题时,您可以分析业务日志,精准定位业务异常。日志分析功能支持分析日志服务 SLS或直接采集的日志,本文介绍如何开通日志分析功能并分析日志服务SLS的日志。

### 前提条件

- 已接入应用监控。具体操作,请参见应用监控接入概述。
- 已开通日志服务SLS。登录日志服务控制台时,根据页面提示开通日志服务。
- 已创建Project。具体操作,请参见创建Project。
- 已创建Logstore,具体操作,请参见创建Logstore。

### 日志源说明

日志分析功能支持分析的日志源包括:日志服务SLS的日志、直接采集的日志。本文仅介绍如何查询并分析 日志服务SLS的日志,直接采集的日志分析详情,请参见日志分析(直接采集)。

 日志分析(直接采集):通过ARMS探针采集日志框架的输出并直接推送到ARMS的日志分析中心,通过一 键开启后无需其他操作即可在ARMS控制台查询分析应用日志。

⑦ 说明 日志分析(直接采集)功能需要将探针升级到v2.7.1.4及以上版本。

日志分析(日志服务SLS):
 您需要将应用的日志采集到日志服务SLS,并在ARMS应用配置中配置相应的Project和Logstore,ARMS会内嵌日志服务的页面方便您进行日志分析。

### 步骤一:关联业务日志

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏中选择应用监控 > 应用列表,在顶部菜单栏选择目标地域,然后在应用列表页面单击目标应用的名称。
- 3. 在左侧导航栏中单击应用设置,并在右侧单击自定义配置页签。
- 4. 在自定义配置页签的应用日志关联配置区域,选择日志源为日志服务,打开关联业务日志与 Traceld开关,选择日志服务所在地域,然后绑定Project和Logstore。

应用日志关联配置 😮				
选择日志调	🔾 默以日志 💿 日志服	8 8		
关联业务日志与Tracelo		开启后业务日志中会自动生成调用链的Traceld,此设置在重启应用后生效。	支持し	og4j/Log4j2/Logback日志追件。业务应用需要在日志的Layout中通过声明%X[EagleEye-TracelD]未输出Traceld信息。
日志服务绑定⑦	已授权,请绑定对应的Proje	ect及Logstore,并关联素引,支持指定字段素引及全文素引 区别,《 希末配置,	清前去	日志服务控制台进行配置。
	* 区域:		~	
	* Project名称:	a second second second	~	创建Project <i>分</i>
	* Logstore名称:		~	创建Logstore
	关联索引:	全文素引	~	创建关环录引。

5. 在自定义配置页签左下角单击保存。

### 步骤二:查询并分析日志

- 1. 登录ARMS控制台。
- 在左侧导航栏中选择应用监控 > 应用列表,在顶部菜单栏选择目标地域,然后在应用列表页面单击目标应用的名称。
- 3. 在左侧导航栏,选择应用诊断 > 日志分析。
- 4. 在右侧页面, 执行以下操作:
  - i. 在搜索框中输入查询分析语句。

查询分析语句由查询语句和分析语句构成,格式为查询语句|分析语句,查询分析语句语法请参见<mark>查</mark> 询语法、SQL分析语法。

ii. 设置查询分析的时间范围。

您可以设置相对时间、整点时间和自定义时间。

⑦ 说明 查询结果有1分钟以内的误差。

iii. 单击查询/分析, 查看查询分析结果。

101-000-0	~ a						
已绑定Project:	Logstore:	·····································					
@ arms-console-all-log		③ 2020-09-09 14406:00-2020-09-09 14:15:00 ▼	自动刷新	分享	查询分析属性	另存为快速查询	另存为告答
✓ 1 0bc0594d1599631784						ି ମ	查询/分析
	3 😡 09-09 14:10:07		mpl - promethe	ius check i	nput param success	from	
C .	4 Q 09-0914:10.07	Long - Indume	gisterPromClus II, pilotid: cfaef7	sterServiet 73c98e3e4	RegisterPromClust a79b4aa0402a8	terServlet Falled. → pro	xxyUserid:

# 3.14.7. 日志分析(旧版)

当应用出现业务异常问题时,您可以通过分析业务日志,精准定位业务异常。本文介绍如何开通并使用 ARMS日志分析功能。

### 前提条件

- ARMS Agent版本为v2.7.1.3或以上。登录ARMS控制台,在应用监控 > Agent列表页面的Java版本发布 说明页签获取2.7.1.3版本的Agent安装包。
- 日志分析功能目前仅支持对阿里云容器服务Kubernetes版(简称ACK)集群下应用的日志进行分析,且 Kubernetes集群应用需要通过Annotations方式接入ARMS应用监控。通过Annotations方式接入ARMS应 用监控的具体操作,请参见为容器服务Kubernetes版Java应用安装探针。

### 版本说明

当前版本的日志分析功能已不支持开通,如果已开通当前版本的日志服务,则仍可以使用。

您可以根据需求重新开通新版日志分析功能。

↓ 注意 开通新版日志分析功能后无法返回旧版的日志分析功能。

新版日志分析功能支持分析日志服务SLS或直接采集的日志:

 日志分析(直接采集):通过ARMS探针采集日志框架的输出并直接推送到ARMS的日志分析中心,通过一 键开启后无需其他操作即可在ARMS控制台查询分析应用日志。更多信息,请参见日志分析(直接采 集)。

⑦ 说明 日志分析(直接采集)功能需要将探针升级到v2.7.1.4及以上版本。

日志分析(日志服务SLS):
 您需要将应用的日志采集到日志服务SLS,并在ARMS应用配置中配置相应的Project和Logstore,ARMS会内嵌日志服务的页面方便您进行日志分析。更多信息,请参见日志分析(日志服务SLS)。

### 日志采集原理



- 1. 应用开通日志采集功能后,采集到的业务日志将会保存至本地。
- 2. 安装日志采集探针后,日志采集探针将读取本地业务日志并发送至ARMS控制台。

⑦ 说明 同一个ACK集群下仅需安装一次日志采集探针。

3. 在ARMS控制台的日志分析页面,您可以通过关键字对业务日志进行查询、筛选、分析等操作。例如通 过搜索订单ID、Trace ID、应用名称等参数,可以查看相关的异常日志信息。

查询	日	志
----	---	---

						1		番印度に	现设查定
2 3						🕑 Lat	st 6 hours	~ 0	2
mms.app producten/ice - reach	mm.spp productsmice - week								
100 K 10 K 1 K40 11450 1450 1450 1450 1450 1450 1450 1	19:40 19:50	20.00 20	0:10 20:20	20:30 20:	0.40 20:5	10 21:0	0 21:10	21:20	21:30
Logs Panel									
Department         Department           Production         Clip         Clip </td									
see"_f4csciption''riddax'_vate productarvice_ffdddea_phrg ('leg''2011-11-22 06:54:40 D40 monartarvice_ffdddea_phrg ('leg''2011-11-22 06:55:41 D40	1.22789.54.48.5 0.54.48.322484 0.55.08.475440	HELE")							

基础查询支持以下筛选操作:

- 在基础查询页面右上角(图示中①),设置日志需要查看的时间段。
- 单击arms_app右侧下拉框(图示中②)可以切换已开通日志采集的其他应用。
- 在search右侧文本框(图示中③)输入关键字,按回车或单击search可以搜索包含对应关键词的日志。 例如输入*WARN*,可以查看所有Warn级别的日志信息。

? 说明 关键字区分大小写。

单击页面右上角的高级查询,在高级查询页面,您可以通过LogQL语句查询日志信息。

hervice(脊髓经验K05) 🗸 🗘 這篇目錄	1
2	□ spilt 4 C 2021-11-23 17:51:33 to 2021-11-23 19:45:45 ∨ > 2 Clear All C Run Query ∨ D Live
Loplateds v {arms_app="productservice"}	Lite text and 0.25 $\Phi$ -
+ Add query 🖄 Query history 🔘 Query inspector	
Logs	
20	
Ś. I	
0	
in in a state that the first state of the large la	- Listen Listen Listen Listen Listen Listen Listen Listen Listen
17:55 18:00 18:05 18:10 18:15 18:20 18:25 18:30	18.35 18.40 18.45 18.50 18.55 19.00 19.05 19.10 19.15 19.20 19.25 19.30 19.35 19.40 19.
- info - warning - error	
Time 🚺 Unique labels 🕥 Wrap lines 🂽 Dedup Nore Exact Numbers :	Fip results order
xmmonlabels: productservice c5dcbcd4f98794cf6b3f1d29f0fb8. 1884900439941128 productservice 1637315	1002 anns-spm-demo/productservice productservice anns-apm-demo ff6d56488 Limit 1000 (1000 returned) Total bytes processed: 1 MB
2021.11.23 10:43:27 /"los":"3021.11.23 11:43:27 THED row taxwar bikari wikariDataGource "trace id	*** _ UlkariDool_1 _ Shubdown completed \n" "stream" "stream" "stream" "S001_11.22711-42-22 £200230147"\
services reaction of the services reaction contraction reaction of the reaction of	Hand about a - and down comparison of a strong - and -
teres to be to be an an an and the second to be an an an and the second terms of ter	2.3 stratight is making building of higher Physics Physics and Physics 1 (1991) 41 (1991) 111
2021-11-23 10:43:27	*/* - HERETHERE 4 - Institute Institutes(n*,*htman*/*HERE*/*INE-10-2012)-HERETERETY)
2022-11-23 19:43:27 2021-11-23 19:43:27	*** - Silarifadi 4 - Badane Initiation(v?.vitewi*.*datoi*,*100**.*000 a - 0.112-0.12.00000000)*
2022-13-22 13:441:27 2023-13-23 13:441:27 2023-13-23 13:441:27	Pri - Karali Mal, L Buttano, Haritano, J. W. Wanani, "Malari, "Marri, "Ball, Sci 2013, ed. (3), Allewand D. T. Ballano, and Malari, "Anna and Malari, "Marri, "Malari, Sci 2013, ed. (3), Allewand D. T. Malari, "Malari, Malari, "Malari, "Malari, Malari, "Malari, "Mal Malari, "Malari, "M
2021-11-22 18:041:27 2021-11-23 18:041:27 2021-11-23 18:041:27 2021-11-23 18:041:27	61 Generativa L Belano Jertano, JY., Navari's Marri, "Amori "and a Laborativa"). 62 Generativa L Belano Jentano, "Anno "Amori
1001-11-11 10-017 1001-11-11 10-017 1001-11-11 10-017 1001-11-11 10-017 1001-11-11 10-017	Pri - Securita L. Holina Instance (Pri Neuron Change) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (2007) (
380-14-3 13-047 380-14-3 13-047 380-14-3 13-047 380-14-3 13-047 380-14-3 13-047 380-14-3 13-047	61 Generative L Bellete Jordense - Generative Constructive Co
202 - 12 - 12 - 04 - 07 202 - 12 - 12 - 04 - 07 202 - 04 - 04 - 07 202 - 04 - 04 - 07 202 - 04 - 04 - 04 202 - 04 - 04 - 04 202 -	Pri - Securita L Unitare Estimate
380-371 98-077 380-371 98-077 380-371 98-077 380-371 98-077 380-371 98-077 380-371 98-077 380-371 98-077 380-371 98-077	61 Generation 1 Bestere Jordsmann, - Lee, "Annex" ("Bester", "Bester", "Bester", "Bester Bester Bestere Beste
364 - 16 2 1 6 - 17 364 - 18 - 18 - 18 364 - 18 - 18 364 - 18 - 18 364 - 18 36	
	Cri - Generalis L, Holsen Jostani, J., Y., Sowith, "Mark," (2007) "Control (2007) Control (2008) Section 12 (2008) Control
1 201-1-0 1 90-07 3 201-1-0 190-07 3 201-1-	
1 201-1-1 2 1 2012 2021-1-2 1 2012 202	*** "Compared a 1 makes (and a 2 makes (and a 2 makes") "make" "make" "make" "makes") "makes (and a 2 makes)" (and a 2 makes) "makes" (and a 2 makes) (and a 2 makes) "makes" (and a 2 makes) (and a 2
1 40 45 1 10 107 1 40 45 1 1007 1	
181 1-12 1 19 -07 181 1-12 19	

高级查询支持以下筛选操作:

- 在高级查询页面右上角(图示中①),设置日志需要查看的时间段。
- 在查询文本框(图示中②)输入LogQL语句,单击右上角的Run Query筛选业务日志信息。
   LogQL语句支持通过日志标签和关键字查询日志信息,LogQL语句中必须包含日志标签。
  - 日志标签格式: {label键值对}。您可以单击Log labels,在下拉框中选择日志标签和标签值。 label键值对支持的运算符如下:
    - = : 等于
    - != : 不等于
    - =~ : 正则匹配
    - !~:正则不匹配

例如: {arms_app="product service"}, 查询名称为product service的应用的业务日志。

- 关键字:在日志标签后添加关键字可以进一步过滤业务日志。
   关键字支持的运算符如下:
  - |= : 包含关键字
  - != : 不包含关键字
  - |~ : 正则匹配
  - !~ : 正则不匹配

? 说明 关键字区分大小写。

例如:{arms_app="product service"}|="WARN",查询名称为product service的应用下Warn级别的业务 日志。

### 高级查询

## 查看日志

在业务日志区域,您可以执行以下操作:

单击日志所在行,可以查看目标日志包含的日志标签和关键字。

2021-11-23 16:57:44 {"log" Possit	12011-11-13 80157/44 WAN consider using a shorter
Log Labels:	
ul 🔍 🔾 app	tservice
.il 🕘 🔾 anns_app	tservice
.d ⊕ ⊝ anns_tenant,	ri_clusterid /4f98794cf805f1d29f6f8871c1
.al ⊕ ⊖ job	m-demo/productservice
ul ⊕ ⊝ namespace	.m-deno
.al 🔍 📿 pod	service-Ffddb4b0-zniðn
.d @ ⊝ anns_tenant,	r1_user1d +439941126
.al ⊕ ⊝ container	tservice
🔐 🕀 🖂 date	1002
🖃 🕀 🖂 filename	<pre>&gt;g/pods/arms-apm-demo// tservice/0.log</pre>
🦽 🕘 🖂 pod_templat	həsh ibb
Parsed Fields:	
al   TraceId	rfebő ams.console.aliyun.com
.d ⊚ ts	2021-11-2JT08:57:44.278Z
.d 💿 tsns	1637657864278697294

在日志展开信息区域,您可以执行以下操作:

- 单击标签前面的 图标,可以筛选包含当前标签和标签值的日志。
- 单击标签前面的 <>>> 图标,可以筛选不包含当前标签和标签值的日志。</>>
- 单击Trace ID右侧的控制台链接,可以查看对应的调用链路和业务轨迹。

如果您需要查看经过筛选后的某条日志的上下文,将鼠标悬浮于日志所在行,单击日志信息尾部的Show context。

展开的上下文默认前后各显示5条日志,如果您需要查看更多上下文,单击上下日志区域的Load 10 more。

POUND AD FORTS - ADMINI AD HIDE &
("log":"201-11-23 11:27:57 THF0 com.alibabeCloud.hipstershop.ToyProductController "trace_id": 9" - getProductList begin(n", "stream": "stdout", "time": "2021-11-23 TLI:27:57,3566380372")
("log":"2221-11-23 11:26:82 10=0
("leg"1921-1123 113/102 100 official files ("start") ()
("log": "2021-11-23 11:25:57 DF0
(*10g*/*2011-11-23-11:05:57-11#0 n***foldas*_/*categories**(*2x0e*)
y-i-isne*future':-/lmg/product
("log":2011.112:11:13:4) umm con_come=histor.pol.hoolass: "inter_101":%99100000000000000000000000000000000000
("log'1731-11-12 1113514) UNNI CONLEXONF MIRINGI, POOLSES ("TARE_1G") POSSIDU, CONSIGNE UNING A DOLTAR MIRINGIAL POOLSES ("MARE_1G") POSSIDU, CONSIGNE UNING A DOLTAR MIRINGIAL POOLSES ("MARE_1G")
("log";"%2014143 1112543 WAN com.taoser.hlikari.pool.RoolBase "trace_id";" Possibly consider using a shorter manifetime value.log","stream";"stdout", "ti
('log':'2821-11-23 11:25:43 INFO com.allbabacloud.hipstershop.ToyProductContr
("log";"2021-11-23 11:24:57 ID#O com.allbabacloud.hipstershop.consumer.Rocket
("log";"2021-11-23 11:24:48 TIMO com.allbabacloud.hipstershop.consumer.Rocket
("log":"#21-11-23 11:24:48 DH9 Com.allbabecloud.hipstershop.topProductContr
found 18 rose. Load 18 nore

# 查看日志详情

查看日志上下文

# 3.14.8. Arthas诊断(新版)

Art has是诊断Java领域线上问题的利器,利用字节码增强技术,可以在不重启JVM进程的情况下,查看程序 的运行情况。

### 前提条件

⑦ 说明 仅应用监控专家版支持Arthas诊断功能。

- 新开通ARMS的账号或15天内未使用旧版Arthas诊断功能的账号仅支持使用新版Arthas诊断功能。
   如果您在使用Arthas诊断中有任何问题,请搜索钉钉群号35396829进入ARMS应用诊断答疑群获取帮助。
- ARMS Agent 版本为v2.7.1.3或以上。
- 已接入应用监控。具体操作,请参见应用监控概述。
- 应用的编程语言需要为Java。
- 3658端口未被占用。

### 背景信息

ARMS提供的Arthas诊断功能主要用于补齐ARMS在实时诊断方面的能力。ARMS的Arthas诊断功能包括以下 几种类型:

- JVM概览:查看当前JVM进程实时的内存使用情况、系统信息、系统变量和环境变量。
- 线程分析:查看当前JVM进程的线程耗时情况以及指定线程的实时方法栈。
- 方法执行分析:抓取任意方法(非JDK方法)满足指定条件的一次执行记录,记录该方法的参数、异常、 返回值以及方法内部各个方法执行耗时。
- 对象查看器: 查看任意类的某个实例实时的属性取值情况。
- 实时看板:常见组件的实时看板,例如,Druid连接池的实时看板可以看到连接池的配置、使用情况以及 SQL执行耗时情况。
- 性能分析: 对CPU耗时、内存分配等对象进行一定时间的采样并生成相应的火焰图。
- Arthas Shell: 通过命令行方式使用Arthas诊断。

### 开通Arthas诊断功能

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏,选择应用监控 > 应用列表。
- 3. 在顶部菜单栏,选择地域。
- 4. 在应用列表页面,单击应用名称。
- 5. 在左侧导航栏中单击应用设置,并在右侧单击自定义配置页签。
- 6. 在**自定义配置**页签的**Arthas监控**区域,打开**Arthas开关**,根据需求选择是否仅对部分IP进行Arthas诊断,并添加目标IP。

Arthas监控			
Arthas开关:		控制是	否打开Arthas诊断功能,Agent 2.7.1.3 以上版本生效
生效实时IP:	XXX.300(.300(.30X,300(		如果填写,Arthas诊断仅针对指定ip生效,如果未配置则对所有IP生效

7. 在自定义配置页签左下角单击保存。

## 查看Arthas诊断信息

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏,选择应用监控 > 应用列表。
- 3. 在顶部菜单栏,选择地域。
- 4. 在应用列表页面,单击应用名称。
- 5. 在左侧导航栏,选择应用诊断 > Art has诊断。
- 6. 在Arthas诊断页面,在顶部下拉列表选择待诊断的应用实例。
  - 。 若该实例的Agent版本未升级至2.7.1.3或以上,页面会提示您需要先升级Agent。
  - 若该实例的Agent版本已升级至2.7.1.3或以上,页面会显示该实例的Arthas诊断信息。

### JVM概览

JVM概览支持查看应用的JVM相关信息,包括JVM内存、操作系统信息、变量信息等,帮助您了解JVM的总体情况。

Arthas诊断页面默认显示JVM概览页签,您可以在JVM概览页签查看以下信息:

• JVM内存: JVM内存的相关信息,包括堆内存使用情况、非堆内存使用情况、GC情况等。

JVM	M P\$#										
	<b>東型</b>	已使用量	8.8	最大量	GC開閉	2.2.B	品料时				
+	heap	195545496	253493248	4026138624	copy	750	15719				
+	butter_pool				marksweepcompact	26	7668				
+	nonheap	106618760	194707456	-1							

• 操作系统信息:操作系统的相关信息,包括平均负载情况,操作系统名称、操作系统版本、Java版本等。

at 1 Southerde						
systemi.cadilverage	0.419921875					
ostiersion	3.10.0-1002.18.1.e7.x06,64					
javaNersion	18.0,212					
processors	1					
ooName	Linux					
javaHome	/uar/lib/jem/jaua-18-openjak/jm					
timestamp	1614668322050					
uptime	540791					

• 变量信息:变量的相关信息,包括系统变量和环境变量。

安量信息							
系统变量		请输入系统支票关键字规索(	Q	环境变量		请输入环境交量关键字搜索	Q
awt.toolkit	sunawtX11XToolkit		Î	РАТН	/usr/local/sbin/usr/local/bin/usr/sb 1.8-openjdk/jre/bin/usr/lib/jvm/javi	ins/use/bins/bins/use/lib/jvm/j s-1.8-openjdk/bin	-
armsagertargs	edasAppl			ACK_JENKINS_DEFAULT_PORT_8080_TCP_ADDR	172.0		
dpath.id.group	defaultGroup			CHECKOUT_MYSQL_SERVICE_PORT	3306		
Reencoding.pkg	sunio			INTRANET_CART_LZOH7Y_SERVICE_HOST	172		
java.specification.version	1.8			ededd	14512		
profiler.micro.service.mock.enable	true			CART BEDIS SERVICE LIGHT	172		
suncpuisalist							
and the same data.	1177 A		•	INTRAKET_CHECKUOT_VOCHIEC_SERVICE_HOST	172		-

### 线程耗时分析

线程耗时分析支持显示该应用的所有线程和查看线程的堆栈信息,帮助您快速定位耗时较高的线程。

在Arthas诊断页面,单击线程耗时分析页签。
 线程耗时分析页签会实时获取当前JVM进程的线程耗时情况,并将相似线程聚合。

NEW:	1: 0 TERMINARE: 0 RUNNARE: 0 RUNNARE: 0 RUNTING: 1374 TIMED, WATTING: 6627 C								
	Name		⊕CPU ≹		数量 (1				
-	arthas-command-execute		88.09		1				
	Name	ld	Сри	State	操作				
	arthas-command-execute	135584	88.09	RUNNABLE	查看实时堆栈				
+	VM Thread		13.18		1				
+	$com.mchangev".async.ThreadPoolAsynchronousRunnersPoolThread-\#^*$		5.82		6				
+	metrics-exporter-*-thread-*		1.7		4				
+	C* CompilerThread*		2.29		4				
+	NioBlockingSelector.BlockPoller-*		0.51		1				
+	Pinpoint-TcpDataSender(SpanDataSender)-Executor(*-*)		0.5		1				
+	http://io-*-ClientPoller-*		0.67999999999999999		2				
+	Pinpoint-Client-Worker(*-*)		0.34		8				
+	http-nio-"-Acceptor-"		0.33		1				
		< +	0 1 2 3 4 5 6 7	8 9 ··· 28 下	5 > 1/28 王師 3	100			

- 2. 单击线程左侧的+图标展开线程明细,可以查看线程的ID、CPU使用率和状态。
- 3. 如需查看目标线程的堆栈信息,您可以在目标线程右侧的操作列,单击查看实时堆栈。

线程详情	×
[arthas-command-execute]       [RUNNABLE]         sun.management.ThreadImpl.dumpThreads0       (ThreadImpLjava 428)         com.taobao.arthas.core.command.monitor200.ThreadCommand.processThread       (ThreadCommand.java 226)         com.taobao.arthas.core.command.monitor200.ThreadCommand.process (ThreadCommand.java 120)       com.taobao.arthas.core.shell.command.impl.AnnotatedCommandImpl.process (AnnotatedCommandImpl.java 82)         com.taobao.arthas.core.shell.command.impl.AnnotatedCommandImpl.process (AnnotatedCommandImpl.java 18)       com.taobao.arthas.core.shell.command.impl.AnnotatedCommandImplSProcessHandler.handle       (AnnotatedCommandImpl.java 111)         com.taobao.arthas.core.shell.command.impl.AnnotatedCommandImplSProcessHandler.handle       (AnnotatedCommandImpl.java 111)         com.taobao.arthas.core.shell.command.impl.AnnotatedCommandImplSProcessHandler.handle       (AnnotatedCommandImpl.java 111)         com.taobao.arthas.core.shell.commandImpl.SprocessHandler.handle       (AnnotatedCommandImpl.java 108)         com.taobao.arthas.core.shell.commandImpl.AnnotatedCommandImplSProcessHandler.handle       (AnnotatedCommandImpl.java 110)         com.taobao.arthas.core.shell.commandImpl.AnnotatedCommandImplSProcessHandler.handle       (AnnotatedCommandImpl.java 110)         com.taobao.arthas.core.shell.commandImpl.SprocessImplSCommandProcessIsasCommandImpl.java 385)       java.util.concurrent.StructorisKun (FutureTask.java 266)       (ScheduledThreadPoolExecutorSCheduledItureTask.accessS201       (ScheduledThreadPoolExecutor.java 180)       java.util.concurrent.ScheduledThreadPoo	

# 方法执行分析

方法执行分析支持抓取方法的某一次执行的耗时、入参、返回值等信息和钻入,帮助您快速定位导致慢调用 的根本原因,以及问题线下无法复现或日志缺失等场景。

- 1. 在Arthas诊断页面,单击方法执行分析页签。
- 2. 在方法执行分析页签的搜索框中输入类名的关键词,然后单击Q图标。
- 在搜索到的类中选择需要诊断的类,然后在右侧方法选择框选取该类的某个方法,单击确定。 页面将会显示ARMS随机抓取的该方法的某一次执行的信息。

制技 点面量有方法原则						c	方话tioGetDatas执行结束	
编制方法	44	R#88		时间题(ma)		NIT .	(33(8): 684.31m	traceld: e78334c6162942863205628862
om alibaba arms conside web module. Ten api TraceAction doGetDatas	-1		684ms				~ ####	
con albabs anns core service syst en base EnvironmentService.taDes	008		Oms			BA	PR0.	
java.lang.System.currentTimeMillis	612		Oms			BA	"special" : "	
com al baba fastjison. JSON parseOb RCT	613		Oms			BA	"adeptions" (* 0164392033537622)" "realizame" (* 016439203537622)" "stebdiame" (* 016439203537622)"	
con alibaba anns metric bean Metri :Quary getFilters	614		Oms			BA.	"Bad" : "child" s true "crimina" :	
contalibaba arms console webs. Ar msCts.getUserid	634		Oma			MA.	"context" : "Pullablemappedcasters ( arrayotil ="	
java.uti.Map.put	634		Oms			MA.	"tep6%" = "41ed1s2455es "isseemetawork" = false	
con alibaba arres metric bean.Metri ciquery.InNeedUseSis.Alg	678		Oms			MA.	"iframekkes" : false ) @Rti	
con albaba arms metric bean Metri cQuery setNeecUseSIsNg	672		Oma			94.X	("netric")" apparat.eq")" measures" ("H")" count")" is net ("dimensions") ("eqlid"), "mensions" ("eqlid"), "mensions") ("eqlid"), "mens	183647/18eev" ("regionid": "on-hangzhou";"pid": "16647625
concalibaba arms concension metr- ic handler.MetricDataHandler.getDat a	629		683ms			BA.	> asse 说置doGetDatas执行条件	
org.apache.commons.lang.esceptio n.ExceptionUtils.getFullStackTrace	631				Orms	NA.	请我择一个重载方法	
org.s84j.Logger.error	631				Irra	NA.	daGetlatas(ArmoCta, String) 🗸 🗸	
java.lang.System.currentTimeMills	643				Oms	RA.	11Ritidkey Williamsky	是正位第千位
con albaba arms bean ArmsDataR essit-cinits	055				ans	BA.	aatagreeasaan	
con alibaba arms bean ArmsDataR esalt.setSuccessful	056				Ome	MA.	a V	
con al baba anna bean AnnaDataR esait, setEntMsg	457				ūms	NA.	* 请被人还读做	
con albaba arms console webs.Ar	659				0ms			

- 左侧执行堆栈区域显示诊断方法的内部执行记录。
  - 如需钻入某个内部方法,在其右侧操作列,单击钻入。
  - 如需查看方法源码,单击执行堆栈区域顶部的查看方法源码。
     如下图所示,每一次内部方法的执行耗时都会以注释的方式显示在源代码中。

	G
● 免费声明 西考大地理品会社図 Iuaa FileStat+ZIDPEE回15字世々会社/PAL 12420回正第三	
com.alibaba.arms.console.web.module.action.api.TraceAction $\checkmark$ 🕜	
<pre>1 public vid defeDibisiAmaCic (ix, gBaranivalue*@perfy) String parami {</pre>	

- 右侧方法执行结果区域显示方法执行的参数值、返回值、异常、成员变量以及此次方法执行的 TraceID。
- 右侧设置执行条件区域执行以下步骤,可以设置方法执行条件来抓取满足条件的方法执行记录。
   a. 选择当前方法中的一个重载方法。

b. 在请选择初始过滤key下拉框选择初始过滤Key的类型,单击《图标。

初始过滤Key的类型:

- params[n]: 方法的第n个参数。
- returnObj: 方法的返回值。
- 方法执行耗时: 方法执行的耗时。
- 是否抛出异常:方法执行时是否抛出异常。

⑦ 说明 如果选取的初始过滤Key为嵌套类型,则还需要继续选择该嵌套类型的内部字段,直到选择字段为基础类型。

- c. 选择过滤条件。
- d. 输入过滤值。
- e. 单击添加。

在当前过滤条件区域会显示已添加的过滤条件。

设置doGetDatas执行条件	C
请选择一个重载方法	
doGetDatas(ArmsCtx, String) 🗸 🗸	
当前过滤key userld类型为java.lang.S	是否切换手动输入
params[0].userid 🤾 请选择 🗸	
, 选择过滤条件	
= 🗸	
"请填入过滤值:	
123	
添加 重置	
当前过滤条件:	

f. 在左侧执行堆栈区域右上角单击 C图标, 系统会按照设置的条件重新抓取一次方法执行。

### 对象查看器

对象查看器用于查看一些单例对象当前的状态,用于排查应用状态异常问题,例如应用配置、黑白名单、成员变量等。

- 1. 在Arthas诊断页面,单击对象查看器页签。
- 2. 在对象查看器页签的搜索框中输入类名的关键词,然后单击搜索图标。
- 在搜索到的类中选择需要诊断的类,然后在右侧实例选择框选择该类的某个实例,单击确定。 页面则会显示该实例中当前字段的实时取值。

	CenterServiet	Q instance[0] 查看toString(結果		~	· 通定 - 重量
instances[0]					
┃ 对象字段实时取值 当前英	com alibaba arms console serviet.trace Event	CenterServlet	C	字段详情	C
请输入关联信息关键字搜索	Q			李龄名称	
與型	李鉛名	洋街		IStrings	
String	LSTRING_FILE	当前值: "javax.servlet.http.LocalStrings"		字段类型	
PropertyResourceBundle	Strings	点击重要详疑		java.util.PropertyResourceBundle	
Long	serialVersionUID	当約進 1		toString()總罪	
StandardWrapperFacade	config	点出查審详情		java.ubil.PropertyResourceBundle@7e4e5bd8	
<u> </u>	7. 10 - 10 - 10 - 10 - 10 - 10 - 10 - 10			<pre>wight().059/0-05cat_EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE</pre>	

○ 对于简单类型的字段,在左侧对象字段实时取值区域的详情列会是字段的取值。

 对于复制类型的字段,在左侧对象字段实时取值区域的详情列单击点击查看详情,在右侧字段详 情区域查看字段取值详情。
 字段详情区域仅支持将复杂类型字段反序列化一层进行展示,如果需要查看更具体的数据,单击点 击查看字段更详细的数据。

### 实时看板

实时看板用于查看系统中用到的关键组件的实时状态,例如查看数据库连接池的使用情况、HTTP连接池的使用情况等,有利于排查资源类型的问题。

- 1. 在Arthas诊断页面,单击实时看板页签。
- 2. 在**实时看板**页签的下拉列表中选择一个看板,然后在右侧实例选择框选择该看板的一个实例,单击添 加到看板。

页面显示该看板的实时情况。如下图显示为一个Druid连接池的实时状态信息,包括基础配置、连接池状态、执行耗时分布等。

Druid海损地看板				<ul> <li>✓ instance]</li> </ul>	0] ##toString()	8 <b>7</b>	~	
Druid酒板								с×
DbType 💮	mysql	URL @	jdbc:mysql://rm-a.	QueryTimeout @	0	TransactionQueryTimeout (*)	0	
WaitThreadCount 🕐	0	ActiveCount (*)	1	PoolingCount (2)	28	MaxActive (1)	3000	
PoolingPeak (*)	0	PoolingPeakTime (*)	1	ActivePeak	3000	ActivePeakTime (*)	1	
ExecuteCount (1)	2118117	ErrorCount (?)	157	CommitCount (1)	9118	RollbackCount (?)	455	
• connection占用耗时分布 2				• 事务执行耗时分布 8800				
1				4400				
0 -1 小于Ims Im	s – 10ms 10ms	-100ms 100ms-1s 109	s-100s 大于100s	2200 0 小于1ms 1m	s – 10ms 10ms – 10	1ms 100ms - 1s 1s - 10s 10s	-100s 100s-1000s	

### 性能分析

性能分析支持对CPU耗时、内存分配等对象进行一定时间的采样并生成相应的火焰图,帮助您快速定位应用的性能瓶颈。

- 1. 在Arthas诊断页面,单击性能分析页签。
- 2. 在性能分析页签下方,单击新建火焰图。
- 3. 在新建火焰图对话框,选择火焰图类型,输入采样时间和备注信息,然后单击确认。

参数	描述	示例值
----	----	-----

参数	描述	示例值
火焰图类型	采样对象的类型。取值: <ul> <li><i>cpu耗时</i></li> <li><i>内存分配</i></li> <li><i>锁耗时</i></li> <li><i>itimer</i></li> </ul>	cpu耗时
输入采样时间(单位:秒)	采样的时长。取值: 10~1800。	30

**性能分析**页签下方显示已创建的火焰图的任务信息,包括开始时间、采样时间、备注信息、火焰图类型 和任务状态。

1880/350/187387187 Q C							
开始时间	采和年期	餐注信息	火焰骤频型	任务状态			
2021-12-20 15:32:36	30		сри	<u> </u>			
2021-12-07 14:46:26	40	asclasda	cpu	重要火焰图			
2021-12-07 14:28:42	30		alloc	<b>查看</b> 火运图			
2021-11-25 13:49:24	30		alloc	查看火烧器			

4. 在**性能分析**页签下方,找到任务记录,在其右侧**任务状态**,单击**查看火焰图**,根据页面提示下载SVG 格式的火焰图文件,然后在浏览器中打开。



Arthas Shell 通过命令行方式自定义Arthas诊断。



# 3.15. JVM配置优化

在JVM配置优化页面您可以查看当前应用下所有的JVM实例,以及实例配置信息。同时,ARMS应用监控还支持对JVM参数调优,为您提供JVM参数优化建议。

## 功能入口

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏选择应用监控 > 应用列表,并在顶部菜单栏选择目标地域。
- 3. 在应用列表页面单击目标应用的名称。
- 4. 在左侧导航栏中单击JVM配置优化。

在JVM配置优化页面,您可以查看当前应用下所有的JVM实例。

5. 单击目标实例所在行,查看实例配置信息,包括JVM参数、实例IP、启动时间、运行时长、主机名、进程 号和JVM版本。

实例信	息				
实例IP		运行时长	进程号	JVM版本	ARMS探针版本
-	47	22 Days 6 Hours 3 Mins	1	1.8.0_312	1.7.0-SNAPSHOT_2.7.1.3_5f7d8db
10.000.2	6	22 Days 6 Hours 2 Mins	1	1.8.0_312	1.7.0-SNAPSHOT_2.7.1.3_5f7d8db
配置	配置 JVM参数调优				
JVM参 政	[+Doneagent.plugin.arms-agent.ena Doneagent.region=cn-hangzhou, -	abled=true, -javaagent:/home/admin/.opt/ArmsAgent/arms-bootstrap-1.7.0-SNAP Xms1024M, -Xmx3072M, -XX:+PrintGCDetails, -XX:+PrintGCDateStamps]	SHOT.jar, -Darms.licenseKey:	Ppolicisticita, Inc. of	compagnitudents, becapter-408, becaptacitude-injusts
实例P	.147				
启动时 间	2022-03-09 11:36:45				
运行时 长	22 Days 6 Hours 3 Mins				
主机名	Northern Technologies Cyll				
进程号	1				
JVM版 本	1.8.0_312				

### JVM参数调优

- 1. 在JVM配置优化页面单击JVM参数调优。
- 在JVM参数调优面板选择供应商、CPU核数和内存大小,输入调优前的JVM参数,然后单击参数检测。
   在优化后参数、原参数优化建议和具体优化动作区域查看JVM参数优化建议。

参数输入		
va版本: 1.8.0_312 ∨ 供应商:	OpenJDK V CPU核数:	4 内存: 8 GiB ~
-Doneagent.plugin.arms-agent.enabled=t Darms.licenseKey= Darms.agent.podinfo.path=/etc/podinfo XX:+PrintGCDateStamps	rue -iavaagent:/home/admin/.opt/ArmsA -Doneagent.region=cn-hangzhou -Xms10	gent/arms-bootstrap-1.7.0-SNAPSHOT.jar - - 124M -Xmx3072M -XX:+PrintGCDetails -
· · · · · · · · · · · · · · · · · · ·	鼠标悬停在参数标签可查看修改原因	
-Doneagent.plugin.arms-agent.enable	-javaagent:/home/admin/.opt/ArmsA	-Darms.licenseKey=
-Darms.appId=	-Darms.agent.env=ACSK8S	-Darms.agent.podinfo.path=/etc/podi
-Doneagent.region=cn-hangzhou	-XX:+PrintGCDetails	-XX:+PrintGCDateStamps
-Xmx3072M	-Xms1024M	-Xmn2121m
-XX:MetaspaceSize=512m	-XX:MaxMetaspaceSize=512m	-XX:+PrintGCTimeStamps
	参数检查	
优化后参数		
<ul> <li>现有规则不一定适合所有应用,</li> <li>内存,可能会导致OOM Killeri是;</li> </ul>	生成内容仅供参考,上线前须严格测 杀。	试JVM参数是否满足业务需求。比如Java非独占
-javaagent:/home/admin/.opt/ArmsAg	-Xms5734m	-Xmx5734m

### JVM参数检查规则说明

参数检查通过规则引擎Drools实现。

○ 注意 默认规则不适合所有应用,需严格测试JVM参数是否满足系统需求。例如一般推荐Java进程 占用70%主机内存,但若Java未独占主机,则Java进程可能被系统OOM Killer误杀。

### 通用规则

参数存在性: ARMS应用监控记录了Java每个版本的所有参数(如果参数不在列表中页面会出现提示),并 会基于字符串相似度算法进行模糊匹配,快速发现并纠正简单的拼写错误。

参数语法:

• 参数类型:

0	标准参数:	-server	•	-verbose	•	-javaagent	•	-version	等。
0	非标准参数	-Xmx1g	•	-Xsslm	等。				
0	高级参数:	-XX:+UseG	GlG	等。					

⑦ 说明 如果参数类型有误,应用监控会自动识别并给出提示。

- 值类型:
  - 内存型: -Xmx1g
  - 整型: -XX:CMSTriggerRatio=75
  - 浮点型: -XX:InitialRAMPercentage=50.0
  - 字符串型: -XX:HeapDumpPath=/disk2/dumps
  - 布尔型: -XX:+UseConcMarkSweepGC
  - 字符串列表型: -cp=/path1:/path2:/path2

⑦ 说明 如果值类型有误,应用监控会自动识别并给出提示。

参数过时:

- Deprecated:当前Java版本支持但未来有可能不支持的参数,例如在Java 8中使用 -XX:MaxGCMinorPause Millis=109。
- Obsolete:当前Java版本已不支持的参数,当您使用此参数时,JVM会给出警告,然后忽略此参数并继续 执行。例如在Java 11中使用 -xx:+UseAppCDS 。
- Expired:当前版Java版本已不支持的参数,当您使用此参数时,JVM会给出警告并退出。例如在Java 11中 使用 -xx:+CMSIncrementalMode 。

⑦ 说明 如果使用了过时的参数,应用监控会自动识别并给出提示。

参数重复:当一个参数重复出现时,出现在参数列表尾部的参数会覆盖出现在前面的参数。应避免出现参数 重复定义,因为后期如果只修改了靠前位置的参数,会导致修改无法生效。

参数值合理性:

- 参数值和默认值相同。
- 参数值超过了正常的范围。

⑦ 说明 出现以上情况时,应用监控会自动识别并给出提示。

参数关系:参数之间存在依赖、互斥等关系。例如想要使用实验性参数,则必须额外增加 -XX:+UnlockExperimentalVMOptions ,例如在Java 8中使用 -XX:G101dCSetRegionThresholdPercent=10 。

### 内存管理相关规则:

- 必须参数:
  - 堆上限值: -Xmx1g
  - o 堆初始值: -Xms1g

⑦ 说明 如果没有指定以上参数,应用监控会自动识别并给出提示。

- 参数关系:
  - 参数冲突: CMS和G1无法一起使用,例如 -XX:+UseConcMarkSweepGC 和 -XX:+UseG1GC 会导致JVM 无法启动。
  - 参数依赖:
    - CMS高级参数需要以开启CMS为前提。例如,如果使用CMSInitiatingOccupancyFraction和 UseCMSInitiatingOccupancyOnly,则必须开启CMS( -xx:+UseConcMarkSweepGC )。
    - G1高级参数需要以开启G1为前提。
- 参数值合理性:
  - 新生代的大小不应该超过初始堆大小。
  - 如果Java独占主机, 堆空间上限一般推荐为主机内存的70%。70%是经验值, 对大多数场景有意义, 但存在某些场景下不适用。
  - 面向吞吐或者低延迟的应用, 初始堆大小一般建议设置为与堆空间上限相同。

。 GC线程数不应该超过CPU核心数。

### 版本相关规则

JDK 8

参数过时:不支持PermSize和MaxPermSize,需要修改为MetaSpaceSize和MaxMetaSpaceSize。

内存管理:

- 必须参数:
  - 当未使用G1GC时,一般必须指定新生代空间( -xmn1g )。
  - 必须指定MetaSpaceSize。
  - 必须指定MaxMetaSpaceSize。
  - 当使用CMS时,一般也需要开启ExplicitGCInvokesConcurrentAndUnloadsClasses。
  - 建议打开的GC参数: -XX:+PrintGCTimeStamps 、 -XX:+PrintGCDateStamps 和 -XX:+PrintGCDet ails 。
- 参数关系: MetaspaceSize不能超过MaxMetaspaceSize。

### JDK 7

内存管理下的必须参数:

- 当未使用G1GC时,一般必须指定新生代空间( -Xmn1g )。
- 必须指定PermSize。
- 必须指定MaxPermSize。
- 当使用CMS时,一般也需要开启Explicit GCInvokesConcurrent And UnloadsClasses。
- 建议打开的GC参数: -XX:+PrintGCTimeStamps 、 -XX:+PrintGCDateStamps 和 -XX:+PrintGCDetail s 。

# 3.16. 应用环境

ARMS应用监控支持查看ACK集群下Dubbo应用的环境信息,包括环境依赖信息和配置信息。

### 前提条件

为容器服务Kubernetes版Java应用安装探针

### 功能入口

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏,选择应用监控 > 应用列表。
- 3. 在顶部菜单栏,选择地域。
- 4. 在应用列表页面,单击应用名称。
- 5. 在左侧导航栏,单击应用环境。

## 依赖信息

### 依赖信息页签显示了应用依赖的注册表、软件、RPC框架等版本信息。

依赖信息	配置信息	
类型		包名
Registry and Co	nfig Center	nacos-
Others		classmi Roaring commo snakey, classes netty-al httpasy reflectii shims-C guava-3 httpcor hiberna eclipse simplec javassie httpclie httpcor hiberna eclipse
RPC Framework	S	dubbo- dubbo- dubbo- dubbo-
		spring- tomcat

# **配置信息** 配置信息页签显示了Dubbo的配置信息。

依赖信息 配置信息	3	
Dubbo		
属性	内容	描述
charset	U77-8	序列化编码
refreshed	true .	
accepts		服务提供方最大可接受连接数
qosEnable	false	是否开启 Dubbo Qos 功能
logger	101	日志输出方式,可选: slf4j,jcl,log4j,log4j2,jdk
threads	200	服务线程池大小(固定大小)
threadpool	fixed	线程池类型,可选: fixed/cached
valid	true .	
serialization	hessiari2	协议序列化方式,当协议支持多种序列化方式时使用,比如:dubbo协议的 dubbo,hessian2.java,compactedjava,以及http协议的json等
metaData	("heatmane"/"dubbs-text: 7986;74794- 2997;"protocol"/"dubbs-fromtet"/"dubbs-provider- denne"/"dubbs-provider- denne"/"gros.anable!"/"fabar"/"gros.anable.compatible!"/"fabar"/	微服务元数据
registry-0	racos (Inse-be245cc0-p.nacos-ans.mae.alyuncs.com.8848	注册中心服务器地址,如果地址没有端口缺省为9090,同一集群内的多个地址用逗号分隔, 如:ip:port,ip:port
codec	skaliter	协议编码方式

# 3.17. 应用设置

# 3.17.1. 自定义配置

应用监控的一些常用设置,例如调用链采样率、Agent开关、慢SQL阈值等,可直接在**自定义配置**页签上配置。

# 前提条件

### 创建应用监控任务

### 功能入口

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏选择应用监控 > 应用列表,并在顶部菜单栏选择目标地域。
- 3. 在应用列表页面单击目标应用的名称。
- 4. 在左侧导航栏中单击应用设置,并在右侧页面单击自定义配置页签。
- 5. 设置自定义配置参数,设置完毕后,在页面底部单击保存。

### 配置调用链采样设置

在**调用链采样设置**区域,可以打开或关闭调用链采样开关,并设置采样率。**采样率设置**字段输入百分比的 数字部分即可,例如输入10代表采样10%。

└ 注意 修改即时生效,无需重启应用。如果关闭采样,则调用链数据将不会被采集,请谨慎操作。

Agent	自定义配置	标签管理	监控方法自定义	自定义参数	删除	
调用链采样设	置					
	采样开启:	D	说明:关闭此配置,调	用链功能将关闭,请您谨慎措	曩作!	
*	采样率设置: 10	00	说明:采样率默认为10	。调大采样率可能会消耗额外	卟的系统资源,	请您谨慎操作!

ARMS基础版支持设置客户端采样策略,并按照采集数据行数进行计费。ARMS默认免费为您账号下所有接口 的每个探针每分钟采集一条调用链。除此之外,您还可以添加自定义的采样策略。

### 客户端采样策略设置项

设置项	说明
策略名称	自定义采样策略名称。
采样类型和采样值	<ul> <li>固定比例采样:按照设置的固定比例进行调用链采样。若选择此选项,则采样值设置为固定比例,如10%。</li> <li>流量限额:若选择此选项,采样值需设置每个探针在设定时间范围内采集的调用链条数,如5条Trace/每个探针每1秒,表示每个探针每1秒钟采集5条调用链。</li> </ul>
	设置采样策略生效的接口范围,可选择 <b>每个接口</b> 或 <b>指定接口</b> 并输入指定的接 口名称。
适用接口	<ul> <li>⑦ 说明 目前每个采样策略在选择指定接口时,仅支持输入一个接口</li> <li>名。若需对多个接口进行调用链采样,则需设置多个采样策略。</li> </ul>

完成采样策略设置后,您可以在控制台选择是否开启该策略。多个采样策略会同时生效,但存在优先级如下:默认采样(免费)>单接口流量限额>单接口固定比例采样>全部接口流量限额>全部接口固定比例采样。您也可以编辑已添加的采样策略,或删除不需要的采样策略。

调用链采样设置 🥐						
是否采集调用链:	是否采集调用链: 《 说明:关闭此起置,调用链功能样关闭,请您 <mark>谨慎操作</mark> !					
• 客户端采样设置:	添加客户端采样策略					
	基础采样策略名称	采样策略适用范围	流量跟额 ?	固定比例采样	是否启用	操作
	默认采样(免费)	每个接口	1条Trace / (每个採针每60秒)			编辑 删除
	test,	/home/assets		100%		编辑 删除
	test_	每个接口	5条Trace / (每个採针每1秒)			编辑 删除

## ARMS基础版

# 配置Agent (探针)开关和日志级别 在Agent开关配置区域,可以打开或关闭探针总开关以及各插件开关,并配置日志级别。

↓ 注意 探针总开关和日志级别的修改即时生效,无需重启应用。如果关闭探针总开关,则系统将无法监控您的应用,请谨慎操作。要使对各插件开关的修改生效,必须手动重启应用。

Agentification						
Agent总开关:		说明:修改动态生效,为	无需重启应用。关闭此配置,	,系统将无法监控您的应用,	也不会产生费用,请您 <mark>谨慎操</mark> 作!	
插件开关:	dubbo-plugin	🗹 mongodb-plugin	🖌 ali-hsf-plugin	httpclient3-plugin	✓ httpclient4-plugin	
	🗾 jdk-http-plugin	🗾 jetty-plugin	🗹 mybatis-plugin	🔽 mysql-plugin	okhttp-plugin	
	🗹 oracle-plugin	postgresql-plugin	🗹 redis-plugin	spring-plugin	springboot-plugin	
	🗹 tomcat-plugin	lettuce-plugin	dirpc-plugin	🗹 thrift-plugin	google-httpclient-plugin	
	hystrix-plugin	🗹 rxjava-plugin	注意:插件开关修改,手	戶工重启应用方可生效		
日志级别配置:	WARN 🗸	说明: Agent日志级别修	1改,动态生效,无需重启应	如用.		

# 配置阈值设置

在阈值设置区域,可以设置慢SQL查询阈值、接口响应时间阈值和限流阈值。

阈值设置		
* 慢SQL查询阈值:	500	说明:单位为毫秒,默认500ms。当SQL查询的耗时大于该阈值的时候,该查询会被标记为喂SQL。
* 接口响应时间阈值:	500	说明:单位为毫秒,默认500ms。当接口响应时间大于该阈值的时候,该接口会被标记为慢调用。
* 限流阈值:	100	说明:Agent诱每秒最大可处理请求数,默认100条。大于该阈值的调用链,不被收集。

## 配置高级设置

在高级设置区域,可以设置需过滤的接口、方法堆栈最大长度等。

### 应用监控·控制台功能

高级设置	
无效接口调用过滤:	/**/*.jpg/**/*.jpg/**/*.jpeg
* 方法堆栈最大长度:	128 词明: 数以为128条,契持最大长度400条。
• 同类异常堆栈区分深度:	2 说明:同些型异常,用于作为不同异常区分的操线深度,一般设置为第一个差异调用的深度
采集SQL最大长度:	1024 说明: 默认为1024个字符,最小长篮: 256. 最大长篮4096.
采集SQL绑定值:	议明:摘获PrepareStatement参数协定的变量值,无需要自由用的生效。
原始 SQL:	说明: 0235QL载新,不做期外处理。
异常过途:	使用は正則用点式正和評判用を注意。 多个情感用意义注意: " 598. 例如: jesking intempted sception, jesking indeoDudDBaundit.sception
	说明: 试配置用于过速应用洋播页面的异%分析因表所展示的异常装置。 查看异常题表
错误数过滤:	駅以いTTP大売売 - 400 行力増決度物時計、可以在於設置需要的場所的描示符。多个 増減時便可定文遺号分隔。50 429 病毒 429.512 (Agent版本高子 2.5.7.2)
调用链新格式:	送胡: 从用压力合调用抽合加存合加存得都指式(增以4377)
调用链压缩:	说明: 显否将重复调用 (比如 for 循环) 能化,无需重回应用即生效
请求入参最大长度:	512 说明:默认为512字符,支持的最大长度为2048字符。
分位数统计:	说明:是否开启分位数统计功能。
RabbitMQ自定义消费者:	通过公司最有论法包运费和合规是有的"展在内部间最有效的决名,可以展到前间 要者的通问检查。多个消费者以及文型号分配。
缓程池监控:	(3) 認用: 支持TomatOubbohKSF每垣間的低型法用标应法: 美介吸Agen研究社医教育成本,注意: <u>見見重要主点</u>
异步适传扫描包名:	语号分额的稳定。当这个场下的 Rumable/Caltable/Suppler 在施旗时,会自动 捕获 Trace 上下文并传递到新续程。 注意: 但用量包生效,Agent 27.13 双以上解年变换。
请求中返国TraceId	C CHYSthttp是型的读求。在iresponse header中级则才留起了eaglerge-traceid".

- 无效接口调用过滤: 输入不需要查看调用情况的接口, 从而将其从接口调用页面隐去。
- 方法堆栈最大长度:默认为128条,最大值为400条。
- 同类异常堆栈区分深度:同类型异常,用于作为不同异常区分的堆栈深度,一般设置为第一个差异调用的深度。
- 采集SQL最大长度:默认为1024个字符,最小值为256,最大值为4096。
- 采集SQL绑定值: 捕获PrepareStatement参数绑定的变量值, 无需重启应用即可生效。
- 原始SQL: 仅对SQL截断,不做额外处理。
- 异常过滤:此处输入的异常不会显示在应用详情和异常分析页面的图表中。
- 错误数过滤:默认情况下,大于400的状态码会计入错误数,您可以自定义大于400但不计入的HTTP状态 码。
- 调用链新格式:采用支持调用链时间排序的新存储格式(默认打开)。
- 调用链压缩: 是否将重复调用(比如for循环)简化, 无需重启应用即生效。
- 请求入参最大长度:默认为512字符,支持的最大长度为2048字符。
- **分位数统计**:是否开启分位数统计功能。
- Rabbit MQ自定义消费者:通过设置自定义消费者的类名或者包含匿名内部消费者的类名,可以看到该 消费者的调用链路。多个消费者以英文逗号(,)分隔。
- **线程池监控**:支持Tomcat/Dubbo/HSF等框架的线程池指标监控,需升级Agent探针至最新版本。该设置需重启应用后生效。
- 异步透传扫描包名:添加异步透传扫描包实现异步任务监控。异步透传扫描包中的Runnable、Callable和Supplier接口在创建新对象时会自动捕获当前线程调用链的上下文,并在异步线程中执行时使用该调用链上下文,完成串联。探针版本必须为v2.7.1.3及以上。
- **请求中返回TraceId**: 仅针对HTTP类型的请求,在Response Header中返回字段为 eagleeyetraceid 。

### 配置线程设置

在线程设置区域,可以打开或关闭线程诊断方法栈开关、线程剖析总控开关,并设置慢调用监听触发阈值。

线程设置		
线程诊断方法栈:		说明:开启后每隔5分钟采集一次方法线
线程制析总控开关:		开启后自动保存便调用本地方法线
慢调用监听触发阈值:	2000	(单位: ms) 耗财离于该调值才启动线程制所,建议设为耗财的99分位线。低于2000ms会增加CPU消耗,不可小于500ms。

⑦ 说明 服务调用耗时超过慢调用监听触发阈值(默认值为1000毫秒)时才会启动监听,并一直持续 到该次调用结束或超过15秒。建议将此阈值设为调用耗时的第99百分位数。假设有100次调用,则按耗 时从小到大排序,排在第99位的耗时就是第99百分位数。

### 设置关联业务日志与Traceld

在**业务日志关联设置**区域,可以设置是否在应用的业务日志中关联调用链的Traceld信息。更多信息,请参考业务日志关联调用链的Traceld信息。

⑦ 说明 仅应用监控专家版支持该功能。

业员日本关款设置 🚱		
关联业务日志与Traceld	开启后业务日志中会自动生成调用输的Traceld,此设置在重启应用后生效。支持	Nog4jAog4g2Aogback日志姐妹、业务应用英鞭在日志的Layout中通过声明%XEagleEye-TraceIO供输出TraceId信息。
日志服务绑定(例: 已接权,清绑定对应的)。	oject及Logstore,并关联索引支持指定字段索引及全文索引 区别。《 有未配置,清晰	·表目志服务控制命进行配置。
* Project名称:	请选择	98∎Project∂
* Logstore名称:	·请1549 ~	
关款(索引):	诸选择	

### 配置URL收敛规则

在URL收敛设置区域,可以打开或关闭收敛功能的开关,并设置收敛阈值、收敛规则和排查规则。URL收敛 是指将具有相似性的一系列URL作为一个单独的个体展示,例如将前半部分都为/service/demo/的一系列 URL集中展示。收敛阈值是指要进行URL收敛的最低数量条件,例如当阈值为100时,则符合规则正则表达式 的URL达到100时才会对它们进行收敛。

URL收敛设置		
收敛URL:		
收敛阈值:	100	说明:大于此阈值即进行收敛。
收敛规则正则:		
	示例:/service/(.*?	)/demo。 注: 多个规则之间以英文逗号分隔。
排除规则正则:		
	示例: /service/\d+	-/demo。 注: 多个规则之间以英文逗号分隔。

### 设置业务监控

在**业务监控设置**区域,可以打开或关闭业务监控开关,并设置HTTP编码。

业务监控设置		
业务监控开关:		控制业务监控是否生效。Agent 2.6.2以上版本支持
HTTP编码:	UTF-8	用于对HTTP参数解析,默认UTF-8,请按实际情况设置

### 设置Arthas监控

在**Arthas监控**区域,可以打开或关闭Arthas诊断功能,并设置生效IP。更多信息,请参见Arthas诊断(新版)。

⑦ 说明 仅应用监控专家版支持该功能。

Arthas监控		
Arthas开关	控制是否打开Arthas诊断功能,Agent 2.7.1.3 以上版本生效	
生效实时IF	 如果填写,Arthas诊断仅针对描定ip生效,如果未配置则对所有iP生效	

# 设置日志分析

在**日志分析配置**区域,可以打开或关闭日志分析功能,并设置日志采集级别和长度。更多信息,请参见日志 分析(旧版)。

日志分析配置					
日志框架自动采集		控制是否将日志	输出到 stdout		
采集日志级别	Debug	✓ Info	Varn	Error	
单条日志最大长度	512	单等	日志最大长度范围 0 ~ 2000	字符	

## 相关文档

- 调用链路查询
- 接口调用
- 使用线程剖析诊断代码层面的问题
- 内存快照

# 3.17.2. 标签管理

您可以在应用列表页面上管理应用标签,包括添加应用标签、通过标签筛选应用和删除应用标签。

### 添加标签

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏选择应用监控 > 应用列表,并在顶部菜单栏选择目标地域。
- 3. 在**应用列表**页面,将鼠标悬浮于目标应用右侧的 🕒 图标上,然后单击编辑标签。
- 4. 在编辑标签对话框可以通过以下方式为目标应用添加标签。
  - 单击绑定右侧的下拉框,选择已有的标签键值对添加为当前应用的标签。
  - 单击新建标签,设置新的标签键值对后,单击确定。
    - ⑦ 说明 单击标签右侧的×图标,可以为当前应用删除该标签。
- 5. 单击确定。

### 通过标签筛选应用

- 1. 在应用列表页面,单击标签下拉框。
- 2. 在标签下拉框选择需要查看的应用对应的标签键值对。

# 3.17.3. 监控方法自定义

除了被ARMS探针自动发现的所有方法和接口外,如果您需要监控应用中的其他方法和接口,则可在ARMS应 用监控中自定义监控方法。

# 前提条件

创建应用监控任务

# 功能入口

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏选择应用监控 > 应用列表,并在顶部菜单栏选择目标地域。
- 3. 在应用列表页面单击目标应用的名称。
- 4. 在左侧导航栏中单击应用设置,并在右侧单击监控方法自定义页签。

# 自定义监控方法

- 1. 在监控方法自定义页签,单击右上角的添加方法。
- 2. 在添加自定义方法对话框中,设置以下参数,并单击确认。

添加自定义方法			×
<ul> <li>1. 实时下发,2分钟</li> <li>2. 若未生效,请先</li> <li>钉服务号arms1608</li> </ul>	中左右生效。 检查是否已执行相应方法和方法是否挑 304并提供Agent日志。	并写正确。如果仍有问题,请联系钉	
* 方法全名:	例如: com.alibaba.arms.Demc	login	
开启此功能:	注意: 支持动态开启或关	铽,修改后无需重启应用。	
设为业务调用入口:	注意:若设置为入口,则会在接	]调用中生成相应统计数据。若 - = =	
	个设直,则仅任调用链的方法传	₩機示。	
		确认取	消
参数		描述	
方法全名		需要监控的方法的名称,具	有唯一性。

参数	描述
	开启此功能,即可监控此方法,并且可在本地方法栈 中展示此方法,更多信息,请参见 <mark>调用链路查询</mark> 。默 认为开启状态。
开启此功能	⑦ 说明 ARMS支持动态开启或关闭此功能, 并且无需重启应用。
设为业务调用入口	设置为业务调用入口后,则可通过调用链对其进行业 务查询,并且可在 <b>接口调用</b> 模块中展示对应接口,更 多信息,请参见 <mark>接口调用</mark> 。默认为关闭状态。

自定义监控方法添加成功后,自动显示在方法列表中。

### 相关文档

- 调用链路查询
- 接口调用

# 3.17.4. 删除应用

当您不需要再使用ARMS监控您的应用,并且需要在ARMS中删除您的应用时,可以在应用设置页面彻底删除。

### 操作步骤

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏选择应用监控 > 应用列表,并在顶部菜单栏选择目标地域。
- 3. 在应用列表页面单击目标应用的名称。
- 4. 在左侧导航栏中单击应用设置,并在右侧页面单击自定义配置页签。
- 5. 在自定义配置页签的Agent开关配置区域,关闭Agent总开关,然后单击保存。

↓ 注意 开关修改动态生效,无需重启应用。关闭此开关后,系统将无法监控您的应用,同时也
 不会产生费用,请您谨慎操作。

- 6. 卸载ARMS Agent,具体操作,请参见卸载Agent的常见问题。
- 7. Agent卸载完成后,在应用设置页面单击删除页签。

🚨 警告 此操作将会清除应用所有的监控数据,并且删除之后无法恢复。

		~			
Agent	自定义配置	标签管理	监控方法自定义	自定义参数	删除
删除应用					
此操作将会清 注意:删除之 删除应用	静险可用所有的监控数据 2.后无法恢复!	B.			

8. 在删除页签单击删除应用,在弹出的删除应用对话框选择删除原因,单击确定彻底删除应用。

删除应用	×
能否告知我们您为何离开ARMS?我们将竭力改进。	
* 删除原因: 功能不完善,无法定位到问题根本原因。	
Agent不稳定,影响应用运行。	
服务不稳定,经常打不开界面。	
✓ 报警相关问题	
价格太贵	
其他原因	
* 请单击以下链接查看详细的Agent卸载步骤。	
自动安装Agent: 卸载步骤	
手动 <del>安装</del> Agent: <mark>卸载步骤</mark>	
https://help.aliyun.com/document_detail/63797.html#title-w7y-kp7-9az	
请先卸载Agent或关闭该应用的Agent总开关,否则无法删除应用。 [立即前往]	
确定	以消

# 执行结果

返回**应用列**表页面,您可以查看到**应用列表**不再显示已删除的应用。 相关文档

• 应用监控常见问题

# 3.18. 应用监控告警规则(新版)

通过创建应用监控告警规则,您可以制定针对特定应用监控的告警规则。当告警规则被触发时,系统会以您 指定的通知方式向告警联系人或钉群发送告警信息,以提醒您采取必要的解决措施。

### 前提条件

已接入应用监控,请参见应用监控接入概述。

# 操作步骤

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏中选择**应用监控 > 应用监控告警规则**。
- 3. 在告警规则页面的右上角单击创建应用监控告警规则。
- 4. 在**创建应用监控告警规则**页面输入所有必填信息,完成后单击**保存**。

Home / 医管筋膜 / 細胞瘤 / 点所回照 创建应用监控告警规则 和例如编版用运行管理规则			
田田市● 2004 GC/2015			
	日前の時      「(cr-hangshou](stetgary service X (sr-hangshou]user-service X >      アチョン      アオポロ用が目的などの表面を引いていた     マチョン      アオポロの目的などの表面を引いていた     マチョン      マチャン      マチャン      マーン      マチャン      マチャン      マチャン      マチャン      マチャン      マチャン      マチャン      マーン      ローン      ローン      ローン      ローン		
田田敏政规则 (同时满足下述条件 ) 萬足下述任何一个条件 * 田田敏共和学生、新闻学生、学校研究中的一个条件			
NRI1 最近 5 分钟 JVM FullSC次数	マ 単約 マ 単約 マ 単上/時間に上作 マ 100 S マ マ Q X		
· 解选条件 五倍和₽ 遍历 ✓			
. There exists the transmission of the transmission $\mathcal{C}$ is the transmission of			
高校告告论量 ✔ 告告教编师订策略: ● ○ # 0 ○ # 1 ● # Nivul	(3)		
RAP DIA			
参数	说明		
告警名称	告警的名称。例如:JVM-GC次数同比告警。		
告警应用	选择需要设置告警的应用。可以选择多个应用。		
新建应用时自动在此告警规则中追 加	是否将之后接入的应用自动接入当前告警。		
指标类型	<ul> <li>选择监控指标的类型:</li> <li>JVM监控</li> <li>异常接口调用</li> <li>应用调用类型统计</li> <li>主机监控</li> <li>应用调用统计</li> <li>线程池监控</li> <li>数据库指标</li> </ul>		
	<ul> <li>⑦ 说明 不同的指标类型,告警规则的条件字段和筛选条件不同。</li> </ul>		
告警触发规则	<ul> <li>。同时满足下述规则:需满足所有告警条件才会触发告警。</li> <li>。满足下述一条规则:满足任意一条告警条件就会触发告警。</li> </ul>		

参数	说明
告警条件	<ul> <li>单击+添加条件,设置告警规则表达式。例如:最近5分钟JVM FullGC次数 平均与上小时同比上升100%。然后单击右侧 ✓ 图标。</li> <li>② 说明 <ul> <li>单击规则右侧 ✓ 图标,可以修改告警规则表达式。</li> <li>单击规则右侧 ④ 图标,可以预览应用在当前规则下的指标走 券图。</li> <li>单击规则右侧 ★ 图标,可以删除该告警规则表达式。</li> </ul> </li> <li>若需设置多条告警规则,单击+添加条件,即可编辑第二条 告警规则。</li> </ul>
筛选条件	告警指标的维度: • 无:告警内容中透出这个维度所有数值的和。 • =:告警中只透出当前设置维度的内容。 • 遍历:会在告警内容中透出实际触发告警的维度内容。
通知策略	<ul> <li>不指定通知规则:告警被触发时不会发送告警,仅当通知策略的分派规则被触发时才会发送告警。</li> <li>指定通知规则发送告警:告警被触发时,ARMS通过指定通知策略的通知方式发送告警信息。您可以选择已有的通知策略,也可以新建一个通知策略。更多信息,请参见通知策略。</li> <li>⑦ 说明 单击查看,可以查看选中的通知策略详情。</li> </ul>
高级告警设置	
告警数据修订策略	用于无数据、复合指标和环比同比等异常数据的修复。当告警指标没有达 到设置的条件时,告警数据补0、补1或不补充数据。 • 补零:将被判断的数值修复为0。 • 补一:将被判断的数值修复为1。 • 补Null:不会触发报警。 更多详细信息,请参见告警管理名词解释。

# 管理告警

创建的应用监控告警规则在**告警规则**页面上,您可以对告警规则执行启动、停止、编辑、删除、查看告警详 情等操作。

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏中选择应用监控 > 应用监控告警规则。
3. (可选)在告警规则页面的搜索框中输入告警名称,并单击搜索图标。

⑦ 说明 您可以输入告警名称的一部分内容进行模糊搜索。

4. 在搜索结果列表的操作列中,按需对目标告警规则采取以下操作:

	告蟄规则								
議論入規範条称 Q									
		报警答称	後型	所屬应用	报警规则	更新时间	状态	攝作	
		gateway-应用些控默认报题-数编框异常按题	默认应用监控报警	• gateway	最近5分钟数据库调用南应时间_ms平均值大于等于 2000  最近5分钟数据库调用错误次数平均值大于 等于1	2021-04-26 14:24:46	●运行中	编辑 停止 删除  报誓历史	
		gateway-应用並控默认报题-异常调用报题	默认应用监控报警	• gateway	最近5分钟调用调应时间_ms平均值大于每于2000   最近5分钟调用错误次数平均值大于每于1	2021-04-21 23:33:42	●运行中	编辑 停止 删除  报誓历史	
		gateway-应用蓝控默认报警-主机监控报警	默认应用监控报警	• gateway	最近5分钟节点机用户使用cpu_百分比平均值大于 等于901最近5分钟节点机空闲磁盘_byte平均值小	2021-03-26 15:51:16	●已停止	编辑 启动 删除  报誓历史	
		gateway-应用监控默认报答-进程异常报答	默认应用监控报警	• gateway	最近1分钟/vm_线程总数平均值与上小时同比下 降%50	2021-03-26 15:51:16	●已停止	编辑 启动 删除  报警历史	

- 如需编辑告警规则,请单击编辑,在编辑告警页面中编辑告警规则,并单击保存。
- 如需删除告警规则,请单击**删除**,并在提示对话框中单击确认。
- 如需启动已停止的告警规则,请单击启动,并在提示对话框中单击确认。
- 如需停止已启动的告警规则,请单击**停止**,并在提示对话框中单击确认。
- 如需查看告警事件历史和告警发送历史,请单击告警历史,在告警事件历史和告警发送历史页面上 查看相关记录。

### 相关文档

- 查看告警发送历史
- 查看告警事件历史

# 4.最佳实践

# 4.1. 使用调用链采样策略

本文介绍ARMS支持的多种调用链采样策略,帮助您根据自身场景选择合适的调用链采样策略,从而以较低 成本获取想要的调用链数据。

调用链采样适用于访问量比较高的大流量应用,能够帮助您以较低的成本和性能开销记录最有价值的链路数据。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。ARMS提供的调用链采样策略如下:

- 基于链路特征采样
- 基于业务特征采样
- 基于运维特征采样
- 基于时间特征采样

⑦ 说明 您可以组合使用以上调用链采样策略,以充分满足个性化的采样需求。

#### 基于链路特征采样

基于链路特征采样是指根据调用链本身的属性进行采样,例如耗时、状态等。ARMS支持按照固定比例采样 和线程剖析慢调用采样。

固定比例采样就是根据Traceld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。

基于固定比例采样适用于以下场景:

- 压测或大促峰值期间流量过大,为了避免调用链全量日志上报影响客户端性能,建议调低固定采样率比例 至1%~10%。
- 日常态全量调用链日志上报导致网络带宽成本高,可以考虑按需调整固定采样率的比例。

基于固定比例采样的基本原理如下:



设置固定比例采样的操作步骤如下:

1. 登录ARMS控制台。

2. 在左侧导航栏选择应用监控 > 应用列表,并在顶部菜单栏选择目标地域。

- 3. 在应用列表页面单击目标应用的名称。
- 4. 在左侧导航栏中单击应用设置,并在右侧页面单击自定义配置页签。
- 5. 在**调用链采样设置**区域,可以打开或关闭调用链采样开关,并设置采样率。**采样率设置**字段输入百分 比的数字部分即可,例如输入*10*代表采样10%。

↓ 注意 修改即时生效,无需重启应用。如果关闭采样,则调用链数据将不会被采集,请谨慎操作。

Agent	自定义配置	标签管理	监控方法自定义	自定义参数	删除	
调用链采样设置						
采样开启:		说明:关闭此配置,调	用链功能将关闭,请您谨慎	鼻作!		
* 采样率设置:		00	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,	请您谨慎操作!

线程剖析慢调用采样就是记录触发线程剖析慢调用监听的调用链数据。开启线程剖析后,ARMS在记录慢调用的同时会向下传递一个采样标记,并保留该次慢调用的下游链路。不过,为了避免影响客户端性能,线程 剖析监听数是有限的,如果有多个慢调用同时发生,仅会记录部分满足条件的慢调用及其下游调用链路。

线程剖析慢调用采样适用于以下场景:

- 系统出现偶发性慢调用,例如夜间服务响应耗时从0.5s突增到10s。如果提前打开线程剖析(默认触发阈 值为2s),ARMS会自动记录该次请求2s至10s之间的慢调用本地方法栈,及其下游调用链路。
- 秒杀或周期性大促峰值期间系统响应慢,线程剖析会记录满足触发阈值的慢调用本地方法栈,及其下游调用链路。



线程剖析慢调用采样的基本原理如下:

设置线程剖析慢调用采样的操作步骤如下:

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏选择应用监控 > 应用列表,并在顶部菜单栏选择目标地域。
- 3. 在应用列表页面单击目标应用的名称。
- 4. 在左侧导航栏中单击应用设置,并在右侧页面单击自定义配置页签。
- 5. 在**线程设置**区域,可以打开或关闭线程诊断方法栈开关、线程剖析总控开关,并设置慢调用监听触发阈 值。

线裡设置					
线程诊断方法线:		说明:开启后每隔5分钟采集一次方法线			
线程剖析总控开关:		开启后自动保存侵调用本地方法线			
慢调用监听触发阈值:	2000	(单位:ms) 耗时高于读调值才启动线程制析,建议设为耗时的9分位线。低于2000ms会增加CPU消耗,不可小于500ms。			

⑦ 说明 服务调用耗时超过慢调用监听触发阈值(默认值为1000毫秒)时才会启动监听,并一直持续到该次调用结束或超过15秒。建议将此阈值设为调用耗时的第99百分位数。假设有100次调用,则按耗时从小到大排序,排在第99位的耗时就是第99百分位数。

### 固定比例采样

### 线程剖析慢调用采样

### 基于业务特征采样

基于业务特征采样是指根据应用的业务流量特征进行采样。ARMS支持针对入口应用的HTTP流量特征进行规则配置。业务特征提取支持针对Header/Method/Cookie/Parameter信息进行过滤筛选,满足多场景业务特征匹配条件。开启全量开关后满足业务条件的调用链数据可以优先全量采集。配置后及时生效并支持运行期动态修改。

基于业务特征采样适用于以下场景:

- 业务人员需要监控所关注的业务,但已有的监控系统无法表达业务语义。
- 应用系统包含很多业务语义,运维人员需要快速配置并监控各个业务的流量数据。
- 新业务接口上线后,接口不稳定导致出现异常或错误,开发人员需要针对此接口的每个调用分析定位问题。
- 业务负责人需要梳理某个业务的依赖情况,并基于业务依赖进行相关业务优化,保障重点业务的稳定性。

基于业务特征的基本原理如下:



设置业务特征采样的操作步骤如下:

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏选择业务监控 > 应用接入,并在页面顶部选择目标地域。
- 3. 在应用接入页面的右上角单击创建业务监控。

4. 在新建业务监控页面的基础信息区域,设置以下信息,单击高级设置,在高级设置区域,设置以下信息,然后单击保存。

新建业务监控	(钉钉答疑群: 30004969)			
基础信息				
* 业务名称:	业务特征测试			
* 入口应用:	mall-center	~	注意: <b>支持Ag</b> e	nt <mark>版本2.6.2+</mark> ,修改此配置,将重置服务名称和规则参数。
* 服务类型:	● HTTP入口 ○ Kubernetes pod r	netadata	注意: 修改此配	置,将重置规则参数。
* 服务名称:	等于 🗸 /api/buy		说明: 支持手动	边修改服务名称。
过滤规则关系:	● 同时满足下述规则 ○ 满足下述	一条规则		
过滤规则:	Parameter V brand		== '	$\vee$
	Alibaba	Ô		
	十 添加规则			
分组规则:	十 添加规则			
◇ 高级设置				
向下游诱传·			说明· 沈岳标道	P是不在週田链由向下选应田节占沃佐
Dump业务参数:		说明: 采巴你记定否任调用进刊问『 <i>甜</i> 应用印度】。 说明: 是否将业务参数记录到带染色标记的调用链上。 说明: 是否对带染色标记的调用链并行全量采集。		
是否全量采集:				
	_			
保存 注意:保	存时将自动打开对应应用的业务监控开关。	。可以在此关闭		
参数		描述		示例
业务名称		业务监控任务的名称,必均	真。	业务特征测试。
		<b>入口应用</b> 列表显示所有已 监控探针的Java应用,必逆 所需应用后,ARMS自动检 用探针的版本号。	安装应用 选。选中 :测其应	
入口应用		<ul> <li>⑦ 说明 应用探针升级至</li> <li>2.6.2+版本才能使用业务监控功能,若检测到探针版本非</li> <li>2.6.2+时,请您先升级探针,</li> <li>具体操作,请参见升级探针。</li> </ul>		mall-center。
服务类型		设置服务类型,必选。ARI 仅支持 <b>HTTP入口</b> ,适用于 HTTP流量特征进行业务链 场景。	MS目前 F按 路染色	НТТР入口。

参数	描述	示例
服务名称	即应用提供的接口名称,必填。 ARMS会根据您设置的入口应用自 动匹配出该应用最近提供的接口列 表以便您选择。如果推荐的接口不 满足需求时,您也可以编辑修改。 服务名称的匹配模式支持以下4种 类型: • 等于:完全匹配服务名称的业 务接口,默认匹配模式。 • 开始等于:匹配以服务名称为 前缀的业务接口。若您需要监控 具备相同前缀的业务接口时,可 选择此模式。 • 包含:匹配包含服务名称的业 务接口。若您的应用提供大量业 务接口时,可选择此模式快速监 控您所需的业务接口。 。 结束等于:匹配以服务名称为 后缀的业务接口。 以".do"和".action"配置结 尾的典型Web框架比较适合使 用此模式。 • 模式匹配:匹配动态URI Path,支持Ant-style路径模式 匹配规则,可实现对某一类模式 的URI的监控及分析。	等于 /api/buy。
过滤规则关系	可以选择同 <b>时满足下述规则</b> 或满 足下述一条规则。	同时满足下述规则。
过滤规则	对配置的业务接口进一步筛选过 滤,选填。过滤规则需要设置匹配 参数 (Parameter、Cookie、Meth od、PathVariable和Header) 、匹配Key值、匹配方式 (==、!=和contains)和阈值, 其中,只有在服务名称选择模式 匹配时,且输入的字符串中包含占 位符大括号({}),则匹配参数会 出现PathVariable选项。支持设 置多条过滤规则,多条过滤规则之 间逻辑关系由您设置的过滤规则 关系决定。	假设您的应用提供 /api/buy? brand=*** 这样的URL, 您希望监 控brand=Alibaba的接口调用时, 可在表单中设置 <b>过滤规</b> 则为 <b>Parameter</b> brand == Alibaba。
是否全量采集	是否对带染色标记的调用链进行全 量采集。	是

## 基于运维特征采样

基于运维特征采样是指根据应用的运维特征进行采样,例如部署环境、网络环境等。ARMS支持根据应用的 Kubernetes Pod Metadata特征进行采样。ARMS支持针对入口应用Kubernetes Pod Metadata特征进行运 维特征规则配置,能够自动识别应用Pod实例元数据信息(如label/annotation/pod name/namespace 等)并进行过滤筛选,满足多场景运维特征匹配条件。开启全量开关后满足运维特征条件的调用链数据可以 优先全量采集。配置后及时生效并支持运行期动态修改。

基于运维特征采样适用于以下场景:

- 应用探针差异如语言、版本、环境变量、启动参数等信息需要针对性采样设置。
- 应用部署机器差异如规格、机型、区域、可用区等信息需要针对性采样设置。
- 应用部署环境差异如开发、测试、预发、生产隔离性需求需要针对性采样设置。
- 应用所处的网络环境如物理机、虚拟机、容器等部署形态差异或经典、VPC网络形式差异需要针对性采样 设置。

基于运维特征采样的基本原理如下:



设置运维特征采样的操作步骤如下:

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏选择业务监控 > 应用接入,并在页面顶部选择目标地域。
- 3. 在应用接入页面的右上角单击创建业务监控。
- 在新建业务监控页面的基础信息区域,设置以下信息,单击高级设置,在高级设置区域,设置以下信息,然后单击保存。

新建业务监控	(钉钉答疑群: 30004969)				
基础信息					
* 业务名称:	运维特征测试				
* 入口应用:	product-center	$\checkmark$	注意: <b>支持Ag</b> e	nt <mark>版本2.6.2+</mark> ,修改此配置,将重置服务名称和规则参数。	
* 服务类型:	○ HTTP入口 ● Kubernetes pod m	netadata	注意: 修改此酮	2置,将重置规则参数。	
过滤规则关系:		一条规则			
过滤规则:	podLabel V brand		==	~	
	Alibaba	<b>D</b>			
	十 添加规则				
分组规则:	十 添加规则				
∨ 高级设置					
向下游透传:			说明: 染色标i	己是否在调用链中向下游应用节点透传。	
Dump业务参数:			说明: 是否将	业务参数记录到带染色标记的调用链上。	
是否全量采集:			说明: 是否对帮	带染色标记的调用链进行全量采集。	
保存 注意:保存	<b>F时将自动打开对应应用的业务监控开关。</b>	可以在此关闭			
参数		描述		示例	
业务名称		业务监控任务的名称,必均	<b></b>	运维特征测试。	
		入口应用列表显示所有已会 监控探针的Java应用,必逆 所需应用后,ARMS自动检 用探针的版本号。	安装应用 匙。选中 测其应		
入口应用		<ul> <li>? 说明 应用探针升</li> <li>2.6.2+版本才能使用业务</li> <li>功能,若检测到探针版本</li> <li>2.6.2+时,请您先升级将</li> <li>具体操作,请参见升级将</li> </ul>	级至 务监控 \$非 释针 <i>,</i> <b>深针</b> 。	product-center。	
服务类型		设置服务类型,必选。ARM 仅支持Kubernetes Pod Metadata,适用于按照K 环境特征进行业务链路染色	MS目前 《8s Pod 色场景。	Kubernetes Pod Metadata。	
过滤规则关	系	可以选择同 <b>时满足下述规</b> 足下述一条规则。	则或满	同时满足下述规则。	

参数	描述	示例
过滤规则	对配置的业务接口进一步筛选过 滤,选填。过滤规则需要设置匹配 参数 (podLabel、podAnnotation 、podName、podNamespac e、podUID、podIp、nodeNa me、hostIp和podServiceAcc ount)、匹配方式 (==、!=和contains)和阈值。 支持设置多条过滤规则,多条过滤 规则之间逻辑关系由您设置的过滤 规则关系决定。	假设您的应用在 dev/test/stagging/prod等环境 均有部署,通过Pod label stge标 签进行区分,您希望监控dev环境 并对调用链全量采集,可在表单中 设置 <b>过滤规则为podLabel</b> stage == dev。
是否全量采集	是否对带染色标记的调用链进行全 量采集。	是

### 基于时间特征采样

基于时间特征采样是指根据应用的时间特征进行采样,例如在线诊断和离线分析。不同的诊断场景对于采样率的要求也不尽相同。ARMS支持在线诊断场景临时全量采集,方便您快速定位线上问题。开启实时诊断后,ARMS会持续监控应用5分钟,并在这5分钟内全量上报调用链数据。接下来,您就能以出现性能问题的调用链路为起点,通过方法栈瀑布图和线程剖析等功能定位问题原因。

基于时间特征采样适用于以下场景:

当您需要密切监控一小段时间内的应用性能时,例如发布应用或者对应用进行压测时。

基于时间特征采样的基本原理如下:



───── 开启实时诊断前,按固定比例采样

设置时间特征采样的操作步骤如下:

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏选择应用监控 > 应用列表,并在顶部菜单栏选择目标地域。
- 3. 在应用列表页面单击目标应用的名称。
- 4. 在左侧导航栏选择应用诊断 > 实时诊断。

首次进入**实时诊断**页面时,默认自动开启实时诊断。其他情况下,如需开启实时诊断,请单击页面右上 角的**开启实时诊断**。

实时诊断将于自动开启5分钟后自动终止。如需提前终止实时诊断,请单击页面右上角的**暂停定时刷 新**。

## 4.2. 使用线程剖析诊断代码层面的问题

ARMS线程剖析是代码级的诊断工具,能够自动捕获慢调用的堆栈快照,真实还原代码执行的第一现场。

### 使用场景

- 当促销活动出现慢调用时, ARMS线程剖析可为您快速定位问题代码。
- 当系统出现大量慢调用时, ARMS线程剖析可为您自动保存第一现场。
- 当业务太复杂, 偶发性慢调用无法复现时, ARMS线程剖析可为您还原代码真实执行轨迹。

#### 设置线程剖析参数

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏选择应用监控 > 应用列表,并在顶部菜单栏选择目标地域。
- 3. 在应用列表页面,单击目标应用名称。
- 4. 在左侧导航栏中单击应用设置,然后单击自定义配置页签。
- 5. 在线程设置区域,可以打开或关闭线程剖析总控开关,并设置慢调用监听触发阈值。
  - ? 说明
    - 服务调用耗时超过该阈值(默认值为2000ms)时才会启动监听,监听一直持续到此次调用 结束但不超过15s。
    - 建议将此阈值设为调用耗时的第99分位值。假设有100次调用,则按耗时从小到大排序,排 在第99位的耗时就是第99分位值。

线程设置	线程设置						
线程诊断方法栈:		说明:开启后每隔5分钟采集一次方法线					
线程剖析总控开关:		开启后自动保存侵调用本地方法找					
慢调用监听触发阈值:	2000	(单位:ms) 耗时高于该调值才启动线程剖析,建议设为耗时的99分位线。低于2000ms会增加CPU消耗,不可小于500ms。					

## 通过接口快照查看线程剖析详情

- 1. 在控制台左侧导航栏选择应用监控 > 应用列表,并在顶部菜单栏选择目标地域。
- 2. 在应用列表页面,单击目标应用的名称。
- 3. 在左侧导航栏中单击接口调用,并在页面右侧选择目标接口,然后单击接口快照页签。
- 4. 在接口快照页签上单击目标Traceld链接。 进入调用链路页签。
- 5. 在线程剖析列中单击放大镜图标。 弹出线程剖析对话框。

线程剖析			×			
应用名称: arms	服务名称:	IP地址:				
Traceld: ac	严生时间: 2020-11-10 19:17:3.325	耗时: 6548ms				
方法栈    线程剖析	方法线 线程制析					
机器IP:						
org.springframework.web.method.support.InvocableHan java.lang.reflect.Method.invoke():498 [4600ms] sun.reflect.DelegatingMethodAccessorImpl.invoke():43 [	dlerMethod.dolnvoke( ):205 [4600ms] 4600ms]					

? 说明

- 实际耗时是服务调用的实际执行时间,不受线程剖析影响。
- 。 监听耗时是能够被线程剖析监听到的耗时。通常情况下,监听耗时≈实际耗时-慢调用触发阈值。

### 通过调用链路查询查看线程剖析详情

- 1. 在控制台左侧导航栏中选择应用监控 > 调用链路查询。
- 2. 在调用链路查询页面的参数类型下拉列表中选择仅含线程剖析快照 / 并单击添加到查询条件。

调用链路查询			
仅含线程剖 ✓ true	添加到查询条件	常用查询条件	~
查询条件: 仅含线程剖析快照: 选 X			

- 在搜索结果中单击目标Traceld链接。
   进入调用链路页签。
- 在线程剖析列中单击放大镜图标。
   弹出线程剖析对话框。

### 常见问题

- 实际耗时是什么?
   答:实际耗时是服务调用的真实执行时间,不受线程剖析影响。
- 监听耗时是什么?
   答:监听耗时是指能够被线程剖析监听到的调用执行时间。为了尽量降低监听压力,线程剖析只会对每次调用超过慢调用监听触发阈值(默认为2s)后的执行时间进行监听。例如一次实际耗时为5s的慢调用,前2s不会监听,只监听3s~5s这一区间。如果一次调用耗时只有1.8s,那它将不会被监听。
- 为什么监听耗时小于实际耗时?甚至有些超过监听触发阈值的慢调用也没有被监听?
   答:
  - 由于线程剖析只监听一次调用超过触发阈值后的执行时间,因此,通常情况下,监听耗时≈实际耗时-慢调用监听触发阈值。
  - 如果系统在同一时间内出现大量慢调用,由于监听线程有限,无法保证每个慢调用在满足触发阈值的第一时间就被监听。此时,就可能出现监听耗时远小于实际耗时的情况,甚至不会被监听。
  - > 为了保证超慢调用被监听,线程剖析针对5s以上的超慢调用设置了独立的监听线程,因此会出现:监听 耗时≈实际耗时-5s。

# 4.3. 诊断服务端报错问题

网页抛错是互联网应用最常见的问题之一,但其错因分析是一个难点。为应用安装ARMS探针后,就能在不 改动应用代码的情况下,借助ARMS应用监控的异常自动捕捉、收集、统计和溯源等功能,准确定位应用中 所有异常并进行线上诊断。

### 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。

示例:常见的Java应用错误日志

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。

- 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。
- 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。
- 查看一个异常对应的Web请求和相关参数。
- 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。

### 解决方案

为应用安装ARMS探针后,即可在不改动应用代码的情况下,利用ARMS应用监控的异常自动捕捉、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ARMS探针

为应用安装ARMS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。具体操作,请参见应用监控接入概述。

#### 步骤二:查看关于应用异常的统计信息

为应用安装ARMS探针后,ARMS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. 登录ARMS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选择目标地域,然后单击目标应用名称。
- 在应用总览页面的概览分析页签下方,查看异常的总数、周同比和日同比数据。
   异常次数统计



4. 滑动页面至概览分析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。

异常类型	出现次数 /(占比)
org.springframework.web.util.NestedServletException at org.springframework.web.servlet.FrameworkServlet.processRequest(Framework Servlet.java:982) at org.springframework.web.servlet.FrameworkServlet.doGet(FrameworkServlet.java:861) Caused by: java.lang.NullPoint erException	53 / (22.18%)
java.lang.NullPointerException at com.alibaba.arms.demo.util.Util.sendGet(Util.java:108) at com.alibaba.arms.demo.TestComponentInvod eController.invokeComponent(TestComponentInvodeController.java:31)	47 / (19.67%)
java.net.UnknownHostException at java.net.InetAddress.getAllByNameO(inetAddress.java:1281) at java.net.InetAddress.getAllByName(ine tAddress.java:1193)	47 / (19.67%)
java.net.UnknownHostException at java.net.Inet4AddressImpl.lookupAllHostAddr(Native Method) at java.net.InetAddress\$2.lookupAllHo stAddr(InetAddress.java:929)	46 / (19.25%)

5. 在左侧导航栏,单击**应用详情**,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。



### 步骤三:诊断异常出现的原因

各类型异常出现的次数

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。ARMS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。

- 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。
- 2. 在调用链查询页签下,单击某个错误调用的Traceld。

#### > 文档版本: 20220713

⑦ 说明 如何查找目标调用链路,请参见调用链路查询。

接调用链查询页签

输入接口名称搜索	~					
产生时间 🖡	接口名称	所屬应用	耗时↓	状态↓	Traceld	操作
2022-06-29 11:31:35	/api/	arms	3.01min	•		查看日志
2022-06-29 11:33:20	/api/	arms	2min	•		查看日志
2022-06-29 11:37:16	/api/	arms	1.09min	•	1 To 10 TO 10 TO 10	查看日志
2022-06-29 11:23:31	/api/	arms	1.04min	•		查看日志
2022-06-29 11:30:16	/api/	arms	1.008min	•		查看日志
2022-06-29 11:36:34	/uplc	arms	58.9s	•		查看日志

3. 在弹出的页面,查看异常的调用链路信息,在**详情**列,单击放大镜图标,查看调用的方法栈,从而获得 异常的上下文信息。

#### 异常的完整调用链路信息

详情		×	
应用条称: am P地址: * 严生时间: 2022-06-29 1136:34.802 方法线 分析总流 描标运控		服务合称:/wp Traceld: 量費日志 相對: 58947ms	
调用方法	行号	扩展信息 时间镜 (ms)	
<ul> <li>Tomcat Servlet Process</li> </ul>		58947ms	
sunnet.www.protocol.http.HttpURLConnection.getInputStream()	1476	整数: http://osrearshanghai-inte 開電:juw.io.10Exception. Sarvat at rum.act.www.protool.l. 2 mms <	
org.apache.http.impl.client.CloseableHttpClient.execute(org.apac he.http.client.methods.HttpUriRequest request, org.apache.http.pr otocol.HttpContext context)	81	104ms	
org.apache.http.impl.conn.PoolingHttpClientConnectionMan ager.connectlorg.apache.http.HttpClientConnection managed Conn, org.apache.http.conn.routing.httpRoute route, int conn ectTimeout, org.apache.http.protocol.HttpContext context)	332	整數: prod-rowsense.oss-co-rshand (	
org.apache.http.protocol.HttpRequestExecutor.execute(org.a pache.http.HttpRequest request, org.apache.http.HttpClientCo nnection conn, org.apache.http.protocol.HttpContext context) Tage	118	数数: //litutwchqeM0558124+05433d/ (	
com alibaba.arms.console.service.impl.ArmsContextServiceImpl.cl ear()	79	Oms	

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。

### 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ARMS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。如何创建告警,请参见应用监控告警规则(新版)。

### 相关文档

- 创建应用监控任务
- 接口调用
- 调用链路查询
- 使用线程剖析诊断代码层面的问题
- 应用监控告警规则(新版)

# 4.4. 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,ARMS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
 业冬发展过修

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ARMS探针后,即可在不改动应用代码的情况下,使用ARMS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ARMS探针

为应用安装ARMS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。具体操作,请参见应用监控接入概述。

### 步骤二:查看慢SQL的统计信息

为应用安装ARMS探针后,ARMS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ARMS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选择目标地域,然后单击目标应用名称。
- 3. 在**应用总览**页面的概览分析页签下,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

ARMS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。

	~ a		最近15分钟 図
۲ (۱۹۵۵)	时间 / 请求数 / 错误数 / 异常数 三L	電流 SQL调用分析 NoSQL调用分析 异常分析 磁路上線 磁路下游 描层分析 调用磁面调	
清暗入	Q	1119116月 (2)	
•	l/sum 4.9s / 7 / 0 / 0	Y IIII	
	3.3s/2/0/0		
•	3.2s / 473 / 0 / 6		
·	1.8s/2/0/0		
•	1.6s / 4036 / 0 / 2	C Rope da	
•	1/list.js 1.3s / 8 / 0 / 0		
•	1.3s / 4787 / 0 / 3224		
	1.3s / 11.6K / 0 / 137		
•	byEBJ 1.3s / 4 / 0 / 0		
	1.08s / 398 / 0 / 0	0 Q Q C	
·	983ms / 2 / 0 / 0		
	976.6ms / 55 / 0 / 0	请求数 / 毎分仲 パー 単位 日 単 明 位 时间 / 毎分中 点 由 総 原電 約 (3) (5) 所	
	886.9ms / 47 / 0 / 0		
	on 728ms / 6 / 0 / 0		2
	676.2ms / 3616 / 0 / 0		
	594.2ms / 48 / 0 / 0	0 0 06-29 11:34 06-29 11:38 06-29 11:32 06-29 11:34 06-29 11:38	06-29 11:42
	10 542mc (14 / 0 / 0	• НТТРДД • НТТРДД	88

### 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全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。

- 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Traceld。
- 在弹出的页面,查看异常的调用链路信息,在方法栈列,单击放大镜图标,查看调用的方法栈,从而获 得异常的上下文信息。

⑦ 说明 如何查找目标调用链路,请参见调用链路查询。

在本示例中,可以看到在耗时为705毫秒的调用中,大部分的时间都消耗在了 SELECT * FROM 1_emplo yee 这次SQL调用中。

pro_ser_member (	-			×
调用方法	行号	扩展信息	时间轴(单位:毫秒)	
ProviderProcessor.handleRequest(com.taobao.hsf.invocation.Invocation invocatio	41	4f3989d0881		705
<ul> <li>EmployeeServiceImpl.queryEmpByEmpId(java.lang.String empId)</li> </ul>	2087			705
<ul> <li>MemEmployeeSmemServiceImpl.queryEmpByEmplds(java.util.List emplds)</li> </ul>	1238			705
<ul> <li>SqlSessionTemplate.selectList(java.lang.String statement, java.lang.Obj</li> </ul>	230			705
ConnectionImpl.prepareStatement(java.lang.String sql)	4015	参数: SELECT*	FROM 't_employee' WHERE	
PreparedStatement.execute()	1129	0#=1# AND emp	old IN (?)	705
				*2
				大肉

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ARMS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。如何创建告警,请参见应用监控告警规则(新版)。

## 相关文档

- 创建应用监控任务
- 接口调用

- 调用链路查询
- 使用线程剖析诊断代码层面的问题
- 应用监控告警规则(新版)

## 4.5. 通过诊断报告排查异常情况

定位、排查异常需要对多项指标逐一排查,过程漫长且复杂。针对此类问题,ARMS应用监控提供主动诊断 功能,帮助您快速准确地定位应用中各类异常,提供诊断报告,进而解决应用响应时间过长等问题。

### 步骤一:安装ARMS探针

为应用安装ARMS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。具体操作,请参见应用监控接入概述。

### 步骤二:查看诊断报告

为应用安装ARMS探针后,ARMS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等指标。ARMS支持查看包含所有排查指标的诊断报告,也可以通过**应用总览**页面查看单个指标的诊断报告。

### 查看包含所有指标的诊断报告

- 1. 登录ARMS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选择目标地域。

在**应用列表**页面,若应用存在异常,则**状态**列显示为红色。

3. 在**应用列表**页面上,将鼠标悬浮于目标应用所在行,出现检查到异常,单击查看详情的提示,单击红 点,加载诊断报告。加载完成后,将鼠标悬浮于红点上,单击诊断报告,进入诊断报告页面。

应用列表							资源消耗统	iit v	购买应用监控资源包	设置 🗸	接入应用
<b>提示:</b> 您已开通应用监控专家版,购买并(	更用相应地域的应用	用监控资源包最多可料	<del>9成本</del> 降低60%。立	即购买							<b>II</b> 🛞
标签: exception-maker sdsda								共11个应用	输入应用名称筛选应	用	Q
应用名称	标签②	健康度❷↓	请求数❷♪	错误数♪	响应时间小	异常数 ↓	状态♪ 最	近10分钟响应	团时间		操作 全局拓扑
	🖉	100.0%	7230	7230	5.2ms	14.5K	• .	$\checkmark$	$\sim$	$\varphi$	3D拓扑 设置
		100.0%	3614	0	0.2ms	0	诊断报告 应用异常,点击查君		$\sim \sim \sim$	$\mathcal{A}$	3D拓扑 设置
<u>چ</u>		78.2%	1478	1238	283.3ms	4952	• 7	$\sim$	~~~~	$\prec$	3D拓扑 设置

也可以在应用列表页面单击目标应用名称,进入应用总览页面,将鼠标悬浮于应用健康概览右侧的 <<br/>
图标,出现检查到异常,单击查看详情的提示,单击<br/>
图标,加载诊断报告。加载完成后,再次单击<br/>
<br/>
<

10100-000		
总请求量 1920	平均响应时间 5 2ms	错误数
周同比 ◆ 113.3% 日同比 ◆ 6.7%	周同比 ✿ 2.8% 日同比 ✿ 0.7%	周同比 ◆ 113.3% 日同比 ◆ 6.7%

在诊断报告页面查看诊断应用名称、诊断时间、故障现象、故障定界、根因分析和所有指标的检测结果。

诊断报告

RT诊断报告	©
诊断应用: nan 诊断时间: 10/10 14:06:00 ~ 10/10 15:06:00 故障调整: 应用	
故障定界 应用nan的主机的RT相较其他主机较高, RT高调用主要集中在调用服务	
com.上 根因分析	
<ul> <li>(1)可能的原因: 应用存在慢调用,关键转时方法栈为         <ul> <li>inner.business.diff.inpl.ComaodityDiffServiceInpl.diffPrice(</li></ul></li></ul>	diff.ConfigDiffRequest request, diff.ConfigDiffRequest gDiffRequest request, .io.Output output)

## 查看单个指标的诊断报告

- 1. 在应用列表页面单击目标应用名称,进入应用总览页面。
- 在应用总览页面各指标的曲线图上拖动鼠标选取目标时间段,单击查看所选时间段的诊断报告,进入详情面板。

系统信息		
CPU / 每分钟 点击曲线突增点	深度分析 (专家版可用)	<ul> <li>✓ □</li> <li>● CPU使用率</li> </ul>
16%	11-10 19:29 ~ 11-10 19:34	
12%	进入所选时间段	
8%	查看所选时间段的诊断报告	
4%	查看所选时间段的调用链路	
0%	查看所选时间段的日志	
11-10 19:24 11-10 19:	28 11-10 19:32	11-10 19:36

3. 在详情面板查看该指标所选时间段的检测结果。



## 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ARMS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。

创建告警操作步骤请参见应用监控告警规则(新版)。

## 相关文档

- 创建应用监控任务
- 接口调用

- 调用链路查询
- 使用线程剖析诊断代码层面的问题
- 应用监控告警规则(新版)

# 4.6. 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。

### 前提条件

⑦ 说明 仅应用监控专家版支持该功能。

您已将Agent版本升级至2.6.1.2及以上版本,具体操作,请参见更新Java探针版本。

### 背景信息

ARMS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。

### 业务日志关联调用链的Traceld信息

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏中选择应用监控 > 应用列表,在顶部菜单栏选择目标地域,然后在应用列表页面单击目标应用的名称。
- 3. 在左侧导航栏中单击应用设置,并在右侧单击自定义配置页签。
- 4. 在自定义配置页签的业务日志关联设置区域,打开关联业务日志与Traceld。

	业务日志关联设置 💡		
	关联业务日志与Traceld	开启后业务日志中会自动生成调用链的Traceld,此设置在重启应用后生效,支持	ilog4j/log4j2/logback日志组件,业务应用需要在日志的Layout中通过声明%X[EagleEye-TraceID]未输出TraceId信息。
	日志服务绑定⑦: 已授权,请绑定对应的Pro	oject及Logstore,并关联索引,支持指定字段索引及全文索引 区别。//,若未配置,请前	去日志服务控制台进行配置。
	* Project名称:	清选择	创建Project》
	* Logstore名称:	清选择 イン	
~	关联索引:	清选择	

? 说明

- 开启此开关后, 会在业务日志中自动生成调用链的Traceld。
- 如果您还需要绑定Project和Logstore,实现精准定位业务异常问题,具体操作,请参见通过调用链路和日志分析定位业务异常问题。
- 5. 在您业务日志的Layout的Pattern属性中添加 %X{EagleEye-TraceID} 配置。以Logback组件添加此配 置为例,如下图所示。

?	说明	如何在业务代码中获取{EagleEye-TraceID},请参见ARMS SDK使用说明。
<enco< th=""><th>der class charset&gt;U pattern&gt;% oder&gt;</th><th>="ch.qos.logback.classic.encoder.PatternLayoutEncoder"&gt; JTF-8 wd{yyyy-MM-dd HH:mm:ss.SSS} [%X{EagleEye-TraceID}] [%thread] %-5level %logger{50} - %msg%n</th></enco<>	der class charset>U pattern>% oder>	="ch.qos.logback.classic.encoder.PatternLayoutEncoder"> JTF-8 wd{yyyy-MM-dd HH:mm:ss.SSS} [%X{EagleEye-TraceID}] [%thread] %-5level %logger{50} - %msg%n

6. 重启应用。

在应用的业务日志中成功打印出Traceld信息,则说明业务日志关联调用链的Traceld关联成功,如下图 所示。

2020-03-17 22:41:48.418 [3041cae11584456 ] [http-bio-8008-exec-9] INFO - Loading XML bean definitions from class path resource [ 2020-03-17 22:41:48.438 [3041cae11584456 ] [http-bio-8008-exec-9] INFO - Loading XML bean definitions from class path resource [ 2020-03-17 22:41:48.449 [3041cae11584456 ] [http-bio-8008-exec-9] INFO - Loading XML bean definitions from class path resource [

# 4.7. 非阿里云环境下部署ARMS

本文介绍如何在非阿里云环境下将应用部署到ARMS。

非阿里云环境存在以下3种场景,请根据环境的实际情况选择部署ARMS的操作。

- 场景一:有专线连通到阿里云Region。
   等同于在对应Region下的正常部署,安装Java Agent的操作,请参见为Java应用手动安装Agent。
- 场景二:没有专线,机器环境可以开放部分公网。
   确认如下网络环境是否可以开放。
   ARMS端口:
  - 。 8442(元数据)
  - 。 8443 (统计数据)
  - 。 8883 (明细数据)
  - 8848 (微服务相关数据)
  - 。 9092 (应用诊断功能)

```
ARMS域名:
```

⑦ 说明 此处以北京地域为例,不同地域的网络域名不同,请确认需要接入的地域对应的网络域名 是否可以访问。各地域的网络域名,请参见各地域网络域名。

- 公网: arms-dc-bj.aliyuncs.com
- 内网(VPC网络): arms-dc-bj-internal.aliyuncs.com
- 。 公网: acm.aliyuncs.com

链路追踪端口:

- 80 (Jaeger HTTP)
- 1883 (Jaeger gRPC)

链路追踪域名:

⑦ 说明 此处以北京地域为例,不同地域的网络域名不同,请确认需要接入的地域下的网络域名是 否可以访问。各地域的网络域名,请参见各地域网络域名。

- 公网: tracing-analysis-dc-bj.aliyuncs.com
- 内网(VPC网络): tracing-analysis-dc-bj-internal.aliyuncs.com

```
各地域网络域名
```

地域	ARMS公网	ARMS内网(VPC网 络)	链路追踪公网	链路追踪内网 (VPC网络)
----	--------	-------------------	--------	-------------------

#### 应用实时监控服务 ARMS

地域	ARMS公网	ARMS内网(VPC网 络)	链路追踪公网	链路追踪内网 (VPC网络)
华东1(杭州)	arms-dc- hz.aliyuncs.com	arms-dc-hz- internal.aliyuncs.c om	tracing-dc- hz.aliyuncs.com	tracing-dc-hz- internal.aliyuncs.c om
华北2(北京)	arms-dc- bj.aliyuncs.com	arms-dc-bj- internal.aliyuncs.c om	tracing-dc- bj.aliyuncs.com	tracing-dc-bj- internal.aliyuncs.c om
华东2(上海)	arms-dc- sh.aliyuncs.com	arms-dc-sh- internal.aliyuncs.c om	tracing-dc- sh.aliyuncs.com	tracing-dc-sh- internal.aliyuncs.c om
华北1(青岛)	arms-dc- qd.aliyuncs.com	arms-dc-qd- internal.aliyuncs.c om	tracing-dc- qd.aliyuncs.com	tracing-dc-qd- internal.aliyuncs.c om
华南1(深圳)	arms-dc- sz.aliyuncs.com	arms-dc-sz- internal.aliyuncs.c om	tracing-dc- sz.aliyuncs.com	tracing-dc-sz- internal.aliyuncs.c om
华北3(张家口)	arms-dc- zb.aliyuncs.com	arms-dc-zb- internal.aliyuncs.c om	tracing-dc- zb.aliyuncs.com	tracing-dc-zb- internal.aliyuncs.c om
中国 (香港)	arms-dc- hk.aliyuncs.com	arms-dc-hk- internal.aliyuncs.c om	tracing-dc- hk.aliyuncs.com	tracing-dc-hk- internal.aliyuncs.c om
亚太东南(新加 坡)	arms-dc- sg.aliyuncs.com	arms-dc-sg- internal.aliyuncs.c om	tracing-dc- sg.aliyuncs.com	tracing-dc-sg- internal.aliyuncs.c om
政务云	arms-dc- gov.aliyuncs.com	arms-dc-gov- internal.aliyuncs.c om	tracing-dc- gov.aliyuncs.com	tracing-dc-gov- internal.aliyuncs.c om
华东1金融云(杭 州)	arms-dc-hz- finance.aliyuncs.c om	arms-dc-hz- finance- internal.aliyuncs.c om	tracing-dc-hz- finance.aliyuncs.c om	tracing-dc-hz- finance- internal.aliyuncs.c om

⑦ 说明 仅以上地域的网络环境支持通过此方式接入应用。

如果上述网络环境可以开放,参考以下文档安装Java Agent:

- 非Kubernetes集群下的应用:为Java应用手动安装Agent
- 。 Kubernetes集群下的应用:为开源Kubernetes环境中的应用安装探针
- ●场景三:没有专线,大部分机器环境不可以开放公网。
   在堡垒机上开放以上网络端口和对应Region的网络域名后,通过Gateway组件上传数据到ARMS应用监控中。具体操作,请参见在用户专有网络中部署ARMS。

# 4.8. 在用户专有网络中部署ARMS

本文介绍如何通过Gateway组件将用户网络中的应用接入公有云ARMS应用监控服务中。

### 使用场景

Gateway组件主要解决混合云场景下,由于网络不通原因,导致应用无法接入公有云ARMS应用监控服务问题,将其部署在阿里云环境(经典网络或VPC网络)充当代理角色。

### 部署原理

ARMS部署图



- 1. 部署在DMZ中的ARMS Gateway集群A对阿里云专有云和用户网络暴露内网地址,部署在专有云中的 ARMS探针和WebLogic上的ARMS探针把采集的监控数据发送到Gateway上。
- 2. DMZ通过专线和阿里云公有云打通,部署在公有云VPC内的ARMS Gateway集群B通过专线连接Gateway 集群A,起桥接作用。
- 3. ARMS Gateway集群B把收集到的ARMS监控数据发送到公有云服务端。
- 4. 阿里云公有云上的ARMS探针,包括部署在页面上的前端探针,将数据发送到ARMS公有云服务端。

### ARMS混合云监控部署要求

- 按照采集专有云和用户网络500个节点来规划,需要6台规格为2核8 G的虚拟机,部署2个包含3台 Gateway组件的集群。
- Gateway向阿里公有云传输的监控数据量比较小,可以控制在1 M/s内,所以利用现有的专线带宽,专有云EDAS和WebLogic都可以通过安装ARMS探针开启应用监控,无需修改代码。

## 安装ARMS

- 1. 下载Gateway软件包,此处以杭州地域为例。
- 2. 将Gateway部署在代理机器上,通过如下命令启动。

java -jar arms-gateway-1.7.0.jar

⑦ 说明 JDK版本需要为JDK 1.7+。

### Gateway默认连接杭州地域的ARMS服务,如需连接其它地域,可通过-D参数修改地域域名。连接到北 京地域的示例命令如下:

java -jar -Darms.server.endpoint=arms-dc-bj.aliyuncs.com arms-gateway-1.7.0.jar

地域	域名
华东1(杭州)	arms-dc-hz.aliyuncs.com
华东2(上海)	arms-dc-sh.aliyuncs.com
华北1 (青岛)	arms-dc-qd.aliyuncs.com
华北2(北京)	arms-dc-bj.aliyuncs.com
华北3(张家口)	arms-dc-zb.aliyuncs.com
华北5(呼和浩特)	dc-cn-huhehaote.arms.aliyuncs.com
华北6(乌兰察布)	dc-cn-wulanchabu.arms.aliyuncs.com
华南1(深圳)	arms-dc-sz.aliyuncs.com
华南2(河源)	dc-cn-heyuan.arms.aliyuncs.com
华南3(广州)	dc-cn-guangzhou.arms.aliyuncs.com
西南1(成都)	dc-cn-chengdu.arms.aliyuncs.com
中国(香港)	arms-dc-hk.aliyuncs.com
亚太东南1(新加坡)	arms-dc-sg.aliyuncs.com
亚太东南2(悉尼)	dc-ap-southeast-2.arms.aliyuncs.com
亚太东南3(吉隆坡)	dc-ap-southeast-3.arms.aliyuncs.com
亚太东南5(雅加达)	arms-dc-indonesia.aliyuncs.com
亚太东北1(东京)	arms-dc-jp.aliyuncs.com
欧洲中部1(法兰克福)	arms-dc-frankfurt.aliyuncs.com
欧洲西部1(伦敦)	dc-eu-west-1.arms.aliyuncs.com
美国东部1(弗吉尼亚)	dc-us-east-1.arms.aliyuncs.com
美国西部1(硅谷)	arms-dc-usw.aliyuncs.com
亚太南部1(孟买)	dc-ap-south-1.arms.aliyuncs.com
政务云	arms-dc-gov.aliyuncs.com
杭州金融云	arms-dc-hz-finance.aliyuncs.com

地域	域名
上海金融云	arms-dc-sh-finance.aliyuncs.com
深圳金融云	arms-dc-sz-finance.aliyuncs.com

3. 下载Agent。

可以在ARMS控制台的接入中心页面下载,具体操作,请参见为Java应用手动安装Agent。

- 4. 解压Agent安装包。
- 5. 在arms-agent.config文件中,修改profiler.collector.ip指定的服务地址为代理机器地址。

```
profiler.collector.ip={代理机器IP}
```

 启动应用,在ARMS控制台的应用监控 > 应用列表页面查看监控数据是否正常上报,如果有数据上报, 则说明接入成功。

# 4.9. 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ARMS应用监控的性能瓶颈。ARMS应用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。

### 前提条件

- 已开通日志服务SLS。登录日志服务控制台时,根据页面提示开通日志服务。
- 已创建Project,详情请参见创建Project。
- 已创建Logstore, 详情请参见创建Logstore。

#### 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。

### 关联业务日志与Traceld

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏中选择应用监控 > 应用列表,在顶部菜单栏选择目标地域,然后在应用列表页面单击目标应用的名称。
- 3. 在左侧导航栏中单击应用设置,并在右侧单击自定义配置页签。
- 4. 在自定义配置页签的业务日志关联设置区域,打开关联业务日志与Traceld开关,然后绑定Project和 Logstore。

应用日志关联配置 ?				
选择日志源	) 默认日志 🖲 日志	服务		
关联业务日志与Traceld		开启后业务日志中会目动生成调用链的Traceld。此设置在重启应用后生效	、支持	Log4j/Log4j2/Logback日志组件。业务应用需要在日志的Layout中通过声明%X(EagleEye-TraceID)未输出TraceId信息。
日志服务绑定⑦:	已授权,请绑定对应的Pro	oject及Logstore,并关联索引,支持指定字段索引及全文索引 区别分,若未配置	1. 清前:	去日志服务控制台进行配置。
	*区域:		~	
	* Project名称:		~	创建Project。
	* Logstore名称:		~	创建Logstore
	关联索引:	全文素引	~	创建关联责引。

5. 在自定义配置页签左下角单击保存。

### 从应用指标的角度排查业务异常问题

- 1. 登录ARMS控制台。
- 在左侧导航栏中选择应用监控 > 应用列表,在顶部菜单栏选择目标地域,然后在应用列表页面单击目标应用的名称。
- 在左侧导航栏单击应用总览,在顶部选择概览分析,然后在右上角选择或自定义设置目标时间段。
   概览分析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。
- 在概览分析页面选择某个应用指标,在该指标的曲线图上拖动鼠标选取目标时间段。
   本示例以应用提供服务平均响应时间指标为例。

☰ (-) 阿里云	Q 建变文档、控制台、API、解决方案和资源 费用 工单 备款企业支持 官网 🖸 🗘 🏹 🔞 体 🍥
<	应用提供服务⑦
应用总览	应用提供服务请求量 / 每分钟 点击曲线突增点深度分析 (专家版可 ビ 🖬 🕷 应用提供服务平均响应时间 / 每分钟 点击曲线突增 ビ 分位数 🗸 🗷 🛸
应用详情	● HTTP入口 ● 自定义方法         点源處分析 (专家版可用)         ● HTTP入口 ● 自定义方法
接口调用	48K     3s     09-17 11:35 - 09-17 11:37       36K     2.3s     进入所送时间段
事件详情	24K 1.5s 查看所选时间段的诊断报告
数据库调用	12K 查看所选时间段的调用链路
NoSQL调用 <new< th=""><th>0 09-17 11:24 09-17 11:32 09-17 11:40 09-17 11:48 09-17 11:24 09-17 11:32 09-17 11:40 09-17 11:48</th></new<>	0 09-17 11:24 09-17 11:32 09-17 11:40 09-17 11:48 09-17 11:24 09-17 11:32 09-17 11:40 09-17 11:48
外部调用 <new< th=""><th></th></new<>	
MQ监控	应用依赖服务③
应用诊断 ン	
应用设置	

- 5. 查看步骤4所选时间段的调用链路。
  - i. 单击查看所选时间段的调用链路。

ii. 在调用链路列表面板选择**状态**异常(显示为)的调用链路记录,单击该调用链路记录**Traceld**列下 的Traceld值。●

您也可以在该调用链路记录**操作**列下单击**查看日志**,查看该时间点的业务日志,分析业务异常原因。

☰ (-) 阿里云		Q 搜索文档、控制台	、API、解决方案和资源	费用 工单	备案 1	È业 支持	官网 🖸	Ų. Ä	⑦ 简体	0
<	应用提供服务⑦	<b>論 \ 按口名数</b> 迪泰		0						<b>^</b>
应用总览		48/12010101238		4						
	应用提供服务请求里	产生时间 🕆	接口名称	所属应用	耗时 小	状态 🕇	Traceld		操作	
应用详情 接口调用	用) 48K ————————————————————————————————————	2020-09-17 11:35:30	/api/version.json	$\sum_{i=1}^{n-1} (1-i) = (1-i) + \frac{1}{n}$	79ms	•			查看日志	
事件详情	36K 24K	2020-09-17 11:35:10	/api/version.json	uru analy M	86ms	•	and the second s	5.75897.00	查看日志	
数据库调用	12K	2020-09-17 11:35:18	/registerPromCluster	arra-straite N	566ms	•	2****	onen-pro-	查看日志	
NoSQL调用《KKW	09-17 11:24 09-	2020-09-17 11:35:23	/registerPromCluster	$_{\rm N}^{\rm constant}$	648ms	•	and the second		查看日志	
MQ脑控	应用依赖服务⑦	2020-09-17 11:35:28	7	$\frac{1}{2} \sum_{i=1}^{n} \frac{1}{i} \sum_{i=1}^{n} \frac{1}$	1ms	•	10, 100, 100 100	1.11.1	查看日志	
应用诊断 中田20番	应用依赖服务请求星 • [:]	2020-09-17 11:35:07	/api/version.json		97ms	•		100.00	查看日志	
型用设直	9						05-1667016003	10700014745440		- <b>•</b>

- ⅲ. 单击调用链路页签, 然后在方法栈列下单击图标。 <del>Q</del>
- iv. 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。

	eservice prometheus PrometheusServicel mpLaddExporter(javaJang String regionid, javaJang String use rid, javaJang String cluterid, javaJang.httpare zoorterTyre, ja	986	异葉: java.lag.DuntineException : 8艘grafana文件夹头数, C 「 felderHene: pass	488ms	支持	a 网	2	0. A	0	简体	0
调用链路 业务轨迹	vaJang.String config, javaJang.Boolean isRefresh)		. core. service. prometheur. grafana. Grafanal.o nfi dervice. refrethViesDobreldGrafanal.onfi dervice. java.2								
CRON	pectj Jang Proceeding JoinPoint joinPoint)	22	23) 4t	Zms							
	<ul> <li>org.mybatis.spring.SqlSessionTemplate.selectOne(java.l ang.String statement, java.lang.Object parameter)</li> </ul>	167	service, prosethour, prafama. Grafamalo nfigService, installBaceBBaBhry(GrafamalonfigService, java-11 1)	2ms							
	<ul> <li>til die initial pool DruidDataSource.getConne ction()</li> </ul>	984	at 🤉 orra zarri ca nomatham orafana. Grafanam 🔻	1ms							
	com.mysql.jdbc.StatementImpl.executeQuery(ja va.lang.String.sql)	1474	(SS): SELECT D#+1#	1ms							

- 6. 查看步骤4所选时间段的业务日志。
  - i. 单击查看所选时间段的日志。
  - ii. 在日志分析页面选择异常错误信息,查看日志并定位业务异常原因。

⊟ (−) 阿里云				Q 提家文档、拉约台、API、解决方面和资源	裁用	I# %	<b>s</b> ≙₩	支持	官网	D.	٥.	9 R	简体	0
<	in the second se	∼ a												
应用他选 应用详辑	已绑定Project:	Logstore:	and and in the 编辑Z											
按口调用	𝔍 arms-console-all-log			© 2020-09-09 14:06:00-2020-09	9-09 14:15:00	- ÉS	1月新 :	<b>享</b> 6	查询分析版	B性	另存为协	速查询	另存为常	10
事件详情	✓ 1 0bc0594d159963178		ribert / krivka kaap / unappilari / skithetina i liseri n' 12 k ki / an i kinada kun a' i ava jann									ି 🕜	查8/5	9 <b>9</b> 5
設掘岸调用 NoSQL週用 4550		3 😡 09-09 14:10:07	Coert Scooler and Antonio Coerto, Tocholar Coerto, Fight Reng. Source: tag: hostname: tag: path. tag: path. tag. tag: path. tag. tag. tag. tag. tag. tag. tag. tag	The second second										
外部调用 4000			topic	nio-8080-exec-1] WARN c.a.a.c.service.prometheus.P	PrometheusS	erviceImpl - j	rometheus (	check inp	ut param su	iccess fr	rom			
<u>▲</u> 应用沙断 ~		4 🔯 09-09 14:10:07	source: _taghostname: _tagpath:											
应用设置	<		Listicin exervise and the second seco	ttb-rio-8880-exec-11 INFO arms statistic registerProm images and the statistic registerProm forderfame: prefershVereich/solf/contractors/Service jave 223 e installBaseKB1Emr(GrafeneConfgService jave 2111) e created statistic profer/Dashboard (GrafeneConfgService jave 2111) e created statistic Service images and statistic statistics and statistics a	Cluster logge 9b4aar III	er - [Registeri	PromCluster Id: claef73c5	3ervlet) R 18e3e4a7	RegisterPror 7964aa0402	nCluster 38	Serviet Fa	iled. → pr	.xvyUserid:	

## 从接口调用的角度排查业务异常问题

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏中选择**应用监控 > 应用列表**,在顶部菜单栏选择目标地域,然后在**应用列表**页面单击目 标应用的名称。
- 3. 在左侧导航栏单击接口调用。
- 4. 在接口调用页面的接口列表区域单击目标接口,然后单击右侧的接口快照页签。
- 5. 在接口快照页签选择状态异常的接口调用记录, 异常状态显示为。

☰ (-) 阿里云				Q 提家文档、控制台、API、I	解决方面和资源 <b>数</b> 目	目 工単 智嘉 企	业 支持 官网	ы ф <b>.</b>	₩ @	) 简体	0
<									半天		
应用他流											
应用详情	⑦ 响应时间 / 请求数 / 错误数 / 异军数 副	概范 数据库调用分析 NoSQLiji	1用分析 异常分析 错误	具分析 链路上游 <mark>1888</mark> 链路下	游 1997 接口快照						
接口调用	iji航入 Q	输入接口名称搜索	Q								
事件详情	• comapi.TraceAction.doDiagnosis 7.01s / 99 / 0 / 0	产生时间小	接口名称	所墨应用	耗时4 状态4	Traceld				操作	
数据库调用	/containerOperation     6.9s / 41 / 0 / 39	2020-09-17 10:02:33	/upload.json	managements for	7.0s					2288 (	đ
NoSQL调用 4997	• /upload.json	2020-09-17 09:53:04	/upload.json	100.000	6.7s	10000				<b>主</b> 著日:	8
外部週用 4000	• /api/result.json 1.9s / 3 / 0 / 0	2020-09-17 10:12:48	/upload.json	100 100 100	1.5s	-				查看日 <b>2</b>	*
MQ追控	• /api/query.json 1.7s / 126.4K / 0 / 81.3K	2020-09-17 10:05:46	/upload.json	and the second sec	1.06s	<b>NAME</b>				查看日7	<del>а</del>
应用诊断 〜	• /api/task.json 1.2s / 233 / 0 / 0				-						
应用设置	• /api/solution.json 805.1ms / 99 / 0 / 0										

- 6. 查看接口调用的调用链路。
  - i. 在目标接口调用记录的Traceld列下单击Traceld的值。
  - ii. 单击调用链路页签, 然后在方法栈列下单击图标。
  - iii. 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。

Ξ 6	-) 阿里云	reservice prometheus.PrometheusServicel mpLaddExporter(java.lang.String regionid, java.lang.String use ol java.lang.Strine durated java.lang.totage serverter(java.jav	986	异菜: java.lang.DuntineEverption: 8艘grafam文件夹失败, 0 个 folderHew: pass	488ms	支持	官同	5.	¢. ¥	0	简体	0
100 100.00	11-02.45.00	valang.String.config. javalang.Boolean isRefresh)		at								
· · · · · · · · · · · · · · · · · · ·	32793002	ole.aspect.DaoAspectaround(org.as pectj.lang.Proceeding.loinPoint joinPoint)	22	nti gësrvice, retreshvicebjërgid (GratanaConti gësrvice, jave/2 23) 4t	Zms							
-		<ul> <li>org.mybatis.spring.SqlSessionTemplate.selectOnc(java.l ang.String statement, java.lang.Object parameter)</li> </ul>	167	nfigervice.installBaseBBeRev(GrafmaLonfigErvice.jsva:11 1)	2ms							
		tion()	984	at y cora carri ca nomatham prafana Brafanam 🔻	1ms							
		com.mysql.jdbc.StatementImpl.executeQuery(ja va.lang.String.sql)	1474	御助: SHLBCT O#+1#	1ms							

- 7. 查看接口调用的日志。
  - i. 在目标接口调用记录的操作列下单击查看日志。
  - ii. 在日志分析页面选择异常错误信息,查看日志并定位业务异常原因。

■ (-) 阿里云				Q 接张文档、控制台、API、解决方面和资源	裁用	工単	音震	企业 支	18 官冈	D.	٥.	Э. Ц	3) ne	et 🌔
<	100.0000	× a												
应用他该	and the state of t													
应用详情	已绑定Project:	Logstore:	● 「 」 「 」 「 」 「 」 「 」 「 」 「 」 」 「 」 」 「 」 」 「 」 」 」 「 」 」 」 「 」 」 」 」 」 」 」 」 」 」 」 」 」 」 」 」 」 」 」 」											
接口调用	𝔍 arms-console-all-log			© 2020-09-09 14:06:00-2020-	09-09 14:153	00 🕶	自动刷新	分享	查询分析	宿住	另存为	快速查询	另存	为告誓
事件详情	✓ 1 0bc0594d1599631788		moorr (ruke teas/unassilari/skm+ms_licerin" 1X t+li/s1(nin-sahuh, e' isvs isnn h	uironten-vontion								0	) 🖆	刷/分析
数据库调用		3 🔯 09-09 14:10:07	source: taghostname:	an one Loopoon										
NoSQL调用 4550			fegpath: _fegreceive_time:											
外部调用 - 1887			content	io-8080-exec-1] WARN  c.a.a.c.service.prometheus	.Prometheus	ServiceIm	pl - prome	theus check	input param	success	from			
MQ监控		4 🔯 09-09 14:10:07	SOURCE											
应用诊断 、														
应用设置			tag_:receive_time:											
	< c		Ocontent: 2020-09-09 14 10 07 057 (bcc/99401599531788) 01 PM 109837003 ) userid 1330240 cm accessive sectors functionoptions null, ret: Jwa Jang Runtime Exception @Bargaranz/HERER, ID at service promethicus gratina Gratina Configence at service promethicus gratina Gratina Configence at service promethicus gratina Gratina Configence	p=nio-8080-exec-11 INFO arms statistic registerPro a, regionid: cn-beijing, clusterid: cfser/3c98e344 (entrame: 2) refreshWewBy Orgid(CartanaConfigService, java 22) instalBaseKBsEmr(GrafanaConfigService, java 21) restleGrafanaExporterDashboard(GrafanaConfigService)	mCluster.log a79b4aar 3) ervice.java:8:	ger - [Regi: 	sterPromC pilotid: cfa	lusterServie ef73c98e3e	t) RegisterPr 4a79b4aa04	omCluste 02a8	erServiet F	'ailed> p	proxyUser	ld:

# 4.10. 将ARMS页面嵌入自建Web应用

如需在自建Web应用中免登录查看ARMS控制台的页面,您可将ARMS控制台嵌入自建Web应用,以此避免系统间的来回切换。

### 背景信息

将ARMS控制台嵌入自建Web应用可以实现以下效果:

- 可登录自有系统并浏览嵌入的ARMS控制台的应用列表、应用详情、调用查询等页面。
- 可隐藏ARMS控制台的顶部菜单栏和左侧导航栏。具体操作,请参见隐藏导航栏。
- 可通过访问控制RAM控制操作权限,例如将完整权限改为只读权限。具体操作,请参见借助RAM用户实现 分权。

### 示例代码

如需快速体验将ARMS页面嵌入自建Web应用的效果,您可以下载并使用示例代码。

## 访问流程

外部系统访问ARMS的流程如下图所示。



### 步骤一: 创建RAM用户并添加权限

使用阿里云账号创建RAM用户并为其添加调用STS服务扮演RAM角色的权限。

- 1. 登录RAM控制台。
- 2. 在左侧导航栏,选择人员管理 > 用户。
- 3. 在用户页面,单击创建用户。
- 在创建用户页面的用户账号信息区域,输入登录名称和显示名称,在访问方式区域,选中编程访问,然后单击确定。

↓ 注意 RAM会自动为RAM用户创建AccessKey(API访问密钥)。出于安全考虑, RAM控制台只 提供一次查看或下载AccessKeySecret的机会,即创建AccessKey时,因此请务必将 AccessKeySecret记录到安全的地方。

- 5. 在手机验证对话框,单击获取验证码,输入收到的手机验证码,然后单击确定。
- 6. 在用户页面,找到创建好的用户,单击操作列的添加权限。
- 在添加权限面板的选择权限区域,通过关键字搜索需要添加的权限策 略AliyunSTSAssumeRoleAccess,并单击权限策略将其添加至右侧的已选择列表,然后单击确定。

添加权限					
被授权主体					
系统权限策略 > Aliyu	nSTS	8	Q	已选择(1)	清除
权限策略名称	备注			AliyunSTSAssumeRoleAccess	×
AliyunSTSAssumeRoleAccess	调用STS服务AssumeRole接口的权限				

8. 在添加权限的授权结果页面,查看授权信息摘要,然后单击完成。

### 步骤二: 创建RAM角色并添加权限

创建RAM角色并为其添加访问控制台的权限。RAM用户将会扮演该RAM角色访问控制台。

- 1. 登录RAM控制台。
- 2. 在左侧导航栏,单击RAM角色管理。
- 3. 在RAM角色管理页面,单击创建RAM角色。
- 4. 在创建RAM角色面板,执行以下操作:
  - i. 在选择类型页面的当前可信实体类型区域,选择阿里云账号,然后单击下一步。
  - ii. 在配置角色页面的角色名称文本框,输入RAM角色名称,然后单击完成。
  - iii. 在创建完成页面, 单击为角色授权。
- 5. 在**添加权限**面板的选择权限区域,搜索需要添加的权限策略,单击权限策略将其添加至右侧的已选 择列表,然后单击确定。

您可以根据需要为RAM角色赋予以下ARMS权限:

- 。 AliyunARMSFullAccess: ARMS的完整权限。
- AliyunARMSReadOnlyAccess: ARMS的只读权限。
- 6. 在添加权限的授权结果页面,查看授权信息摘要,然后单击完成。

### 步骤三: 获取临时AccessKey和Token

登录自建Web,然后在Web服务端调用STS的AssumeRole接口获取临时AccessKey和Token,即临时身份。 AssumeRole接口详情,请参见AssumeRole。

您可以选择以下方式调用AssumeRole接口:

- 通过OpenAPI开发者门户调用。
- 通过Java SDK调用。

本文以通过Java SDK调用为例。

您在使用Java SDK时需要注意填写以下参数:

```
String accessKey = "<accessKeyId>"; //RAM用户的AccessKeyId。
String accessSecret = "<accessKeySecret>"; //RAM用户的AccessKeySecret。
String roleArn = "<roleArn>"; //RAM角色的标识ARN。
```

RAM用户的AccessKeyId和AccessKeySecret为创建RAM用户时获取。

获取RAM角色的roleArn的操作步骤如下:

- 1. 登录RAM控制台。
- 2. 在左侧导航栏,单击RAM角色管理。
- 3. 在RAM角色管理页面下方的RAM角色列表,单击目标RAM角色名称。
- 4. 在RAM角色详情页面的基本信息区域,复制ARN信息。

RAM访问控制 / RAM角色	管理 / admin-aliwaredoc		
← admin-al	iwaredoc		
基本信息			
RAM角色名称	admin-aliwaredoc	创建时间	2019年3月7日 14:24:57
备注		ARN	acs:ram::1428

## 步骤四: 获取登录Token

通过STS的AssumeRole接口获取临时AccessKey和Token后,调用登录服务接口获取登录Token。

↓ 注意 STS返回的安全Token中可能包含特殊字符,请对特殊字符进行URL编码后再输入。

请求样例:

```
http://signin.aliyun.com/federation?Action=GetSigninToken
&AccessKeyId=<STS返回的临时AccessKeyId>
&AccessKeySecret=<STS返回的临时AccessKeySecret>
&SecurityToken=<STS返回的安全Token>
&TicketType=mini
```

### 步骤五: 生成免登录链接

利用获取到的登录Token与待嵌入的ARMS控制台页面链接生成免登录访问链接,以最终实现在自建Web中 免登录访问ARMS控制台页面的目的。

⑦ 说明 由于Token有效期为3小时,建议在自建Web应用中将链接设置为每次请求时生成新登录 Token。

1. 在ARMS控制台获取待嵌入的控制台页面的URL链接。

例如上海地域应用页面的URL链接为:

https://arms.console.aliyun.com/apm?iframeMode=true&pid=\${pid}&regionId=\${regionId}#/
\${pid}/home

? 说明

- URL链接必须为ARMS应用监控或前端监控的控制台地址,链接中的 {pid} 和 {regionId}
   请从实际URL中获取。
- 在URL链接的search部分添加 iframeMode=true ,可以隐藏ARMS控制台原有的顶部菜单 栏和左侧导航栏。
- 在URL链接的search部分添加 hideTopbar=true 可以隐藏顶部菜单栏,添加 hideSidebar
   =true 可以隐藏左侧导航栏,同时使用等同于添加 iframeMode=true 。
- 2. 利用登录Token与ARMS控制台页面链接生成免登录访问链接。

```
http://signin.aliyun.com/federation?Action=Login
&LoginUrl=<登录失效跳转的地址,一般配置为自建Web配置302跳转的URL>
&Destination=<ARMS控制台页面链接>
&SigninToken=<获取到的登录Token>
```

3. 在浏览器访问该免登录访问链接。

# 4.11. 使用ARMS监控异步任务

对于通过Spring @Async标签实现的异步任务,ARMS默认支持对其进行监控。此外,ARMS还支持通过添加 异步透传扫描包和使用ARMS SDK进行手动透传对自定义异步任务实现监控。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。

### 前提条件

该功能要求ARMS探针版本为公测版本v2.7.1.3及以上。请联系ARMS钉钉服务账号arms160804,为您进行探 针新版本升级。

## 方式一:默认支持Spring @Async标签

ARMS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, ARMS会自动完成增强:

- Executor:
  - org.springframework.scheduling.concurrent.ConcurrentTaskExecutor
  - o org.springframework.core.task.SimpleAsyncTaskExecutor
  - $\circ \ org. spring framework. scheduling. quart z. Simple Thread Pool Task Executor \\$
  - o org.springframework.core.task.support.TaskExecutorAdapter
  - org.springframework.scheduling.concurrent.ThreadPoolTaskExecutor
  - org.springframework.scheduling.concurrent.ThreadPoolTaskScheduler
  - org.springframework.jca.work.WorkManagerTaskExecutor
  - o org.springframework.scheduling.commonj.WorkManagerTaskExecutor
- Task:
  - org.springframework.aop.interceptor.AsyncExecutionInterceptor\$1
  - org.springframework.aop.interceptor.AsyncExecutionInterceptor\$\$Lambda\$

## 方式二:添加异步透传扫描包

您可以选择在应用设置中添加异步透传扫描包实现异步任务监控。异步透传扫描包中的Runnable、Callable和Supplier接口在创建新对象时会自动捕获当前线程调用链的上下文,并在异步线程中执行时使用该调用链上下文,完成串联。

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏,选择应用监控 > 应用列表。
- 3. 在顶部菜单栏,选择地域。
- 4. 在应用列表页面,单击应用名称。
- 5. 在左侧导航栏, 单击**应用设置**, 然后单击**自定义配置**页签。
- 6. 在高级设置区域的异步透传扫描包名对话框中,添加异步透传扫描包。

异步透传扫描包名:	com.alibaba.arms.brightroar.console.service	注意:应用重启生效。

#### 7. 在页面底部, 单击保存。

 ↓ 注意 配置后需重启应用才能使功能生效。如需添加多个异步透传扫描包,可以使用半角逗号(,) 分隔。

## 例如,对于通过以下示例代码创建的异步任务,您可以使用添加异步透传扫描包的方式对此异步任务进行监控。此示例代码中,异步透传包名为com.alibaba.arms.bright roar.console.service。

◆ 注意 在配置时,您可以按需调整异步透传包名的范围。若需监控的异步任务过多,您可以缩小异 步透传包名的范围。在本示例中,除了输入完整的异步透传包名
 com.alibaba.arms.brightroar.console.service以外,您还可以输入更短的前缀包名com.alibaba.arms来
 自动扫描此目录下的所有子透传包。但需注意的是,若前缀包名范围过大,会对性能产生影响,请谨慎 操作。

```
package com.alibaba.arms.brightroar.console.service;
@Service
public class NameService {
    private ExecutorService es = Executors.newFixedThreadPool(5);
    public void name() {
        es.submit(new Runnable() {
           @Override
           public void run() {
            System.out.println(System.currentTimeMillis()+ ": my name is john, " + Thre
        ad.currentThread().getId());
        }
    });
    });
  }
}
```

### 方式三: 使用ARMS SDK手动透传

当您的使用方式比较复杂,上述场景无法满足需求时,您还可以选择使用ARMS SDK进行手动透传。通过手动增强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. 在Maven项目的pom.xml中添加以下依赖。

<dependency>
 <groupId>com.alibaba.arms.apm</groupId>
 <version>1.7.5</version>
 <artifactId>arms-sdk</artifactId>
 </dependency>

2. 增强Runnable接口、Callable接口和Executor。您可以按需选择以下任意一种增强方式。

```
○ 方式一:增强Runnable接口和Callable接口
```

```
使用 TraceRunnable.asyncEntry() 增强Runnable接口,使用 TraceCallable.asyncEntry() 增强Callable接口。示例代码如下:
```

```
public class AsyncEntryExample {
    private final ExecutorService executor = Executors.newSingleThreadExecutor();
    @GetMapping(value = "/sdk-async-plugin/asyncEntry-propagation")
    public String asyncEntryAndExecute() throws Exception {
        CompletableFuture<String> future = new CompletableFuture<>();
        Runnable command = TraceRunnable.asyncEntry(() -> future.complete("asyncEntry
-execute"));
        executor.execute(command);
        Thread.sleep(1000);
        return future.get();
    }
}
```

### ○ 方式二: 增强Executor

```
使用 TraceExecutors.wrapExecutorService(executr, true) 增强Executor。示例代码如下:
```

方式三:同时增强Runnable接口、Callable接口和Executor
 示例代码如下:



## 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,请参见调用链路查询。

调用链路 业务日志	详情					×	
应用名称 ARMS-	应用を称: ARMS - ■ ■			服务名称: /u Traceld: 1e 耗时: 3005	服务名称:/user/show Traceid: tee ■ ■ ■ ■ ■ ■ 耗时: 3006ms		
	调用方法 ▼ Asynchronous Invocation	行号	扩展信息		时间轴 (ms) 3005ms		
	org.springframework.aop.interceptor.AsyncExecutionInterce     ptor54Lambda5878.call()         com.alibaba.arms.brightroar.console.service.UserService         e.say()	11			3004ms		

# 4.12. 通过TraceExplorer实时分析链路数 据

本文将介绍如何通过链路分析快速定位五种经典线上问题,更直观的了解链路分析的用法与价值。

### 背景信息

除了使用调用链排查单次请求的异常,或者使用预聚合的链路统计指标进行服务监控与告警之外,链路追踪 还支持基于明细链路数据的后聚合分析,简称链路分析(Trace Explorer)。相比调用链,链路分析能够更 快的定界问题;相比预聚合的监控图表,链路分析可以更灵活的实现自定义诊断。

链路分析是基于已存储的全量链路明细数据,自由组合筛选条件与聚合维度进行实时分析,可以满足不同场 景的自定义诊断需求。例如,查看耗时大于3秒的慢调用时序分布,查看错误请求在不同机器上的分布,或 者查看VIP客户的流量变化等。

### 问题一:流量不均

负载均衡配置错误,导致大量请求打到少量机器,造成"热点"影响服务可用性,怎么办?

流量不均导致的"热点击穿"问题,很容易造成服务不可用。在生产环境中出现过多起这样的案例,比如因 负载均衡配置错误,注册中心异常导致重启节点的服务无法上线,DHT 哈希因子异常等。

流量不均的最大风险在于能否及时发现"热点"现象。它的问题表象更多是服务响应变慢或报错,传统的监控无法直观的反映热点现象,所以大部分运维人员都不会第一时间考虑这个因素,从而浪费了宝贵的应急处理时间,造成故障影响面不断扩散。

通过链路分析按IP分组统计链路数据,可以直观地看到调用请求分布在哪些机器上,特别是问题发生前后的 流量分布变化。如果大量请求突然集中在一台或少量机器,很可能是流量不均导致的热点问题,然后再结合 问题发生点的变更事件,快速定位造成故障的错误变更,及时回滚。

在Trace Explorer页面设置按IP聚合,如下图所示,可以发现大部分流量集中在XX.XX.XX.108这台机器上。



### 问题二: 单机故障

网卡损坏、CPU超卖、磁盘打满等单机故障,导致部分请求失败或超时,如何排查?

单机故障每时每刻都在频繁发生,特别是核心集群由于节点数量比较多,从统计概率来看几乎是一种"必然"事件。单机故障不会造成服务大面积不可用,但是会造成少量的用户请求失败或超时,持续影响用户体验和答疑成本,需要及时处理。

单机故障可以分为宿主机故障和容器故障两类(在Kubernetes环境可以分为Node和Pod)。例如CPU超卖、 硬件故障等都是宿主机级别,会影响所有容器;而磁盘打满、内存溢出等故障仅影响单个容器。因此,在排 查单机故障时,可以根据宿主机IP和容器IP两个维度分别进行分析。

面对这类问题,可以通过链路分析先筛选出异常或超时请求,然后再根据宿主机IP或容器IP进行聚合分析, 可以快速判断是否存在单机故障。如果异常请求集中在单台机器,可以尝试替换机器进行快速恢复,或者排 查该机器的各项系统参数:例如磁盘空间是否已满、CPU Steal Time是否过高等。如果异常请求分散在多台 机器,那么大概率可以排除单机故障因素,可以重点分析下游依赖服务或程序逻辑是否异常。
在Trace Explorer页面筛选错误调用或慢调用,并设置按IP进行分组统计,如果异常调用集中出现在特定机器,则有较大概率是机器故障。



# 问题三: 慢接口治理

新应用上线或大促前性能优化,如何快速梳理慢接口列表,解决性能瓶颈?

新应用上线或大促备战时通常需要做一次系统性的性能调优。第一步就是分析当前系统存在哪些性能瓶颈, 梳理出慢接口的列表和出现频率。

此时,可以通过链路分析筛选出耗时大于一定阈值的调用,再根据接口名称进行分组统计,这样就可以快速 定位慢接口的列表与规律,然后对出现频率最高的慢接口逐一进行治理。

找到慢接口后,可以结合相关的调用链、方法栈和线程池等数据定位慢调用根因。常见原因包括以下几类:

- 数据库或微服务连接池过小,大量请求处于获取连接状态。可以调大连接池最大线程数解决。
- N+1问题。例如一次外部请求内部调用了上百次的数据库调用,可以将碎片化的请求进行合并,降低网络 传输耗时。
- 单次请求数据过大,导致网络传输和反序列化时间过长,而且容易导致Full GC。可以将全量查询改为分页 查询,避免一次请求过多数据。
- 日志框架"热锁"。可以将日志同步输出改为异步输出。



#### 在Trace Explorer页面筛选大于5秒的慢调用,并设置按接口名进行分组统计,发现慢接口的规律。

### 问题四:业务流量统计

如何分析重保客户或渠道的流量变化和服务质量?

在实际生产环境中,服务通常是标准的,但业务却是分类分级的。同样的订单服务,我们需要按照类目、渠道、用户等维度进行分类统计,实现精细化运营。例如,对于线下零售渠道而言,每一笔订单、每一个POS 机的稳定性都可能会触发舆情,线下渠道的SLA要求要远高于线上渠道。那么,应该如何在通用的电商服务 体系中,精准的监控线下零售链路的流量状态和服务质量呢?

这里可以使用链路分析的自定义Attributes过滤和统计实现低成本的业务链路分析。例如,在入口服务针对 线下订单打上 {"attributes.channel": "offline"} 的标签,然后再针对不同门店、用户客群和商品类 目分别打标。最后,通过对 attributes.channel = offline 进行过滤,再对不同的业务标签进行group by来分组统计调用次数、耗时或错误率等指标,就可以快速地分析出每一类业务场景的流量趋势与服务质 量。

#### 问题五:灰度发布监控

500台机器分10批发布,如何在第一批灰度发布后,就能快速判断是否有异常?

变更三板斧"可灰度、可监控、可回滚"是保障线上稳定性的重要准则。其中,分批次灰度变更是降低线上风险,控制爆炸半径的关键手段。一旦发现灰度批次的服务状态异常,应及时进行回滚,而不是继续发布。 然而,生产环境很多故障的发生都是由于缺乏有效的灰度监控导致的。

例如,当微服务注册中心异常时,重启发布的机器无法进行服务注册上线。由于缺乏灰度监控,前几批重启 机器虽然全部注册失败,导致所有流量都集中路由到最后一批机器,但是应用监控的总体流量和耗时没有显 著变化,直至最后一批机器也重启注册失败后,整个应用进入完全不可用状态,最终导致了严重的线上故 障。

在上述案例中,如果使用 {"attributes.version": "v1.0.x"} 对不同机器流量进行版本打标,通过链路 分析对 attributes.version 进行分组统计,可以清晰的区分发布前后或不同版本的流量变化和服务质 量,不会出现灰度批次异常被全局监控掩盖的情况。

#### 链路分析的约束限制

链路分析虽然使用灵活,可以满足不同场景的自定义诊断需求,但是它也有几点使用约束限制:

• 基于链路明细数据进行分析的成本较高。

链路分析的前提是尽可能完整的上报并存储链路明细数据。如果采样率比较低导致明细数据不全,链路分析的效果就会大打折扣。为了降低全量存储成本,可以在用户集群内部署边缘数据节点,进行临时数据缓存与处理,降低跨网络上报开销。或者,在服务端进行冷热数据分离存储,热存储进行全量链路分析,冷存储进行错慢链路诊断。

- 后聚合分析的查询性能开销大,并发小,不适合用于告警。
   链路分析是实时的进行全量数据扫描与统计,查询性能开销要远大于预聚合统计指标,所以不适合进行高并发的告警查询。需要结合自定义指标功能将后聚合分析语句下推至客户端进行自定义指标统计,以便支持告警与大盘定制。
- 结合自定义标签埋点,才能最大化释放链路分析价值。
   链路分析不同于标准的应用监控预聚合指标,很多自定义场景的标签需要用户手动埋点打标,这样才能最有效的区分不同业务场景,实现精准分析。

# 4.13. 通过OpenTelemetry Java SDK进行 手工埋点

本文主要介绍如何通过OpenTelemetry开源SDK手动埋点监控您的应用。

### 前提条件

该功能要求ARMS探针版本为公测版本v2.7.1.3及以上。请联系ARMS钉钉服务账号 arms160804 ,为您进行 探针新版本升级。

#### 背景信息

云原生背景下,由OpenTracing演进而来的OpenTelemetry提供了多达数十种语言的SDK,基于一套规范的 API和统一的数据格式,成为可观测的事实标准。

ARMS Java Agent从版本v2.7.1.3开始,内置对OpenTelemetry Java SDK的支持。如果您的应用已经通过 OpenTelemetry Java SDK手动埋点,则无需任何改动,ARMS会将相关的Span记录作为独立入口或者内部方 法栈,从而监控您的应用。如果您的应用尚未埋点,请先通过依赖OpenTelemetry Java SDK,实现手动埋 点。

### 手动埋点

如果您的应用尚未埋点,请先通过引入以下Maven依赖,实现手动埋点。更多信息,请参见OpenTelemetry 官方文档。

<dependencies></dependencies>
<dependency></dependency>
<proupid>io.opentelemetry</proupid>
<pre><artifactid>opentelemetry-api</artifactid></pre>
<dependency></dependency>
<groupid>io.opentelemetry</groupid>
<pre><artifactid>opentelemetry-sdk-trace</artifactid></pre>
<dependency></dependency>
<proupid>io.opentelemetry</proupid>
<pre><artifactid>opentelemetry-sdk</artifactid></pre>
<dependency></dependency>
<proupid>io.opentelemetry</proupid>
<pre><artifactid>opentelemetry-exporter-logging</artifactid></pre>
<dependencymanagement></dependencymanagement>
<dependencies></dependencies>
<dependency></dependency>
<groupid>io.opentelemetry</groupid>
<pre><artifactid>opentelemetry-bom</artifactid></pre>
<version>1.2.0</version>
<type>pom</type>
<scope>import</scope>

# ARMS对OpenTelemetry埋点的兼容

ARMS对OpenTelemetry埋点的兼容的介绍涉及以下名词, OpenTelemetry相关的其他名称解释, 请参见OpenTelemetry Specification。

- Span: 一次请求的一个具体操作,比如远程调用入口或者内部方法调用。
- SpanContext: 一次请求追踪的上下文, 用于关联该次请求下的具体操作。
- Attribute: Span的附加属性字段,用于记录关键信息。

OpenTelemetry的Span可以分为三类:

• 入口Span: 会创建新的SpanContext, 例如Server、Consumer。

⑦ 说明 对于此类Span, ARMS的埋点入口多位于框架内部,手动埋点时链路上下文已存在, ARMS会将OpenTelemetry的入口Span作为内部Span处理。对于ARMS没有埋点的入口,则 OpenTelemetry的入口Span保持不变,例如异步调用或者自定义RPC框架入口,同时ARMS会在客户端聚合,生成相关统计数据。

- 内部Span: 会复用已经创建的SpanContext, 作为内部方法栈记录。
- 出口Span: 会将SpanContext透传下去, 例如Client、Producer。

目前,ARMS对于入口Span和内部Span做了兼容。对于出口Span,ARMS暂不支持按照OpenTelemetry标准 透传,而是按照ARMS自定义格式透传。

#### 例如以下代码:

```
package com.alibaba.arms.brightroar.console.controller;
import io.opentelemetry.api.OpenTelemetry;
import io.opentelemetry.api.common.AttributeKey;
import io.opentelemetry.api.common.Attributes;
import io.opentelemetry.api.trace.Span;
import io.opentelemetry.api.trace.SpanKind;
import io.opentelemetry.api.trace.StatusCode;
import io.opentelemetry.api.trace.Tracer;
import io.opentelemetry.api.trace.propagation.W3CTraceContextPropagator;
import io.opentelemetry.context.Scope;
import io.opentelemetry.context.propagation.ContextPropagators;
import io.opentelemetry.exporter.logging.LoggingSpanExporter;
import io.opentelemetry.sdk.OpenTelemetrySdk;
import io.opentelemetry.sdk.trace.SdkTracerProvider;
import io.opentelemetry.sdk.trace.export.BatchSpanProcessor;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.ResponseBody;
import org.springframework.web.bind.annotation.RestController;
import javax.annotation.PostConstruct;
import java.util.concurrent.Executors;
import java.util.concurrent.ScheduledExecutorService;
import java.util.concurrent.TimeUnit;
@RestController
@RequestMapping("/ot")
public class OpenTelemetryController {
   private Tracer tracer;
    private ScheduledExecutorService ses = Executors.newSingleThreadScheduledExecutor();
    @PostConstruct
   public void init() {
        SdkTracerProvider sdkTracerProvider = SdkTracerProvider.builder()
                .addSpanProcessor(BatchSpanProcessor.builder(new LoggingSpanExporter()).bui
ld())
                .build();
        OpenTelemetry openTelemetry = OpenTelemetrySdk.builder()
                .setTracerProvider(sdkTracerProvider)
                .setPropagators (ContextPropagators.create (W3CTraceContextPropagator.getInst
ance()))
                .buildAndRegisterGlobal();
        tracer = openTelemetry.getTracer("manual-sdk", "1.0.0");
        ses.scheduleAtFixedRate(new Runnable() {
            QOverride
            public void run() {
                Span span = tracer.spanBuilder("schedule")
                        .setAttribute("schedule.time", System.currentTimeMillis())
                        .startSpan();
                try (Scope scope = span.makeCurrent()) {
                    System.out.println("scheduled!");
                    Thread.sleep(500L);
                    span.setAttribute("schedule.success", true);
                } catch (Throwable t) {
                    span.setStatus(StatusCode.ERROR, t.getMessage());
                } finally {
```

```
span.end();
                }
           }
       }, 10, 30, TimeUnit.SECONDS);
   }
   @ResponseBody
   @RequestMapping("/parent")
   public String parent() {
       Span span = tracer.spanBuilder("parent").setSpanKind(SpanKind.SERVER).startSpan();
       try (Scope scope = span.makeCurrent()) {
           child();
           span.setAttribute("http.method", "GET");
           span.setAttribute("http.uri", "/parent");
        } finally {
           span.end();
        }
       return "parent";
    }
   private void child() {
       Span span = tracer.spanBuilder("child").startSpan();
       try (Scope scope = span.makeCurrent()) {
           span.addEvent("Sleep Start");
           Thread.sleep(1000);
           Attributes attr = Attributes.of(AttributeKey.longKey("cost"), 1000L);
           span.addEvent("Sleep End", attr);
        } catch (Throwable e) {
           span.setStatus(StatusCode.ERROR, e.getMessage());
        } finally {
           span.end();
       }
   }
}
```

以上示例代码中通过OpenTelemetry SDK创建了三个Span:

- parent : 按照OpenTelemetry标准是HTTP入口,但是由于ARMS在Tomcat内置代码中已经创建了链路 上下文,因此这里会作为一个内部方法记录在ARMS方法栈上。
- child : parent Span的内部Span,作为内部方法记录在方法栈上。
- schedule : 独立线程入口Span, 默认情况下ARMS没有为此类入口创建上下文, 因此这里会作为一个 自定义方法入口, 并生成相应的统计数据。

#### 在ARMS控制台查看 parent 和 child

在ARMS控制台找到/ot/parent的HTTP入口的内部方法栈,可以看到多了以下Span的展示。更多信息,请 参见<mark>调用链路查询</mark>。

#### 应用监控·最佳实践

详情				
原用名称: ARMS-Brightroar IP地址: 产生封局: 2021-06-08 17:08:6.122 方法技長 线程副所 分析总宽 指标监控			服务名称: /d Traceld: 1e 耗时: 1006r	V(parent 11) In Hand Diff. Dollar Tophological Teles ns
握示:如果方法栈不足以定位代码问题,请继续宣看线程创析结果。				
调用方法	行号	扩展信息		时间轴 (ms)
▼ Tomcat Servlet Process				1006ms
org.apache.catalina.core.StandardHostValve.invoke(org.apa ▼ che.catalina.connector.Request request, org.apache.catalina. connector.Response response)	111			1005ms
org.springframework.web.servlet.FrameworkServlet.doG vet(javax.servlet.http:HttpServletRequest request, javax.ser vlet.http:HttpServletResponse response)	898			1005ms
com.alibaba.arms.brightroar.console.controller.OpenT elemetryController.parent()	69			1003ms
v parent Tags				1003ms
child				1003ms

#### 在ARMS控制台查看 schedule

ARMS控制台支持通过以下几个页面查看:

• 可以在ARMS控制台应用总览页面看到自定义入口的统计。更多信息,请参见应用总览。

应用提供服务②						
> 自用提供服务请求量 / 每分钟 点面面出来地点采用分布		イビ 回 省 • HTTPAD • 日定文方法	应用器供服务平均购应时间	/ #99 01000000000000000000000000000000000		<ul> <li>ビ SEE V 図業</li> <li>HTTPAD ・目前の方法</li> </ul>
3 225 15	$\wedge$	/	800ms 800ms 800ms		$\wedge$	
0.25 0.05 06-06 15:58 06-06 17:52	56-08 17:05 D6-08 17:	10	01-08 16:58	06-08 17:52	06-08 12:56	05-08 17:10

• 可以在接口调用页面查看Span详细信息。更多信息,请参见接口调用。

0	响应时间 / 请求数 / 错误数 / 异常数 🗐	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 🚾 计	周用链查询		
请输入	Q										\$
• /ot/parent	1.04s / 2 / 0 / 0										
schedule	512.5ms / 2 / 0 / 0							0	_		
• /error	60ms / 1 / 1 / 0										
								8.00%/Jat U	SER_METHOD		
								平均 51	2.50ms		
								AP	)		
											RESET
		请求	遗 / 每分钟					~⊻⊠ ≹	响应时间 / 每分钟 点击曲线突增点深度分	Ť	∠⊾⊇*
		2						/	600ms		1
		1							300ms		/
		0.5						_/	150ms		
		0 06-0	8 16:58	06-08 17:02	06-08	8 17:06	06-08 17:10	/	0ms	06-08 17:06	06-08 17:10
					<ul> <li>自定义方</li> </ul>	法				<ul> <li>自定义方法</li> </ul>	
		慢调	用次数 / 每分钟					2 🗠 🖂 🕷	HTTP-状态码统计		
		4						/	2		
		2						_/	1		
		1									
		06-0	8 16:58	06-08 17:02	06-08	17:06	06-08 17:10		0 -06-08 16:59 06-08 17:03	06-08 17:07	06-08 17:11
					自命() 也)	±			•	100 • 2VV • 2VV • 2VV • EVV	

● 可以在**调用链路**页面看到独立的入口。更多信息,请参见<mark>调用链路查询</mark>。

 Allowa
 Diamatrix
 Ris mark
 Allowa
 Pall
 Holds
 Pall
 Holds
 Holds

单击**详情**列的放大镜图标,可以查看入口的详细信息。

详情			×
<ul> <li>2)用名称: ARM5-Brightroar</li> <li>P地址:</li> <li>P地址前回: 2021-06-08 17:13:20.559</li> <li>方法线 分析总流 指标监控</li> </ul>		服 Tra 耗	客容許: schedule kceld: 时: 521ms
调用方法	行号	扩展信息	时间输 (ms)
OpenTelemetry Entry Span			521ms
schedule Tags			503ms

目前ARMS支持将OpenTelemetry Span的Attribute作为Tags展示,将鼠标悬浮于Span名称后面的Tags即可看到Span的Attribute。

详情						×
应用名称: ASMS-Srightnoar 服务名称: IP地址: Traceld:: I 产生时间: 2021-06-08 17:1320.559 转时: 521				服务名称: s Traceld: 耗时: 521m	s	
方法栈 分析总览	指标监控					
调用方法			行号	扩展信息		时间轴 (ms)
OpenTelemetry Entry	Taos					521ms
schedule Tags	schedule.success schedule.time	true 1623143600536				503ms

# 4.14. 在阿里云Prometheus监控下查看应 用监控大盘

阿里云Prometheus默认集成了ARMS应用监控数据源,您可以直接在Prometheus监控下查看应用监控大盘。

### 前提条件

已开通应用监控和Prometheus监控。

## 功能入口

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏选择Prometheus监控 > Prometheus实例列表。
- 3. 在**Promet heus监控**页面的顶部菜单栏,选择应用监控所在的地域。 在**Promet heus监控**页面,实例类型为promet heus for 应用监控的实例即为默认集成的应用监控。

Prometheus监控			资源消耗统计 > 新建prometheus实例			
遗示,纪己开遗专家乐。了解众意识则						
使用制作,使用AAKS Promethanu出现的常意世纪NM、MySQL Go、Redis情应用影响中,并以大量展示监控发展。						
实例备称	实例接到	grafana folder	攝作			
	Prometheus for 智器服务	进入grafana folder	1972   9P45			
arms_metrics_cn-hangzhou_cloud_hangzhou	prometheus for 应用监控	进入grafana folder	设置   卸稿			
final point providence in angles	Prometheus for 云服筒	进入grafana folder	1239   \$PAS			

4. 单击应用监控实例名称,可以查看预置大盘并获取Remote Read地址、Remote Write地址、HTTP API 地址和Token信息。

## 查看预置大盘

在大盘列表页面可以查看应用监控预置大盘。大盘内各指标含义,请参见应用监控指标说明。

# 获取Remote Read地址等信息

在设置页面获取Remote Read地址、Remote Write地址、HTTP API地址和Token信息。

5.参考信息

# 5.1. ARMS Java探针性能压测报告

本文为ARMS Java探针的性能压测报告。在本报告中,测试人员对同一个应用以不同的压测流量进行了多轮压测,以此来验证ARMS Java探针的性能。

# 背景信息 压测环境基本信息

- ARMS Java探针版本: v2.7.3
- 压测平台: 阿里云性能测试PTS
- 测试应用所在ECS实例的硬件配置: 2 Intel(R) Xeon(R) Platinum 8369B CPU @ 2.70GHz, 4G memory
- 操作系统版本: Cent OS Linux release 7.6.1810 (Core)

# 测试应用基本信息

该测试应用基于Spring,包含Spring Boot、Spring MVC,模拟的Redis客户端和JDBC连接池(用于模拟 MySQL客户端)。在监控该应用的过程中,针对每个事务,ARMS Java探针会抓取7个Span,包括1个 Tomcat,1个Spring MVC,1个Method,2个Jedis和2个MySQL。

下载测试应用。

# 基线指标

基线是测试应用在不安装探针的情况下进行压测得到的基准指标。指标包括:

- CPU: CPU总体百分比
- MEM: 所占用的内存
- TPS: 每秒事务数
- RT:响应时间,单位为毫秒

### 探针性能开销计算方法

在得到基线指标后,为测试应用安装ARMS Java探针,并在安装探针的情况下再次进行压测,然后观察 CPU、MEM、TPS和RT指标的情况,以此来计算探针性能开销。

探针性能开销 = (安装探针后的压测结果 - 基线指标)÷基线指标

## 基线指标

测试人员测试了应用在500、1,000以及2,000 TPS下的性能状况,并将得到的结果作为后续对比实验的基线指标。基线指标结果如下图所示:



## ARMS Java探针压测结果

本压测报告中使用的测试场景如下,每个场景分别使用500、1,000以及2,000 TPS进行压测:

• 场景一:使用ARMS Java探针默认配置,探针采样率为10%。

• 场景二: 使用ARMS Java探针默认配置, 探针采样率为100%。

② 说明 ARMS Java探针默认配置指的是在ARMS控制台的应用设置 > 自定义配置中默认开启的配置 项。

# 场景一压测详情

500 TPS下与基线指标对比:



可以看到在500 TPS的压测条件下,ARMS Java探针的额外CPU开销为1.49%(相对基线指标额外损耗 6.94%),额外内存开销为1.2%,RT损耗约为1毫秒。

1,000 TPS下与基线指标对比:



可以看到在1,000 TPS的压测条件下,ARMS Java探针的额外CPU开销为2.12%(相对基线指标额外损耗 6.54%),额外内存开销为1.18%,RT损耗约为1毫秒。



#### 2,000 TPS下与基线指标对比:

可以看到在2,000 TPS的压测条件下, ARMS Java探针的额外CPU开销为3.67%(相对基线指标额外损耗 6.90%),额外内存开销为1.2%, RT损耗为0毫秒。

### 场景二压测详情

#### 500 TPS下与基线指标对比:



可以看到在500 TPS的压测条件下,ARMS Java探针的额外CPU开销为1.51%(相对基线指标额外损耗 7.03%),额外内存开销为1.25%,RT损耗为0毫秒。

1,000 TPS下与基线指标对比:



可以看到在1,000 TPS的压测条件下,ARMS Java探针的额外CPU开销为2.85%(相对基线指标额外损耗 8.79%),额外内存开销为1.54%,RT损耗为1毫秒。



#### 2,000 TPS下与基线指标对比:

可以看到在2,000 TPS的压测条件下,ARMS Java探针的额外CPU开销为4.64%(相对基线指标额外损耗 8.72%),额外内存开销为1.15%,RT损耗为0毫秒。

# ARMS Java探针性能总结 CPU

对比项	500 T PS	1,000 TPS	2,000 T PS
基线	21.47%	32.43%	53.21%
场景一	22.96%	34.55%	56.88%
场景二	22.98%	35.28%	57.85%

# Memory (MEM)

对比项	500 T PS	1,000 TPS	2,000 T PS
基线	53.51%	53.62%	54.09%
场景一	54.71%	54.80%	55.29%
场景二	54.76%	55.16%	55.24%

#### RT

对比项	500 T PS	1,000 TPS	2,000 T PS
基线	107 ms	109 ms	129 ms
场景一	108 ms	110 ms	129 ms
场景二	107 ms	110 ms	129 ms

在不同TPS下,经过对相同默认配置不同采样率的ARMS Java探针进行性能压测和分析,可以看到ARMS Java 探针能够在CPU额外开销5%~10%以内,内存开销100M以内的情况下(其中Met aspace会带来30M左右的增长),正常提供大部分服务。

# 5.2. ARMS应用监控支持的Java组件和框架

本文列出了ARMS应用监控支持的Java第三方组件和框架。如果待监控应用使用的组件或框架不在支持范围内,则需要通过配置通用Filter拦截器的方式进行监控数据采集。

### ARMS应用监控支持的JDK版本

- JDK 1.7.0+
- JDK 1.8.0_25+

⑦ 说明 Kubernet es集群应用部署建议: JDK 1.8.0_191+。

• JDK 11.0.8+

# ARMS应用监控支持的Java组件和框架

⑦ 说明 ARMS对于JDK 11版本组件的支持目前处于公测期,如需体验,请联系ARMS服务钉钉账号 arms160804 申请使用。

组件	JDK 1.7	JDK 1.8	JDK 11
Aliware MQ	4.1.X+	4.1.X+	4.1.X+
Druid	1.0.0+	1.2.4+	1.2.4+
DSF(Daojia Service Framework)	不支持	2.1.X+	2.1.X+
Dubbo	2.5.X+	2.5.X+	2.5.X+
Elasticsearch Rest Client	5.X+	7.X+	7.X+
Elasticsearch Rest High Level Client	5.X+	7.X+	7.X+
Feign	不支持	9.X+	9.X+
Google HTTP Client	1.10.X+	1.10.X+	1.10.X+

#### 应用实时监控服务 ARMS

组件	JDK 1.7	JDK 1.8	JDK 11
GRPC-Java	1.15+	1.15+	1.15+
HikariCP	2.3.0+	2.3.13+	2.3.13+
Ali-HSF(High Speed Framework)	2.2.4.6-TLS+	2.2.4.6-TLS+	2.3.13+
HttpClient 3	3.X+	3.X+	3.X+
HttpClient 4	4.X+	4.X+	4.X+
Hystrix	1.5.X+	1.5.X+	1.5.X+
JDK HTTP	1.7.X+	1.7.X+	1.7.X+
Jedis	2.X+	2.X+	2.X+
Jetty	8.X+	8.X+	8.X+
Lettuce	不支持	4.0+	4.0+
MariaDB	1.3+	1.3+	1.3+
MemCached	2.8+	2.8+	2.8+
MongoDB	3.7+	3.7+	3.7+
MyBatis	3.X+	3.X+	3.X+
MySQL JDBC	5.0.X+	5.0.X+	5.0.X+
OKHttp	2.X+	2.X+	2.X+
Oracle JDBC	10.2.X+	10.2.X+	10.2.X+
Play Framework	不支持	1.4.X	1.4.X
PostgreSQL JDBC	9.4+	9.4+	9.4+
RabbitMQ	不支持	2.1.X+	2.1.X+
Reactor	不支持	3.X+	3.X+
Reactor Netty	不支持	0.9+	0.9+
Redisson	不支持	3.10.0+	3.10.0+
Resin	3.0+	3.0+	3.0+
RxJava	2.X+	2.X+	2.X+
SchedulerX	1.2.0+	1.2.0+	1.2.0+

组件	JDK 1.7	JDK 1.8	JDK 11
SofaRPC	5.4.X+	5.4.X+	5.4.X+
Spring	4.X+	4.X+	4.X+
Spring Boot	1.3.X+	1.3.X+	1.3.X+
Spring Cloud Gateway	不支持	2.1.0.RELEASE+	2.1.0.RELEASE+
Spring WebMVC	不支持	5.0.6.RELEASE+	5.0.6.RELEASE+
Spring WebFlux	不支持	5.0.0.RELEASE+	5.0.0.RELEASE+
SQLServer JDBC	6.4+	6.4+	6.4+
Thrift	0.8+	0.8+	0.8+
Tomcat	7.X+	7.X+	7.X+
Undertow	1.3X+	1.3X+	1.3X+
WebLogic	12.X+	12.X+	12.X+

# 配置通用Filter拦截器以采集数据

如果应用的组件或框架不在支持范围内,可以通过配置通用Filter拦截器的方式采集数据。配置步骤:

1. 在工程的pom.xml文件中引入arms-sdk-1.7.1.jar。

```
<dependency>
    <groupId>com.alibaba.arms.apm</groupId>
    <artifactId>arms-sdk</artifactId>
        <version>1.7.1</version>
</dependency>
```

⑦ 说明 如果无法获取pom.xml文件,请下载arms-sdk-1.7.1.jar。

2. 在web.xml中配置ARMS的Filter拦截器。

```
<filter>
<filter-name>EagleEyeFilter</filter-name>
<filter-class>com.alibaba.arms.filter.EagleEyeFilter</filter-class>
</filter>
<filter-mapping>
<filter-name>EagleEyeFilter</filter-name>
<url-pattern>/*</url-pattern>
</filter-mapping>
```

- 3. 登录ARMS控制台。
- 4. 将Java应用接入ARMS。具体操作,请参见为Java应用安装探针。
- 5. 重启应用令配置生效。

# 5.3. ARMS应用监控支持的Kafka版本

2.7.1.2及以上版本的ARMS Agent增加了Kafka插件,使ARMS应用监控可以对应用中特定版本和接入方式的 Kafka进行Trace跟踪。本文列出了ARMS应用监控支持的Kafka版本,以及不同接入方式的注意事项。

### ARMS Agent版本要求

仅v2.7.1.2及以上版本的ARMS Agent增加了Kafka插件。如果你需要使用包含Kafka插件的ARMS Agent,请 联系ARMS钉钉服务账号 arms160804 ,为您进行探针版本升级。

### Kafka版本要求

ARMS Agent的Kafka插件实现Trace跟踪需要依赖Kafka消息协议(Message Protocol)所支持的Header的能力,Kafka消息协议共有以下3个版本。

消息版本	Magic Number	Kafka版本	是否支持Header
v0	0	<v0.10.0.x< td=""><td>否</td></v0.10.0.x<>	否
v1	1	[v0.10.0.x, v0.11.0.x)	否
v2	2	≥v0.11.0.x	是

由于仅v0.11.0及以上版本的Kafka支持Header,因此,ARMS应用监控仅支持v0.11.0及以上版本的Kafka。

## Kafka的客户端(Producer和Consumer)和服务端(Broker)版本要求

在理想情况下,客户端和服务端的版本应该是相同的,即Kafka Broker如果使用X版本,那么Producer和 Consumer应该使用的也是X版本,这样可减少很多运行中的负担。

但是实际上,很多用户经常会采用不匹配的版本,这不可避免的带来了一些额外的复杂度。

因此, ARMS Agent的Kafka插件对不同版本的客户端和服务端设置了以下约束, 即不论Kafka客户端 (Producer和Consumer)和服务端(Broker)版本是否相同, 只要都大于v0.11.0, ARMS Agent的Kafka插 件就会注入Trace数据。

服务端(Borker)	客户端(Provider和Consumer)	Kafka插件是否会注入Trace数据
[v0.10.0.x, v0.11.0.x)	<v0.10.0.x< td=""><td>否</td></v0.10.0.x<>	否
	[v0.10.0.x, v0.11.0.x)	否
	≥v0.11.0.x	否
≥v0.11.0.x	<v0.10.0.x< td=""><td>否</td></v0.10.0.x<>	否
	[v0.10.0.x, v0.11.0.x)	否
	≥v0.11.0.x	是

## Kafka接入方式

ARMS对以下两种Kafka接入方式进行了验证。

#### Producer

Apache Kafka Provider API无需额外操作即可通过官方接入方式接入ARMS。更多信息,请参见Apache Kafka官方文档。

#### Consumer

Apache Kafka Consumer API并不支持Consumer逐个消费消息,而是采用Pull模式不间断的轮询来消费消息,因此ARMS Agent的Kafka插件没办法直接将Trace注入,需要一个额外的地方进行Consumer消息的注入。如果直接采用Apache Kafka客户端进行Consumer端的接入,ARMS需要进行以下操作。

#### Consumer端示例文件:

package arms.test.kafka;
<pre>public class KafkaConsumeTest {</pre>
<pre>public void testConsumer() {</pre>
<pre>Properties props = new Properties();</pre>
<pre>props.put("bootstrap.servers", "PLAINTEXT://XXXX");</pre>
<pre>props.put("group.id", UUID.randomUUID().toString());</pre>
<pre>props.put("enable.auto.commit", "true");</pre>
<pre>props.put("auto.offset.reset", "earliest");</pre>
<pre>props.put("auto.commit.interval.ms", "1000");</pre>
props.put("key.deserializer", "org.apache.kafka.common.serialization.StringDese
<pre>rializer");</pre>
props.put("value.deserializer", "org.apache.kafka.common.serialization.StringDe
serializer");
KafkaConsumer <string,string> kafkaConsumer = new KafkaConsumer<string, string="">(</string,></string,string>
props);
<pre>kafkaConsumer.subscribe(Arrays.asList("test"));</pre>
while (true) {
ListAdapter consumer;
ConsumerRecords <string, string=""> records = kafkaConsumer.poll(100);</string,>
for (ConsumerRecord <string, string=""> record : records)</string,>
handler(record);
}
}
<pre>public void handler(ConsumerRecord<string, string=""> record){</string,></pre>
LOGGER.info( Utils.string(record));
}
}

对于以上示例文件,对应的在ARMS Agent安装包的arms-agent.config文件中,添加以下内容。

profiler.kafka.consumer.entryPoint=arms.test.kafka.KafkaConsumeTest.handler

其中, arms.test.kafka.KafkaConsumeTest.handler 为示例文件中的包、类和方法名。

Spring Kaf ka在Apache Kaf ka客户端之上做了一层封装,因此,引入的Spring Kaf ka版本和Apache Kaf ka客 户端版本需要满足以下映射关系。更多信息,请参见Spring Kaf ka官方文档。

Spring for Apache Kafka Version	Spring Integration for Apache Kafka Version	kafka-clients	Spring Boot
v2.7.0	v5.4.x	v2.7.0或v2.8.0	v2.4.x或v2.5.x
v2.6.x	v5.3.x或v5.4.x	v2.6.0	v2.3.x或v2.4.x
v2.5.x	v3.3.x	v2.5.1	v2.3.x

Spring for Apache Kafka Version	Spring Integration for Apache Kafka Version	kafka-clients	Spring Boot
v2.4.x	v3.2.x	v2.4.1	v2.2.x
v2.3.x	v3.2.x	v2.3.1	v2.2.x

如果您使用的是间接接入Spring Kafka的方式接入Kafka Broker,那么您不需要其他任何配置,ARMS Agent Kafka插件默认支持这种方式的Trace链路追踪。

# 间接接入Spring Kafka

## Overhead Trace数据

ARMS Agent的Kafka插件默认为Kafka Message Header添加以下数据, ARMS应用监控即可通过以下数据实现Trace跟踪。

- EagleEye-TraceID
- EagleEye-RpcID
- EagleEye-SpanID
- EagleEye-pSpanID
- EagleEye-pAppName
- EagleEye-pRpc
- EagleEye-IP
- EagleEye-ROOT-APP
- EagleEye-Sampled
- uber-trace-id
- X-B3-Traceld
- X-B3-Spanld
- X-B3-Parent Spanld
- X-B3-Flags
- X-B3-Sampled
- traceparent

# 5.4. Agent版本说明

本文主要介绍了ARMS应用监控Java Agent的版本更新历史。

# Java Agent版本

版本 发布时间 发布说明

版本	发布时间	发布说明
2.7.1.3	2021年10月18日	<ul> <li>支持XXL-JOB、SchedulerX、 ElasticSearch和VertX。</li> <li>支持Arthas诊断、客户端自定义 采样、应用安全攻击防护以及危 险组件检测。</li> <li>支持JDBC返回大小、支持Redis命 中率、调整了池化监控、优化了 部分异步场景的支持。</li> </ul>
2.7.1.2	2021年09月17日	<ul> <li>支持Kafka、RocketMQ、Sofa等 插件,支持池化监控,增加调用 链展示信息,支持FC场景,支持 基于Spring注解的URL收敛。</li> <li>支持Dubbo、SpringCloud全链路 灰度,支持SpringCloudGateway 金丝雀发布,支持Nacos、 Eureka注册中心迁移,服务降级 功能,Dubbo、Spring Cloud支 持同可用区优先路由。</li> <li>优化启动时间、内存使用率,修 复内存快照、线程诊断、数据上 报以及SDK等部分Bug,修复安全 漏洞。</li> </ul>
2.7.1.1	2020年08月14日	<ul> <li>支持NoSQL监控。</li> <li>支持微服务标签路由。</li> <li>支持N+1调用压缩。</li> <li>修复金融云网络连接问题,优化 内存占用。</li> </ul>
2.7.1	2020年07月16日	<ul> <li>支持新版本Jedis插件,解决拓扑 图Redis集群不识别问题。</li> </ul>
2.7.0	2020年05月20日	● 支持微服务子产品功能。
2.6.2	2020年05月20日	● 支持业务监控。
2.6.1.2	2020年03月19日	<ul><li>支持微服务鉴权。</li><li>支持微服务优雅下线</li></ul>
2.6.1.1	2020年03月16日	<ul> <li>支持SpringCloud Gateway及 Spring Webflux等组件。</li> </ul>

版本	发布时间	发布说明
2.6.1	2020年02月14日	<ul> <li>支持获取微服务元数据等相关功 能。</li> </ul>
2.6.0.2	2020年01月02日	<ul><li>支持新版异常分析。</li><li>修复Thrift插件问题。</li></ul>
2.6.0	2019年12月17日	<ul> <li>支持异步调用链。</li> <li>Dubbo/HSFProvider调用参数将 被记录。</li> <li>修复现有插件若干问题。</li> </ul>
2.5.9.5	2019年11月28日	<ul> <li>支持jfinal-undertow插件。</li> <li>若干错误修复,包括获取不到 Dubbo线程剖析数据等问题。</li> </ul>
2.5.9.3	2019年11月25日	<ul><li>支持ARMS和链路追踪产品打通。</li><li>若干错误修复和性能优化。</li></ul>
2.5.9	2019年09月06日	<ul> <li>修复FastJson拒绝服务漏洞。</li> <li>修改获取网卡IP逻辑。</li> </ul>
2.5.8	2019年08月02日	<ul> <li>支持二元状态报警功能,即针对 仅具有是和否、有和无这两种状 态的指标设置报警规则。</li> <li>支持国产达梦数据库插件。</li> </ul>
2.5.7.2	2019年07月30日	<ul> <li>支持JVM Metaspace指标。</li> <li>支持自定义要忽略的HTTP状态码。默认情况下,大于400的状态码会计入错误数,您可以自定义大于400但不计入的HTTP状态码。[相关文档]</li> </ul>
2.5.7	2019年07月11日	升级依赖的有安全漏洞的FastJson版 本。
2.5.6.1	2019年06月28日	<ul> <li>支持Dubbo/MariaDB插件。</li> <li>自定义配置支持获取SQL绑定值: 捕获PrepareStatement参数绑定 的变量值,无需重启应用即可生 效。[相关文档]</li> <li>优化内存和修复若干错误。</li> <li>去除Log4j日志依赖,避免冲突。</li> </ul>

版本	发布时间	发布说明
2.5.6	2019年06月07日	<ul><li>支持分位数统计功能。</li><li>优化功能和修复若干错误。</li></ul>
2.5.5	2019年06月03日	<ul><li>支持HSF-HTTP调用。</li><li>优化功能和修复若干错误。</li></ul>
2.5.3	2019年03月15日	<ul> <li>支持应用运行过程中的线程指标 上报。</li> <li>支持Spring-Data-Redis插件。</li> <li>支持Druid数据库连接池插件。</li> </ul>
2.5.2	2019年02月21日	<ul> <li>增加文件句柄数采集。</li> <li>支持GC时间及次数瞬时值上报。</li> <li>支持自定义配置请求入参最大长度。[相关文档]</li> </ul>
2.5.1	2019年01月14日	<ul> <li>支持调用链压缩。[相关文档]</li> <li>支持不通过控制台创建应用监控 任务的方式。</li> <li>优化功能和修复若干错误。</li> </ul>
2.5.0	2018年12月28日	<ul> <li>支持一键接入,无需重启应用。</li> <li>完善主机监控,支持Windows系统。</li> <li>支持Spring-webflux。</li> <li>优化功能和修复若干错误。</li> </ul>
2.4.6	2018年10月26日	<ul> <li>支持GRPC、THRIFT、 XMemcached插件。</li> <li>支持接口调用拓扑展示。</li> <li>支持覆盖前后端的拓扑展示。</li> </ul>
2.4.5	2018年09月17日	<ul> <li>支持Lettuce插件(JRE 1.8+)。</li> <li>支持MongoDB插件。</li> <li>采集异常详细信息。</li> </ul>
2.4.4	2018年08月06日	<ul> <li>支持应用线程剖析数据上报。</li> <li>支持Memcached缓存。</li> <li>支持自定义配置异常过滤。[相关 文档]</li> </ul>

版本	发布时间	发布说明
2.4.3.1	2018年06月29日	<ul> <li>支持WebLogic服务器。</li> <li>支持Undertow服务器。</li> <li>优化Agent内存占用。</li> <li>优化Agent启动加载时间。</li> <li>解决JVM监控/主机监控指标可能 无法上报问题。</li> </ul>
2.4.3	2018年05月18日	<ul> <li>支持采集消息队列Rocket MQ监控 指标。</li> <li>支持监控方法自定义。</li> <li>解决限流场景下频繁输出日志的 问题。</li> <li>支持自定义配置本地方法堆栈最 大长度。[相关文档]</li> <li>优化采样功能,不对异常调用链 进行采样。</li> </ul>
2.4.2	2018年04月19日	<ul> <li>支持自定义配置信息读取。</li> <li>支持通过SDK方式实时获取链路信息。</li> <li>支持线程、GC次数/耗时等JVM指标采集。</li> <li>支持HSF方法级调用监控。</li> <li>支持主机监控(CPU/物理内存/网络/磁盘)等指标采集。</li> <li>解决Tomcat环境下通过./shutdown.sh停止进程时可能卡住的问题。</li> </ul>
2.4.1	2018年03月24日	<ul> <li>支持JVM监控,如堆内存、非堆内 存等指标上报。</li> <li>支持PlayFrameWork 1.4.4版本。</li> <li>支持自定义配置采样率、Agent开 关、日志级别、阈值参数等。[相 关文档]</li> </ul>
2.4.0	2018年02月14日	<ul> <li>支持PostgreSQL数据库。</li> <li>支持阿里云各地域的ECS与ARMS服务器进行内网通讯。</li> <li>支持ARMS应用监控正式商用。</li> </ul>

# 5.5. 关键统计指标说明

本文说明了ARMS应用监控各页面的关键统计指标的含义。

#### 基本概念

本文涉及以下基本概念:

● APDEX性能指数

APDEX性能指数(Application Performance Index)是一个国际通用的应用性能计算标准。该标准将用户 对应用的使用感受定义为三个等级:

- 满意(0~T)
- 可容忍(T~4T)
- 不满意(大于4T)



图片来源: apdex.org 计算公式为:

```
Apdex=(满意数+可容忍数/2)/总样本量
```

ARMS取应用的平均响应时间作为计算指标,并将T定义为500毫秒。

● 实例

实例是指被监控的应用所部署的机器,以JVM为粒度。例如在下图中, "a3"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。

响应时间 🛊	请求数 🔹	错误数 🛊	
a3			>
•	18	94.7917ms / 72	2/2 >
•	7	/35.6364ms / 33	3/0 🔉
•	3	860.9189ms / 74	4/3 🔉
	3	344.0238ms / 42	2/0 🕨

# 相关统计页面

应用列表页面

在控制台左侧导航栏中选择**应用监控 > 应用列表**,即可看到各应用的健康度、请求数、错误数、响应时间、异常数、状态和最近10分钟响应时间曲线。

• 应用总览页面

在应用监控列表页面单击应用名称,即可进入**应用总览**页面。在页面顶部选择相应菜单,可以查看其他维度的统计信息。

- 概览分析页签
  - 应用提供的服务:请求量和平均响应时长
  - 应用依赖的服务:请求量、平均响应时长、实例数和HTTP-状态码

- 系统信息: CPU、内存和负载
- 统计分析: 慢接口调用分析和平均响应时间、异常类型和出现次数
- 拓扑图页签
  - 应用拓扑图
  - 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
调用本地API	对本地API进行操作的调用	APl调用
НТТРДП	客户端使用HTTP协议调用该应用 的入口	服务入口调用
调用Dubbo	Dubbo的消费者产生的调用	服务入口调用
调用HSF	HSF服务的消费者产生的调用	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
提供HSF	HSF的生产者产生的调用	服务间调用
提供Dubbo	Dubbo的生产者产生的调用	服务间调用
调用MySQL	对MySQL进行操作的调用	数据库调用
调用Oracle	对Oracle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

- 实例IP: 显示该应用下所有实例的IP。
- 应用每分钟的请求数、响应时间和错误率。

○ 3D拓扑页签

- QPS: Query Per Second (每秒查询数)
- RT(ms): Response Time (响应时间, 单位为毫秒)
- Error: Error QPS (每秒错误查询数)
- 应用详情页面

此页面展示当前应用的调用详细信息。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分析、异常分析、接口快照等维度的详细分析。

● 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。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分析、异常分析、接口快照等维度的详细分析。

● 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。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分析、异常分析等维度的详细分析。

### 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。
错误率	(应用、实例调用的异常调用次数,或数据库操作的异 常次数)/请求数。

#### ● SQL分析页签

上报字段	描述
SQL调用统计	柱状图与左Y轴为数据库请求数统计,折线图与右Y轴为 数据库响应时间。
平均耗时	本次数据库调用的平均耗时。
调用次数	该应用此类型数据库调用次数。

#### ● 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。
异常类型	采集到的抛错类型。
异常详细信息	抛错的详细信息。
平均耗时	本次错误调用的平均耗时。
错误数	该异常类型的错误出现的次数。

#### ● 接口快照页签

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。
Traceld	应用、实例调用的索引ID,单击可以跳转到该调用链详 情。

# 5.6. ARMS SDK使用说明

借助ARMS提供的SDK,您可以在业务代码中动态获取Traceld及相关调用链属性。

⑦ 说明 ARMS已支持通过OpenTelemetry Java SDK埋点监控您的应用,建议您优先选择使用 OpenTelemetry Java SDK进行埋点。更多信息,请参见通过OpenTelemetry Java SDK进行手工埋点。

### 前提条件

- 已在ARMS控制台上创建应用监控,并已在Java程序中挂载和启动应用监控的Agent。具体操作,请参见为 Java应用手动安装Agent。
- 程序中已引入arms-sdk-1.7.3.jar依赖。

```
<dependency>
    <groupId>com.alibaba.arms.apm</groupId>
    <artifactId>arms-sdk</artifactId>
    <version>1.7.3</version>
</dependency>
```

⑦ 说明 如果无法获取pom.xml中的依赖,请直接下载arms-sdk-1.7.3.jar。

# 获取TraceId与RpcId

您可通过以下代码获取Traceld与Rpcld。

```
Span span = Tracer.builder().getSpan();
String traceId = span.getTraceId();
String rpcId = span.getRpcId();
```

# 透传业务自定义标签baggage

若您要透传业务自定义标签,则需要在代码中写入添加和获取自定义标签,具体操作步骤如下:

1. 在业务代码中添加自定义标签baggage。

```
Map<String, String> baggage = new HashMap<String, String>();
baggage.put("key-01", "value-01");
baggage.put("key-02", "value-02");
baggage.put("key-03", "value-03");
Span span = Tracer.builder().getSpan();
span.withBaggage(baggage);
```

2. 在业务代码中获取自定义标签baggage。

```
Span span = Tracer.builder().getSpan();
Map<String, String> baggage = span.baggageItems();
```

### 为Span设置自定义标签tag

为Span设置自定义标签只会在当前Span中有效,并不会透传。您需要在代码中写入添加和获取自定义的标签,具体操作步骤如下:

1. 在业务代码中为Span添加自定义标签tag,可以添加多个标签。

```
Span span = Tracer.builder().getSpan();
// Add a tag to the Span.
span.setTag("tag-key1", "tag-value1");
span.setTag("tag-key2", "tag-value2");
```

2. 在业务代码中获取Span自定义标签tag。

```
Span span = Tracer.builder().getSpan();
// Inspect the Span's tags.
Map<String, String> tags = span.tags();
```

### 根据自定义标签baggage和tag查询调用链

通过Span设置的自定义标签baggage和tag可以用来按标签维度查询调用链。

- baggage上的标签具有透传到下游效果,一般用于业务染色,标签项不建议设置过多。
- tag上的标签只在本Span作用域内有效,可以设置多个业务项。
  - 1. 登录ARMS控制台。
  - 2. 在左侧导航栏选择应用监控 > 调用链路查询,并在顶部菜单栏选择目标地域。
  - 3. 在调用链路查询页面选择参数名,在参数值中填入自定义标签,单击查询。
  - 4. 在调用链路查询列表中单击目标链路的TraceID。

	调用链路	查询												
	调用智	连路查询	全息排查事件	牛查询	❷调用链路	查询								
		参数名	日期			参数值	2020-08-	14 17:11 至 2020-08-14 17:16		0				
		参数名	客户端名			参数值	空		•					
		参数名	服务端名			参数值	空		•					
		参数名	tag-bu			参数值	taobao			0	Q查询	❤收藏	分享	
	为您查询到	100 条结果	l.											
<	TraceID			产生日志	5时间 🛊	接口名和	尔	所属应用名称	耗服	₫ \$	客户端	服务端		
	Sect 1018	5875800	1001102011020	2020-08 17:15:59	3-14 9	/api/buy	/	show-consumer	381	l (ms)	客户端名: - 客户端IP: -	服务端名: 服务端lp:	show-consumer 30.225.16.125	
	Section 4	1012000	een in kaberieka	2020-08 17:15:59	3-14 9	/api/buy	/	show-consumer	347	7 (ms)	客户端名: - 客户端IP: -	服务端名: 服务端lp:	show-consumer 30.225.16.125	
	Sectory	1673802	0029-521-050	2020-08 17:15:59	3-14 9	/api/buy	/	show-consumer	475	ō (ms)	客户端名: - 客户端IP: -	服务端名: 服务端lp:	show-consumer 30.225.16.125	
	Sectory.	10739052	20101111A-1010	2020-08 17:15:52	3-14 2	/api/buy	/	show-consumer	319	) (ms)	客户端名: - 客户端IP: -	服务端名: 服务端lp:	show-consumer 30.225.16.125	
	Sect 1014	09709001	1988-01347068	2020-08 17:15:5	3-14 1	/api/buy	/	show-consumer	385	ō (ms)	客户端名: - 客户端IP: -	服务端名: 服务端lp:	show-consumer 30.225.16.125	

5. 在**调用链路**详情页面,鼠标移动至**服务名称**,会显示当前Span对应的Tags信息(baggage内容会自动 加入到每个Span的Tags上)。

调用链路 业务轨迹									
应用名称		日志产生时间	状态	IP地址	调用类型	服务名称	方法栈	线程剖 析	时间轴(ms)
show-consumer	/api/buy					] 🗍 /api/buy	Ð		35012ms
show-end	show-end Service: show-consumer Duration: 35012 Start Time: 2020-04				20-08-	api/logistics	Ð		376ms
show-provider1	Tags:					] /api/electric	Q		1485ms
	http.status.	code 200							_
	cs	4bbe28c3(A	LIBAB	4)					
	key-01	value-01							
	tag-userid	123456789							
	tprof	1							
key-03 value-03									
	key-02	value-02							
	tag-bu	taobao							

# 5.7. Trace Explorer参数说明

本文介绍了Trace Explorer功能常见的参数说明。

此处仅介绍了ARMS应用监控Trace Explorer功能中常见的参数,更多信息,请参见OpenTelemetry的 Tracing数据模型。

参数	说明
attributes	用于索引的属性。
duration	耗时,单位为纳秒。
event	事件,包含时间戳、名称、属性等非索引字段。
hostname	主机名。
ір	主机IP地址。
kind	SPAN类型, SPAN类型包括以下几种: SPAN_KIND_UNSPECIFIED:未指定 SPAN_KIND_INTERNAL:内部方法 SPAN_KIND_SERVER:服务端 SPAN_KIND_CLIENT:客户端 SPAN_KIND_PRODUCER:生产者 SPAN_KIND_CONSUMER:消费者
links	当前SPAN和同一Trace或者不同Trace的关系。
parentSpanld	父SPAN分配的ID,表示SPAN之间的关系。
resources	SPAN的资源信息,包括进程ID、JVM版本号等。
serviceName	服务名,用于多租户。
spanld	SPAN分配的ID,表示SPAN之间关系。
spanName	SPAN的名称。
startTime	开始时间,单位为纳秒。

参数	说明
statusCode	SPAN状态, SPAN状态包括以下几种: • STATUS_CODE_UNSET:未设置 • STATUS_CODE_OK:正常 • STATUS_CODE_ERROR:错误
statusMessage	状态附加信息。
traceld	调用链的唯一标识。

# 5.8. Trace Explorer查询用法说明

Trace Explorer支持三种查询方式:快捷筛选、查询面板和查询框。查询条件会互相联动,您可以根据个人 习惯选择合适的查询方式。

### 快捷筛选

在Trace Explorer页面左侧快捷筛选区域,通过状态、耗时、应用名称、接口名称和主机地址维度快速筛选链路。例如选中异常,可以快速筛选出 statusCode=2 的异常链路;选中某个应用或接口,可以快速筛选该应用或接口下的链路。

快捷筛选的优势不仅是操作方便,还可以进行多级筛选。通过实时显示当前组合条件下的链路数据分布,帮助您发现潜在的异常特征。

Trace Explo	orer 🏾						最近30	分钟		
查询 分析 王	隐藏快捷第选	点击设置搜索条件, 或直接在输入框中输入搜索语句					Q	0 8	已保存视图 ~	
快速筛选 状态 ∨ □ 正常 □ 月常 純时 ∨	<u>次数</u> / 耗时 1b 4m 反	span®2 dm dm the the the the the the the the	HTTPB/500 25k 00k 50k 50k 50k 11.03.00 11.09.00 10.34.00 10.34.00 10.34.00 10.34.00 10.34.00 10.34.00 10.400 10.34.00 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.400 10.4000 10.4000 10.4000 10.4000 10.4000 10.4000 10.4000 10.4000 10.4000 10.4000 10.4000 10.4000 10.4000 10.4000 10.4000 10.4000 10.4000 10.4000 10.4000 10.4000 10.4000 10.4000 10.4000 10.4000 10.4000 10.4000 10.4000 10.4000 10.4000 10.4000 10.4000 10.4000 10.4000 10.4000 10.4000 10.4000 10.4000 10.40000000000	r41:00 10:44:00 10:51:00 10:5 # 004 406 500 -501	6.00 11.01.00 11.06.00	時町田分位 0ms 10.800 10.43.00 10.48.00 ■ pf9 ■ pf0 ■ pf3 ■ p53 ■ p50 ■ pg3	10:53:00 duration	10:58:00 1	1:03:00 11:08:0¢	
0ns –	26m37s	搜索到1b条span							۲	
应用名称~	<u>次数</u> / 耗时	traceld	startTime 14	serviceName	spanName	duration 14	ip		操作	
Q 搜索serviceNan	me字段 😡	6	03/25 11:08:59	xtrace	Rit	Ons			查看	
×	- ¹⁶	c	03/25 11:08:59	xtrace	Rit	1us			查看	
a □ x	4m 653k	8	03/25 11:08:59	xtrace	Rit	1us			查看	
	. 86k	8	03/25 11:08:59	xtrace	Rit	lus			查看	
×	15k	e	03/25 11:08:59	xtrace	Rit	Ons			查查	
接口名称 ~	<u>次数</u> / 純时	5	03/25 11:08:59	xtrace	Rit	Ons			查看	
Q 搜索spanName	字段的 📡	6	03/25 11:08:59	xtrace	Rit	lus			查看	
□ B	991r	7	03/25 11:08:59	xtrace	Rit	Ons			00	
9	32m		02/25 11:09:59	vtrace	pi,	One			**	
	8m		03/25 11:08:50	Auto	pi.	0				
P	6m		03/25 1108:59	xtrace	KI	uns			里容	Ę
主机地址 ~	<u>次数</u> / 耗时		03/25 11:08:59	xtrace	Ri	Ons				8

## 查询面板

在Trace Explorer页面单击右侧顶部文本框,在下拉查询面板中可以指定筛选条件的具体值,并支持添加 自定义Attributes筛选条件。

Trace Ex	plorer 0				最	后30分钟		
童術 分析	三 隐藏快趣算過	statusCode=2			0 Q	O B	已保存物	~ 259
快捷筛选		常用査询 ~		2 显示	字段说明			
状态~	23数 / 耗时	serviceName 应用名称	-	语因经动输入内容				
☑ 异常	3k	spanName 接口名称	-	请历经成能入内容				
		ip 主机地址	-	語话轻微意入內容			1	
₩KT ns ~	¥.	duration #EP3	BETWEEN	ns ·		14:43:00 14	:48:00 14	:53:00
Ons .	. 2m9s	高阶查询 >						0
		+ 添加查询条件			へ 收起		提作	*
Ons ·	- 2m9s <u>次数</u> / 形时	+ 添加查询条件			~ 收起		操作	

# 查询框

#### 在Trace Explorer页面查询框中直接输入筛选条件。

Trace Explorer 💿								
▲询 分析 三陰震快速等选	点击设置搜索条件,或直接在输入框中输入搜索语句	HQ置操聚条件,或直接在输入框中输入模数语句						
快速筛选 状态 ✓ <u>次数</u> / 耗时 □ 正常 1b □ 异常 4m	รองหมู่ใ อิก อิก	HTTP错误数 25k 00k			耗町百分位			
#町~	0m 0 10 20:00 10:35:00 10:42:00 10:49:00 10:56:00 1 count	50k	1::00 10::46:00 10:51:00 10:56 404 406 500 501	:00 11:01:00 11:06:00	0ms 10:38:00 10:43:00 10:48:0 ■ p99 ■ p90 ■ p75 ■ p50 ■ av	00 10:53:00 10:58:00 1 vg_duration	1:03:00 11:08:00	
0ns – 26m37s	搜索到1b祭span						۲	
应用名称 > <u>次数</u> / 耗时	traceld	startTime 14	serviceName	spanName	duration 14	ip	操作	
Q 搜索serviceName亭段 S	6	03/25 11:08:59	xtrace	Rii	Ons		查看	
<u>x</u> <u>1</u> b	c	03/25 11:08:59	xtrace	Rin	1us		查看	

#### Trace Explorer支持简单的查询语法,常用的运算符说明如下,查询参数请参见Trace Explorer参数说明。

运算符	说明
AND	AND运算符,用于组合多个查询条件,表示多重过滤。例如 serviceName="arms- demo" AND statusCode=2 表示查询arms-demo应用下的异常链路。
=	等于运算符,查询某字段等于某数值的链路,如果值为字符串类型则需要加上半角双引 号(""),数值类型则不需要添加。例如 serviceName="arms- demo",statusCode=2 。
>	大于运算符,仅可用于数值类型判断。例如 duration>10000000 。
<	小于运算符,仅可用于数值类型判断。例如 duration<30000000 。
IN	多选运算符。例如 ip IN ("127.0.0.1","192.168.0.1") 表示IP等于127.0.0.1 或者192.168.0.1,满足任一条件即可返回。 IN的候选值需要包含在半角圆括号内(()),每个值用半角双引号("")包裹,多个候选 值之间用半角逗号(,)分隔。IN的候选值至少有一个,不能为空。

#### 查询语法示例:

#### • 查询arms-demo应用下异常链路:

serviceName="arms-demo" AND statusCode=2

#### • 查询/createOrder/pc和/createOrder/mobile两个接口大于3s的链路:

spanName IN ("/createOrder/pc","/createOrder/mobile") AND duration>300000000

# 5.9. ARMS OpenTelemetry Collector指 标说明

ARMS OpenTelemetry Collector支持将OpenTelemetry Trace数据自动预聚合为APM Metrics指标。为了适应生产环境大规模应用场景, ARMS OpenTelemetry Collector在OpenTelemetry Metrics规范的基础上进行了调整与扩展。

# 基于时间窗口轮转的预聚合统计

ARMS OpenTelemetry Collector预聚合的APM Metrics指标支持时间窗口轮转,每个时间窗口(通常是1分钟)的指标值都是独立的,不受之前窗口统计的影响。这样可以保证在指标发散情况下,内存不会轻易溢出;并且在发生数据延迟、丢失等异常场景下,每个时间窗口内的指标数据都是准确的,更加适合生产环境的大规模应用。

例如:A接口每分钟的调用量是100次,累加计数统计和基于时间窗口轮转的统计结果对比如下所示。

方式\时 间	1分钟	2分钟	3分钟	4分钟	5分钟	6分钟	7分钟	8分钟
累加计数	100	200	300	400	500	600	700	800
时间窗口 轮转	100	100	100	100	100	100	100	100



## 指标说明

ARMS OpenTelemetry Collector目前支持HTTP、RPC、DB、Messaging等多种指标类型,每种类型都包含通用标签和专属类型标签。

指标查询示例如下图所示。



# 通用标签

通用标签是指所有指标类型都有且必须包含的字段。

通用标签	说明	示例
source	数据来源	<ul><li>satellite</li><li>ebpf</li></ul>
regionId	区域名称	cn-hangzhou
clusterId	集群名称	cf055ad09b7714beea8f9e7271e6 1xxxx
ip	主机IP(K8s环境下是Pod IP)	127.0.0.1
serviceName	应用名称	arms-console-hz
spanName	接口名称	/hello
statusCode	调用状态	<ul> <li>0表示UNSET。</li> <li>1表示OK。</li> <li>2表示ERROR。</li> </ul>
componentType	组件类型	mysql、redis、grpc、kafka等。

# 指标类型及专属标签

#### HTTP

指标名称	单位	描述
http_server_duration_ms_count	次数	HTTP服务端调用次数
http_server_duration_ms_sum	毫秒	HTTP服务端调用总耗时
http_client_duration_ms_count	次数	HTTP客户端调用次数
http_client_duration_ms_sum	毫秒	HTTP客户端调用总耗时

专属标签	类型	推荐	描述	示例
httpMethod	String	可选	方法名称	GET、POST、HEAD
httpStatusCode	String	可选	响应码	1xx、2xx、3xx、 4xx、5xx

#### RPC

指标名称	单位	描述
rpc_server_duration_ms_count	次数	RPC服务端调用次数
rpc_server_duration_ms_sum	毫秒	RPC服务端调用总耗时

指标名称	单位	描述
rpc_client_duration_ms_count	次数	RPC客户端调用次数
rpc_client_duration_ms_sum	毫秒	RPC客户端调用总耗时

专属标签	类型	推荐	描述	示例
rpcService	String	可选	含包名的接口全称	myservice.EchoSer vice
rpcMethod	String	可选	仅含方法名	exampleMethod

#### Database

指标名称	单位	描述
db_client_duration_ms_count	次数	DB客户端调用次数
db_client_duration_ms_sum	毫秒	DB客户端调用总耗时

专属标签	类型	推荐	描述	示例
dbStatement	String	必选	SQL语句	<ul><li>SELECT * FROM</li><li>wuser_table</li></ul>
dbName	String	可选	数据库名称	<ul><li>customers</li><li>main</li></ul>
dbTableName	String	可选	表名称	user_info
dbOperation	String	可选	动作类型	<ul><li>SELECT</li><li>HMSET</li></ul>

#### Messaging

指标名称	单位	描述
msg_producer_duration_ms_coun t	次数	消息生产者调用次数
msg_producer_duration_ms_sum	毫秒	消息生产者调用总耗时
msg_consumer_duration_ms_cou nt	次数	消息消费者调用次数
msg_consumer_duration_ms_su m	毫秒	消息消费者调用总耗时

专属标签	类型	推荐	描述	示例
msgTopic	String	必选	Span名称	<ul><li>MyQueue</li><li>MyTopic</li></ul>
msgProtocol	String	可选	协议	<ul><li>AMQP</li><li>MQTT</li></ul>
msgProtocolVersi on	String	可选	协议版本	0.9.1

#### Other

#### 不属于上述类型的指标统一归类到Other类型。

指标名称	单位	描述
other_type_duration_ms_count	次数	其他类型调用次数
other_type_duration_ms_sum	毫秒	其他类型调用总耗时

# 自定义标签

ARMS OpenTelemetry Collector支持在聚合指标中添加自定义标签。

- 1. 在Trace数据的Attributes中添加KV键值对,例如 <userType: xxx> 。
- 2. 在ARMS控制台的应用监控 > 全局配置 > OT Collector配置页签,选择对应地域和集群,在自定义 标签区域添加自定义标签Key,例如userType,相应的Value将会自动从Trace Attributes中提取。

自定义标签 ⑦ 添加自定文标签			
标签名称	标签 key	应用范围 (应用名称列表) ⑦	最高處用 操作
userType	userType	SalLapp	Sibili Hite

## 指标查询

ARMS OpenTelemetry Collector会将聚合后的指标数据自动上报并存储在ARMS Prometheus托管集群,您可以通过Grafana或HTTP API等方式进行查询或展示。

## 通过Grafana查看APM Metrics

登录<mark>共享版Grafana控制台</mark>,在**Explorer**页面选择自动创建的ARMS OpenTelemetry Collector数据源,查看 指标数据。

- ARMS OpenTelemetry Collector数据源名称: ot-collector-{regionId}_{userId}, 例如ot-collector-cn-beijing_1131971649496xxx。
- ARMS OpenTelemetry Collector数据源类型: Prometheus。

# 您也可以在Grafana的Dashboards > Manage页面中搜索查看默认创建的OpenTelemetry APM Dashboard。

器 ot-i		/ OpenT ☆ 🗠			nd+	🖻 💿 🖵 🕑 Last 30	minutes ~ Q Q ~
N 建来源 All → 集群iD .	All -	All ~ 接口名称 All ~					
~应用总览 ◎ 自							
在线应用数 在线实例数 在线接口数		全部调用 (当前)	全部调用(当前) 错误调用(当前)		平均耗时 (当前)		(前)
11	1 4	60K cpm 40K cpm	50 cpm 40 cpm 30 cpm 20 cpm	~~~~~	m	2.0 s 1.5 s	
调用总量 (当前)	调用总量(-24H)	D com	10 cpm			500 ms	
3.52 Mil	4.02 Mil	17:15 17:20 17:25 17:30 17:35	17:40 17:	15 17:20 17:25 17:30	17:35 17:40	17:15 17:20 17:25	17:30 17:35 17:40
错误总量 (当前)	错误总量(-24H)	全部调用 (-24H)	() timeshift -24h	错误调用(-24H)	() timeshift -24h	平均耗时(-2	(4H) ③ timeshift -24h
1.17 К	1.57 K	60K cpm 40K cpm	100 cpm 75 cpm			2.0 s 1.5 s	
平均耗时 (当前)	平均耗时 (-24H)	20K apm	50 cpm 25 cpm		500	1.0 s 500 ms	
627.12 ms	626.41 ms	17:15 17:20 17:25 17:30 17:35	17:40 1:	15 17:20 17:25 17:30	17:35 17:40	17:15 17:20 17:25	17:30 17:35 17:40
~ HTTP 服务端总览							
HTTP 组件类型			HTTP 响应码		HTTP 请求类型		
60 K		60 K			30 K		
40 K		40 K			20 K		
30 K		30 K			15 K		
20 K		20 K			10 K		
10 K		10 K			SK		
17:15 17:20	17:25 17:30 17:35	17:40 17:15 17:20	17:25 17:30	17:35 17:40	17:15 17:20	0 17:25 17:30	17:35 17:40
- (component Type= None )		- (httpstatuscode# 2xx)			- (nttpMethod= GET)	- (nttpMethod= POST )	
		н	TP 服务端接口状态(最近一:	}钟)			
应用名称	接	口名称				全郎调用↓	平均耗时
otlp-server	/he	llo				21.9K cpm	0.017 ms
otlp-server	/he	llo				20.5K cpm	0.017 ms
otlo-convor	/he	llo				5.155K cpm	0.017 ms

## 通过HTTP API获取APM Metrics

通过HTTP API在自建的Graf ana接入ARMS OpenTelemetry Collector上报的APM Metrics,具体使用方式, 请参见HTTP API地址使用说明。

# 5.10. ARMS OpenTelemetry Collector配 置说明

ARMS OpenTelemetry Collector (简称OT Collector)支持集群粒度的动态配置下推,无需重启Collector服务即可按需调整多种参数或开关状态。

### 功能入口

- 1. 登录ARMS控制台。
- 2. 在左侧导航栏,选择应用监控 > 全局配置,然后在右侧页面单击OT Collector配置页签。
- 3. 在顶部菜单栏,选择地域。
- 4. 选择OT Collector归属集群对应的ClusterId。
- 5. 设置自定义配置参数,设置完毕后,在页面底部单击保存。

#### 调用链设置



- 是否采集调用链:控制OT Collector调用链Trace数据的上报,默认开启。手动关闭后将不再上报Trace明 细数据,请谨慎操作。
- 调用链预聚合开关:控制OT Collector端Trace转Metrics预聚合处理,默认开启。手动关闭后将不再执行 预聚合操作,指标数据不可见。

- 调用链采样率:控制调用链Trace上报的采样率,默认为100%上报,取值范围为0.01%~100%,OT Collector只会将符合采样率的Trace数据上报至服务端进行持久化存储。不影响预聚合指标准确性。
- 调用链流量限额:控制每个OT Collector节点上报的Trace最大请求数(按Span计数),默认为10000条/ 秒,超过该阈值的Trace数据将不再上报。不影响预聚合指标准确性。

#### 接口过滤与收敛

为了避免无效接口或发散接口对指标数据的影响,ARMS OpenTelemetry Collector 支持对SpanName接口 进行过滤与收敛。每一条过滤与收敛规则可以设置应用范围,指定部分应用生效(多个应用以英文逗号(,) 分隔),或指定*\$all_app*对全部应用生效。

无效接口过滤	添加无效接口过滤			
匹配方式	匹配内容 ⑦	应用范围(应用名称列表) ⑦	是否启用	操作
后缀匹配	.jpg,.png,.js,.jpeg	\$all_app		编辑 删除
发散接口收敛	添加发散接口收敛			
收敛正则 ⑦		应用范围 (应用名称列表) ②	是否启用	操作
/service/demo	((*)	\$all_app		编辑删除

# 无效接口过滤

接口过滤针对SpanName字段生效,支持前缀匹配、后缀匹配和完全匹配三种方式。默认过滤.jsp、.png、.js和.jpeg结尾的SpanName,不进行预聚合。

## 发散接口收敛

某些SpanName包含的变量会导致接口名称发散严重,需要对其进行收敛操作。例 如/service/demo/123456最后一位是不断变化的用户ID,可以对其添加收敛规则/service/demo/(.*)将发散 接口收敛成一个接口,便于监控与告警。

### 标签管理

ARMS OpenTelemetry Collector根据Trace预聚合的APM Metrics指标,除了少量系统标签外,您可以根据自身场景需要进行灵活的标签扩展,主要支持以下三类标签:资源标签、应用组标签和自定义标签。

### 资源标签

资源标签被大量应用于跨类型可观测数据关联或控制台页面交互。因此,资源标签的Key都是固定的,不允 许修改,只能选择启用或关闭。

目前,资源标签支持四种,全部默认开启。其中,区域和K8s集群ID标签由OT Collector自动赋值。环境和版本标签需要您在应用SDK中添加相应的Attributes才会生效。

资源标签 ②			
标签名称	标签 key	标篮 value (示例)	是否启用
区域	regionId	根据 Satellite 所在区域自动获取,例如 cn-hangzhou	
K8S集群ID	clusterid	根据 Satellite 所在集群自动获取其 K8S 集群 ID	
环境	armsEnv	提取调用链 Attributes 中的 armsEnv 李段值,例如 daily	
版本	armsVersion	提取调用链 Attributes 中韵 armsVersion 字段值,例如 v1.0.0	

# 应用组标签

应用组标签可以将多个应用归并到同一个分组,进行跨应用联合视图统计。例如设置一个交易分组trade,将涉及交易的应用都打上此标签,从而观察整个交易分组的数据变化。
应用组标签 添加应用组织	×.			
标签名称	标签 key	标签 value	应用范围 (应用名称列表) ②	是否启用 操作
交易城	bizDomain	trade	buy,order,goods,pay	编辑 删除

# 自定义标签

您还可以从Trace数据的Attributes中自由提取自定义标签,例如用户类型,经销渠道,产品类目等。

自定义标签 ⑦ 添加自定义	标签		
标签名称	标签 key	应用范围 (应用名称列表) ⑦	是否启用 操作
userType	userType	\$all_app	刻語 新聞

# 5.11. 应用监控指标说明

本文介绍了ARMS应用监控中常见的指标说明。

# 业务类指标

# 公共维度

维度名称	维度Key
服务名称	service
服务PID	pid
机器IP	serverlp
接口	rpc

## 指标列表

所有访问类型都包含下列指标,执行查询操作时,只需要将 \$callType 替换为具体的访问类型即可。详细的访问类型,请参见服务访问类型及可用维度。

例如:需要查询HTTP服务的请求数时,只需将 arms_\$callType_requests_count 修改

为 arms_http_requests_count 。

指标名称	指标	指标类型	采集间隔	单位	维度
总请求数	arms_\$callType_request s_count	Gauge	15s	无	
错误请求 数	arms_\$callType_request s_error_count	Gauge	15s	无	
总请求耗 时	arms_\$callType_request s_seconds	Gauge	15s	秒	不同服务访问类型维度不 同 详细信息 请参见服
慢请求数	arms_\$callType_request s_slow_count	Gauge	15s	无	回, F细语息, 请参见服 务访问类型及可用维度。

#### 应用实时监控服务 ARMS

指标名称	指标	指标类型	采集间隔	单位	维度
总请求耗 时分位数	arms_\$callType_request s_latency_seconds	Summary	15s	秒	<ul> <li>仅当服务访问类型为HTTP 且开启分位数统计的情况 下存在。 quantile分数位维度:</li> <li>0.5</li> <li>0.75</li> <li>0.90</li> <li>0.99</li> </ul>

# JVM指标 公共维度

维度名称	维度Key
服务名称	service
服务PID	pid
机器IP	serverlp

# 指标列表

指标名称	指标	指标类型	采集间隔	单位	维度
累计GC发 生次数	arms_jvm_gc_total	Counter	15s	无	
累计GC耗 时	arms_jvm_gc_seconds_t otal	Counter	15s	秒	
两次采集 间隔之间 的GC次数	arms_jvm_gc_delta	Gauge	15s	无	
两次采集 间隔之间 的GC耗时	arms_jvm_gc_seconds_ delta	Gauge	15s	秒	gen gc发生区域:

• old: 老年代

#### 应用监控·参考信息

#### 应用实时监控服务 ARMS

指标名称	指标	指标类型	采集间隔	单位	维度
JVM线程数	arms_jvm_threads_coun t	Gauge	15s	无	state线程状态: blocked:阻塞状态 live:活跃状态 daemon:守护状态 new:初始状态 dead-lock:死锁状态 runnable:运行状态 terminated:终止状态 timed-wait:超时等待 状态 wait:等待状态
JVM内存区 域初始大 小	arms_jvm_mem_init_byt es	Gauge	15s	字节	area区域: • heap: 堆区
JVM内存区 域最大大 小	arms_jvm_mem_max_by tes	Gauge	15s	字节	<ul> <li>nonheap:非堆区</li> <li>total:总计</li> <li>id区域细分:</li> <li>eden:伊甸区</li> <li>old:老年代</li> <li>survivor:幸存者区</li> <li>metaspace:元数据区</li> <li>code cache:代码缓存区</li> <li>compressed class</li> </ul>
JVM内存区 域使用大 小	arms_jvm_mem_used_b ytes	Gauge	15s	字节	
JVM内存区 域已提交 大小	arms_jvm_mem_commit ted_bytes	Gauge	15s	字节	
JVM内存区 域使用比 例	arms_jvm_mem_usage_r atio	Gauge	15s	比例 (0~1)	space • total: 总计
JVM已加载 类	arms_class_load_loade d	Counter	15s	无	无
JVM已卸载 类	arms_class_load_un_loa ded	Counter	15s	无	无
JVM缓存池 大小	arms_jvm_buffer_pool_t otal_bytes	Gauge	15s	字节	
JVM缓存池 已使用大 小	arms_jvm_buffer_pool_ used_bytes	Gauge	15s	字节	id区域:
JVM缓存池 个数	arms_jvm_buffer_pool_ count	Gauge	15s	无	<ul><li> direct</li><li> mapped</li></ul>

#### 应用实时监控服务 ARMS

指标名称	指标	指标类型	采集间隔	单位	维度
文件描述 符打开个 数	arms_file_desc_open_co unt	Gauge	15s	无	无
文件描述 符打开比 例(已打 开数/最大 允许打开 数)	arms_file_desc_open_ra tio	Gauge	15s	比例 (0~1)	无

# 系统指标 公共维度

维度名称	维度Key
服务名称	service
服务PID	pid
机器IP	serverlp

# 指标列表

指标名称	指标	指标类型	采集间隔	单位
空闲CPU占比	arms_system_cpu _idle	Gauge	15s	百分数
IO等待CPU占比	arms_system_cpu _io_wait	Gauge	15s	百分数
系统CPU占比	arms_system_cpu _system	Gauge	15s	百分数
用户态CPU占比	arms_system_cpu _user	Gauge	15s	百分数
系统负载(1分钟)	arms_system_loa d	Gauge	15s	无
磁盘空闲大小	arms_system_disk _free_bytes	Gauge	15s	字节
磁盘总大小	arms_system_disk _total_bytes	Gauge	15s	字节
磁盘使用率	arms_system_disk _used_ratio	Gauge	15s	比例(0~1)

#### 应用监控·参考信息

指标名称	指标	指标类型	采集间隔	单位
内存Buffer大小	arms_system_me m_buffers_bytes	Gauge	15s	字节
内存缓存大小	arms_system_me m_cached_bytes	Gauge	15s	字节
内存空闲大小	arms_system_me m_free_bytes	Gauge	15s	字节
内存交换区空闲大 小	arms_system_me m_swap_free_byt es	Gauge	15s	字节
内存交换区大小	arms_system_me m_swap_total_by tes	Gauge	15s	字节
内存大小	arms_system_me m_total_bytes	Gauge	15s	字节
已用内存大小	arms_system_me m_used_bytes	Gauge	15s	字节
网络接收流量大小	arms_system_net_ in_bytes	Gauge	15s	字节
网口发送流量大小	arms_system_net_ out_bytes	Gauge	15s	字节
网络入口错误数	arms_system_net_ in_err	Gauge	15s	无
网络出口错误数	arms_system_net_ out_err	Gauge	15s	无

# 线程池指标 公共维度

维度名称	维度Key
服务名称	service
服务PID	pid
机器IP	serverlp
线程池名称	name
线程池类型	type

# 指标列表

#### 应用实时监控服务 ARMS

指标名称	指标	指标类型	采集间隔	维度
线程池核心线程数	arms_threadpool_ core_size	Gauge	15s	无
线程池最大线程数	arms_threadpool_ max_size	Gauge	15s	无
线程池活跃线程数	arms_threadpool_ active_size	Gauge	15s	无
线程池队列大小	arms_threadpool_ queue_size	Gauge	15s	无
线程池当前大小	arms_threadpool_ current_size	Gauge	15s	无
线程池不同状态任 务数	arms_threadpool_ task_total	Gauge	155	<ul> <li>status:任务状态</li> <li>scheduled:已调度</li> <li>completed:完成</li> <li>rejected:拒绝</li> </ul>

# 服务访问类型及可用维度 客户端类

- 访问类型
  - http_client
  - hsf_client
  - mq_client
  - kaf ka_send
  - notify_client
  - dubbo_client
  - grpc_client
  - thrift_client
- 维度
  - parent:上游服务的名称
  - ppid: 上游服务PID
  - destId:请求对端扩展信息
  - endpoint:请求对端地址
  - excepType: 异常Ⅳ
  - excepInfo: 异常ID编码规则
  - stackTraceld:异常栈ID

## DB类

- 访问类型
  - mysql
  - oracle
  - mariadb
  - postgresql
  - ppas
  - sqlserver
  - mongodb
  - dmdb
- 维度
  - parent:上游服务的名称
  - ppid: 上游服务PID
  - destId: 数据库名称
  - endpoint:数据库地址
  - excepType: 异常D
  - excepInfo: 异常ID编码规则
  - stackTraceld: 异常栈ID
  - sqlld: SQL语句ID

## 服务端类

- 访问类型
  - ∘ http
  - hsf
  - $\circ$  dubbo
  - user_method
  - grpc
  - ∘ thrift
- 维度
  - prpc: 上游接口
  - parent:上游服务的名称
  - ppid: 上游服务PID
  - endpoint: 服务地址
  - excepType: 异常D
  - excepInfo: 异常ID编码规则
  - stackTraceld:异常栈ID

# 客户端类

- 访问类型
  - o xxl

- schedulerx
- 维度
  - o prpc: 上游接口
  - parent:上游服务的名称
  - ppid: 上游服务PID

# 6.升级探针

对于在ARMS中监控的EDAS应用、容器服务应用和其他类型应用,请按照本文所述的相应方法更新Java探针版本。

## 如何为EDAS应用更新探针?

如果您需要更新EDAS应用的Java探针版本,重新部署EDAS应用即可。

#### 如何为容器服务应用更新探针?

如果您需要更新容器服务应用的Java探针版本,重启应用Pod即可。

# 如何为其他类型的应用更新探针?

如果您需要更新除了EDAS应用和容器服务应用之外的其他类型应用的Java探针版本,先卸载探针再重新安装 即可。

如需卸载探针,请参见卸载Agent的常见问题。

# 7.应用监控常见问题

本章节汇总了使用ARMS应用监控时的常见问题。

## 本页目录

- 为Java应用手动安装Agent的常见问题
  - ◎ ARMS Agent和其他APM产品Agent(例如Pinpoint)是否兼容?
  - 。 安装Agent后应用启动时报Out Of MemoryError错误怎么办?
  - 如何测试网络连通性?
  - o 如何检查ARMS Agent是否安装成功?
  - ARMS Agent 安装成功,为什么控制台上仍无监控数据?
  - ARMS Agent 安装成功,但IP显示不正确或不显示怎么办?
  - 对于ARMS Agent 日志(路径: ArmsAgent / log)错误,有哪些常用排查方法?
  - o 如何支持单机多实例?
- 为Java应用快速安装Agent的常见问题
  - 运行接入脚本时出现getcwd相关错误,该如何解决?
  - 使用一键接入方式安装Agent后,在哪里查看日志?
- 为ECS中的Java应用快速安装Agent的常见问题
  - Agent安装失败怎么处理?
  - 安装Agent后ECS实例进程信息不准确怎么办?
  - ECS实例中的进程启动应用监控失败怎么办?
- 为容器服务Kubernetes版Java应用安装Agent的常见问题
  - 为什么容器服务K8s集群中JDK11版本的Java应用安装Agent后,应用监控没有数据呢?
  - o 为什么容器服务K8s集群中的Java应用安装Agent后,应用监控没有数据呢?
  - o 容器服务K8s集群中的应用安装Agent失败怎么处理?
- 为开源Kubernetes环境中的Java应用安装Agent常见问题
  - 为什么开源K8s集群中JDK11版本的Java应用安装Agent后,应用监控没有数据呢?
  - 应用无法正常启动怎么办?
  - o 如何查看Agent日志?
- 在不重装Agent的情况下更改Java应用名称的常见问题
  - 以通用方式安装Agent的普通Java应用如何更改应用名称?
  - 部署在容器服务K8s集群中的Java应用如何更改应用名称?
  - 部署在EDAS上的Java应用如何更改应用名称?
- 卸载Agent的常见问题
  - o 如何卸载以通用方式安装的Agent?
  - 。 如何卸载以快速方式安装的Agent?
  - 如何卸载ECS中Java应用的Agent?
  - 如何卸载容器服务K8s集群中Java应用的Agent?
  - 如何卸载开源K8s环境中Java应用的Agent?

- 。 如何卸载Docker中Java应用的Agent?
- 其他问题
  - 使用OpenFeign组件的应用在ARMS中数据不完整怎么办?

#### ARMS Agent和其他APM产品Agent(例如Pinpoint)是否兼容?

ARMS Agent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,而不同商家的插装代码实现不同,代码冲突可能造成严重的性能问题,因此强烈建议您不要同时安装多个APM Agent。

#### [回到顶部]

#### 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。

⑦ 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
```

#### [回到顶部]

#### 如何测试网络连通性?

部署ARMS Agent时,必须确保8883、8443、8442这三个端口的可连通性,Agent才能正常工作。可使用Telnet命令测试目标主机与ARMS服务器网络是否连通。例如,以测试深圳地域的连通性为例,请登录应用所部署的机器,并输入以下命令:

```
telnet arms-dc-sz.aliyuncs.com 8883
telnet arms-dc-sz.aliyuncs.com 8443
telnet arms-dc-sz.aliyuncs.com 8442
```

⑦ 说明 请根据所在地域替换服务器地址,且服务器地址区分经典网络和公网与VPC网络。

各地域ARMS应用监控服务器地址

地域	经典网络和公网	VPC网络
华东1(杭州)	arms-dc-hz.aliyuncs.com	arms-dc-hz-internal.aliyuncs.com
华东2(上海)	arms-dc-sh.aliyuncs.com	arms-dc-sh-internal.aliyuncs.com
华北1(青岛)	arms-dc-qd.aliyuncs.com	arms-dc-qd-internal.aliyuncs.com
华北2(北京)	arms-dc-bj.aliyuncs.com	arms-dc-bj-internal.aliyuncs.com
华北3(张家口)	arms-dc-zb.aliyuncs.com	arms-dc-zb-internal.aliyuncs.com
华北5(呼和浩特)	dc-cn-huhehaote.arms.aliyuncs.com	dc-cn-huhehaote- internal.arms.aliyuncs.com
华北6(乌兰察布)	dc-cn-wulanchabu.arms.aliyuncs.com	dc-cn-wulanchabu- internal.arms.aliyuncs.com
华南1(深圳)	arms-dc-sz.aliyuncs.com	arms-dc-sz-internal.aliyuncs.com
华南2(河源)	dc-cn-heyuan.arms.aliyuncs.com	dc-cn-heyuan- internal.arms.aliyuncs.com
华南3(广州)	dc-cn-guangzhou.arms.aliyuncs.com	dc-cn-guangzhou- internal.arms.aliyuncs.com
西南1(成都)	dc-cn-chengdu.arms.aliyuncs.com	dc-cn-chengdu- internal.arms.aliyuncs.com
中国(香港)	arms-dc-hk.aliyuncs.com	arms-dc-hk-internal.aliyuncs.com
亚太东南1(新加坡)	arms-dc-sg.aliyuncs.com	arms-dc-sg-internal.aliyuncs.com
亚太东南2(悉尼)	dc-ap-southeast-2.arms.aliyuncs.com	dc-ap-southeast-2- internal.arms.aliyuncs.com
亚太东南3(吉隆坡)	dc-ap-southeast-3.arms.aliyuncs.com	dc-ap-southeast-3- internal.arms.aliyuncs.com
亚太东南5(雅加达)	arms-dc-indonesia.aliyuncs.com	arms-dc-indonesia- internal.aliyuncs.com
亚太东北1(东京)	arms-dc-jp.aliyuncs.com	arms-dc-jp-internal.aliyuncs.com
欧洲中部1(法兰克福)	arms-dc-frankfurt.aliyuncs.com	arms-dc-frankfurt- internal.aliyuncs.com
欧洲西部1(伦敦)	dc-eu-west-1.arms.aliyuncs.com	dc-eu-west-1- internal.arms.aliyuncs.com
美国东部1(弗吉尼亚)	dc-us-east-1.arms.aliyuncs.com	dc-us-east-1- internal.arms.aliyuncs.com

地域	经典网络和公网	VPC网络
美国西部1(硅谷)	arms-dc-usw.aliyuncs.com	dc-us-west-1- internal.arms.aliyuncs.com
亚太南部1(孟买)	dc-ap-south-1.arms.aliyuncs.com	dc-ap-south-1- internal.arms.aliyuncs.com
政务云	arms-dc-gov.aliyuncs.com	arms-dc-gov-internal.aliyuncs.com
杭州金融云	arms-dc-hz-finance.aliyuncs.com	arms-dc-hz-finance- internal.aliyuncs.com
上海金融云	arms-dc-sh-finance.aliyuncs.com	arms-dc-sh-finance- internal.aliyuncs.com
深圳金融云	arms-dc-sz-finance.aliyuncs.com	arms-dc-sz-finance- internal.aliyuncs.com

[回到顶部]

#### 如何检查ARMS 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ARMS Agent。

使用PS叩マ旦有叩マ11石砌参数中定百成功女表ARM5 Age

ps -ef | grep 'arms-bootstrap'

成功安装时,如下图所示:

→ ~ git:(master) ¥ ps -ef   grep 'arms-bootstrap'
501 15998 1617 0 7:52下午 ?? 0:25.84 /Library/Java/Java/JitualMachines/jdk1.8.0_131.jdk/Contents/Home/bin/java -agentli
b:jdwp=transport_dt_ <u>socket_address=127_0_0_1:62142_su</u> spend=y,server=n_javaagent:/Users//debug/oneagent/agent/target/arms-agen
t-1.7.0-SNAPSHO <mark>l/arms-bootstrap</mark> -1.7.0-SNAPSHOT.jar -Qarms.licenseKey=F_arms_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans_trans
dStopAtLevel=1 -noverity -Uspring.output.ansi.enabled=always -Dcom.sun.management.jmxremote -Dspring.liveBeansView.mbeanDomain -Dspri
ng.application.admin.enabled=true -javaagent:/Users//Library/Caches/IntelliJIdea2018.3/captureAgent/debugger-agent.jar -Dfile.
and the UTE 0 - 3

命令行中的Darms.licenseKey的值必须与ARMS接入应用页面显示的License Key一致。

☰ (-) 阿里云	
<	據入应用
接入应用	请先选择您使用的语言 License Key:
	Java PHP C C++ P Go NET NET Node Node js

#### [回到顶部]

# ARMS Agent安装成功,为什么控制台上仍无监控数据?

- 1. 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访问 请求,包括HTTP请求、HSF请求和Dubbo请求,并确认开发框架是否在ARMS Agent的支持范围内。关 于ARMS应用监控对第三方组件和框架的支持情况,请参见应用监控概述。
- 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15分钟,然后再次确认是否有监控数据。
- 3. 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。以下是一个正确示例。

java -javaagent:/{user.workspace}/ArmsAgent/arms-bootstrap-1.7.0-SNAPSHOT.jar -Darms.li censeKey=xxx -Darms.appName=xxx -jar demoApp.jar

- 4. 如果*ArmsAgent /log/*下的日志中出现LicenseKey is invalid.异常,请检查应用所属地域与Agent所属地 域是否一致。
- 5. 如果应用启动之后*ArmsAgent*目录下没有log子目录,则表明*arms-bootstrap-1.7.0-SNAPSHOT.jar*未被成功加载,请检查ArmsAgent安装目录的权限是否正确。
- 6. 若应用启动时出现以下报错,请您检查arms-bootstrap-1.7.0-SNAPSHOT.jar软件包及相应权限是否正确。

```
Error opening zip file or JAR manifest missing: /root/ArmsAgent/arms-bootstrap-1.7.0-SN
APSHOT.jar
Error occurred during initialization of VM
agent library failed to init: instrument
```

- 7. 如果仍无监控数据,请将Java Agent日志(路径: *ArmsAgent/log*)保存为压缩文件,并联系钉钉服务 账号 arms160804 解决问题。
- 8. 检查JDK版本。如果JDK版本为1.8.0_25或者1.8.0_31,可能会出现无法安装Agent的情况,建议您升级对 应的JDK版本,或咨询钉钉服务账号 arms160804 。

#### [回到顶部]

#### ARMS Agent安装成功,但IP显示不正确或不显示怎么办?

- 1. 首先通过if config -a命令确认当前机器的网络配置情况,若为多网卡环境,则Agent采集到的IP地址可能会不符合预期(与网络配置环境有关)。
- 2. 选择以下方式之一来解决问题。
  - 配置JVM的 -DEAGLEEYE.LOCAL.IP=10.XX.XX.XX 参数。

⑦ 说明 将 10.xx.xx.xx 替换为实际IP地址。

• 配置Agent获取JVM的 -DNETWORK.INTERFACE=eth0 参数,其中 eth0 为网卡名称。

#### [回到顶部]

# 对于ARMS Agent日志(路径: ArmsAgent/log)错误,有哪些常用排查方法?

LicenseKey is invalid错误:

- 1. 确保在ARMS中已成功创建该应用,并且确认安装Agent时填写的LicenseKey正确。
- 2. 由于ARMS产品是多地域(Region)环境,所以还需要确认Agent的下载地址所属的地域与应用所属地 域是否一致。

#### [回到顶部]

#### 如何支持单机多实例?

如需在同一台机器上部署同一应用的多个实例,可以通过-Darms.agentId(逻辑编号,如001、002)参数 来区分接入的JVM进程,例如:

java -javaagent:/{user.workspace}/ArmsAgent/arms-bootstrap-1.7.0-SNAPSHOT.jar -Darms.licens eKey=<LicenseKey> -Darms.appName=<AppName> -Darms.agentId=001 -jar demoApp.jar

#### [回到顶部]

# 运行接入脚本时出现getcwd相关错误,该如何解决?

若您在运行一键接入Java应用的接入脚本时遇到以下错误:

shell-init: error retrieving current directory: getcwd: cannot access parent directories: N o such file or directory Error occurred during initialization of VM java.lang.Error: Proper ties init: Could not determine current working directory. at java.lang.System.initPropertie s(Native Method) at java.lang.System.initializeSystemClass(System.java:1119)

可能是因为执行脚本过程中当前目录被误删,解决方法为执行 cd 后重新运行脚本。

#### [回到顶部]

#### 使用一键接入方式安装Agent后,在哪里查看日志?

日志的默认目录为/root/.arms/supervisor/logs/,若此目录下没有日志,请执行命令 ps -ef |grep arms 查看日志所在目录。

#### [回到顶部]

#### Agent安装失败怎么处理?

1. 确保您的ECS实例可以访问公网, 且能够访问所在地域的Agent下载链接。

地域	公网地址
华东1(杭州)	arms-apm-cn-hangzhou.oss-cn-hangzhou.aliyuncs.com/ArmsAgent.zip
华东2(上海)	arms-apm-cn-shanghai.oss-cn-shanghai.aliyuncs.com/ArmsAgent.zip
华北1 (青岛)	arms-apm-cn-qingdao.oss-cn-qingdao.aliyuncs.com/ArmsAgent.zip
华北2(北京)	arms-apm-cn-beijing.oss-cn-beijing.aliyuncs.com/ArmsAgent.zip
华北3(张家口)	arms-apm-cn-zhangjiakou.oss-cn-zhangjiakou.aliyuncs.com/ArmsAgent.zip
华北5(呼和浩特)	arms-apm-cn-huhehaote.oss-cn-huhehaote.aliyuncs.com/ArmsAgent.zip
华北6(乌兰察布)	arms-apm-cn-wulanchabu.oss-cn-wulanchabu.aliyuncs.com/ArmsAgent.zip
华南1(深圳)	arms-apm-cn-shenzhen.oss-cn-shenzhen.aliyuncs.com/ArmsAgent.zip
华南2(河源)	arms-apm-cn-heyuan.oss-cn-heyuan.aliyuncs.com/ArmsAgent.zip
华南3(广州)	arms-apm-cn-guangzhou.oss-cn-guangzhou.aliyuncs.com/ArmsAgent.zip
西南1(成都)	arms-apm-cn-chengdu.oss-cn-chengdu.aliyuncs.com/ArmsAgent.zip
中国 (香港)	arms-apm-cn-hongkong.oss-cn-hongkong.aliyuncs.com/ArmsAgent.zip
亚太东南1(新加坡)	arms-apm-ap-southeast-1.oss-ap-southeast-1.aliyuncs.com/ArmsAgent.zip
亚太东南2(悉尼)	arms-apm-ap-southeast-2.oss-ap-southeast-2.aliyuncs.com/ArmsAgent.zip

地域	公网地址
亚太东南3(吉隆坡)	arms-apm-ap-southeast-3.oss-ap-southeast-3.aliyuncs.com/ArmsAgent.zip
亚太东南5(雅加达)	arms-apm-ap-southeast-5.oss-ap-southeast-5.aliyuncs.com/ArmsAgent.zip
亚太东北1(东京)	arms-apm-ap-northeast-1.oss-ap-northeast-1.aliyuncs.com/ArmsAgent.zip
欧洲中部1(法兰克福)	arms-apm-eu-central-1.oss-eu-central-1.aliyuncs.com/ArmsAgent.zip
欧洲西部1(伦敦)	arms-apm-eu-west-1.oss-eu-west-1.aliyuncs.com/ArmsAgent.zip
美国东部1(弗吉尼亚)	arms-apm-us-east-1.oss-us-east-1.aliyuncs.com/ArmsAgent.zip
美国西部1(硅谷)	arms-apm-us-west-1.oss-us-west-1.aliyuncs.com/ArmsAgent.zip
亚太南部1(孟买)	arms-apm-ap-south-1.oss-ap-south-1.aliyuncs.com/ArmsAgent.zip
政务云	arms-apm-cn-north-2-gov-1.oss-cn-north-2-gov- 1.aliyuncs.com/ArmsAgent.zip

- 2. 确保您的ECS实例可以访问控制台。
  - o 中国ARMS控制台
  - o 新加坡ARMS控制台
- 3. 登录ECS控制台,并完成以下检查工作。
  - i. 在左侧导航栏中选择运维与监控 > 发送命令/文件(云助手)。
  - ii. 在ECS云助手页面命令列表页签的搜索框中选择命令名称,输入 InstallJavaAgent 命令并按 Enter键。

ECS云助手						
<ul> <li>ABR. ARRN. RECHER, REPORT (5)</li> </ul>	el, Powershell和Bat) 和波道文件。更多信息					×
会議/法行命令         发送文件           命令予決         余令执行地策         文件发送地策						N <b>B</b> G∏®∳
Q. ####  MARGENER			0			C RM
m@ID/名称	163	日本日間	100/10	913823140	操作	
cs45nuschargzhou		Linux / Shell	systemati stop kubelet.service; system	2020年11月25日 22:02:00	3517   95年   1096	Î
edatEsthTask		Linux / Shell	(content)	2020年9月9日 10:11:00	助行   四禄   副除	

⑦ 说明 若查找结果不存在,请联系ARMS钉钉服务账号 arms160804 。

iii. 在ECS云助手页面的命令执行结果页签的搜索栏输入 InstallJavaAgent 命令的命令ID,在查找 结果中单击该记录右侧操作列的查看,即可查看 InstallJavaAgent 命令是否执行成功。若未执 行成功,请根据详细执行结果排查问题。如ECS磁盘满、未安装Agent等问题,可以通过清理磁盘 或安装Agent来解决。若不能解决,请将详细执行结果反馈给ARMS钉钉服务账号 arms160804 。

897(8 <b>89</b> (5)	<b>行结果</b> 文件发送结果								
Q BRREED		変形の電圧環境度、耐い液素環内中全の							○ 用新
执行状态	命令执行的	命令(0/梁称	命令例型	<b>命令内容</b>	的關約時間	1668	周期性执行	操作	4 1
✓鉄行発成		edas-agent-installation-v4	Shell	ed /root; F+ installah) UF.,	2020年11月20日 15:37:47	总数 1 完成 1	-	遊春   寺出	î
✓供行殉政		ping	Shell	test -f '/uso'bin/docker' &	2020年11月20日 15:37:40	息数 1 完成 1		遊春 号出	

[回到顶部]

#### 安装Agent后ECS实例进程信息不准确怎么办?

若您在ECS上成功安装Agent后没有出现ECS实例进程信息,或者ECS实例进程信息不准确,请单击ECS实例 左侧的-号并单击+号刷新数据。若不能解决,请联系ARMS钉钉服务账号 arms160804 。

#### [回到顶部]

#### ECS实例中的进程启动应用监控失败怎么办?

请检查ECS实例的 /root/.arms/supervisor/logs/arms-supervisor.log 日志中是否有日志级别为Error的 信息,并根据此信息排查。若不能解决,请联系ARMS钉钉服务账号 arms160804 。

#### [回到顶部]

# 为什么容器服务K8s集群中JDK11版本的Java应用安装Agent后,应用监控没 有数据呢?

可能原因:应用的YAML文件中未添加JDK版本参数。

解决方案:执行以下步骤在应用的YAML文件中添加JDK版本参数。

- 1. 登录容器服务管理控制台。
- 2. 在左侧导航栏选择集群,在集群列表页面上的目标集群右侧操作列单击应用管理。
- 3. 在无状态页面顶部选择您的应用所在的命名空间,然后在目标应用右侧选择更多 > 查看Yaml。
- 4. 在编辑YAML对话框的 spec > template > metadata > labels 层级下添加以下参数。

one-agent.jdk.version: "OpenJDK11"



#### [回到顶部]

为什么容器服务K8s集群中的Java应用安装Agent后,应用监控没有数据呢?

- 1. 登录容器服务管理控制台。
- 2. 在左侧导航栏选择集群,在集群列表页面上的目标集群右侧操作列单击应用管理。
- 3. 在容器组页签顶部选择您的应用所在的命名空间,然后单击目标应用右侧单击编辑。
- 4. 在编辑YAML对话框中查看YAML文件中是否存在initContainers。

94     name: default-token-5cm96       95     readOnly: true       96 *     - mountPath: /home/admin/.opt/       97     name: arms-agent-vol       98 *     - mountPath: /ctc/podinfo
96 - nountPath: /home/admin/.opt/ 97 name: arms-agent-vol 98 - mountPath: /etc.podinfo
97 name: arms-agent-vol 98 - mountPath: /etc/podinfo
98 mountPath: /etc/podinfo
99 name: arms-podinfo
100 mountPath: /opt/arms-debug/share
101 name: volume-tmp-share
102 dnsPolicy: ClusterFirst
103 enableServiceLinks: true
104 V initContainers:
105 - args:
106 - '-java=true'
10/ - '-pid=
108 env:
109 - name: AKMS_PILOI_INI_KEGIONID
110 Value: cn-nangznou
111 Indge: >-
112 registry-pet.ch-haigznot.airyuncs.com/arms-uocker-repo/arms-piiot-init.vi.o.i
115 Independent of the interstation
114 name: arms-priot-inficientariner
116 resources. ()
117 termination/accore/acity_File
117 terminactomessagerority. File
110 volumenoutits.
120 nome: arms_agent_vol
121 nodeName: cn-bangzhou
122 priority: 0
123 restartPolicy: Always
124 schedulerName: default-scheduler
125 securityContext: {}
126 serviceAccount: default
127 serviceAccountName: default
128 terminationGracePeriodSeconds: 30
更新 下载 另存为 取消

- 如果不存在,则说明未被注入arms-init-container,执行步骤5。
- 如果存在,则说明已被注入arms-init-container,执行步骤8。
- 5. 在容器组页签顶部选择命名空间为ack-onepilot。查看Pod列表中是否存在名称前缀为ack-onepilot的Pod。
  - 如果存在,则执行步骤6。
  - ◎ 如果不存在,则在应用市场中安装ack-onepilot。具体操作,请参见安装ARMS应用监控组件。
- 6. 在无状态或有状态页签目标应用右侧操作列中选择更多 > 查看Yaml, 在编辑YAML对话框查看YAML 文件中是否存在以下Labels注解。

```
labels:
    armsPilotAutoEnable: "on"
    armsPilotCreateAppName: "<your-deployment-name>" //请将<your-deployment-name>替换为
您的应用名称。
    one-agent.jdk.version: "OpenJDK11" //如果应用的JDK版本是JDK11,则需要配置此参数。
```

- 如果存在,则执行步骤7。
- 如果不存在,则在编辑YAML对话框中的*spec.template.metadata*层级下添加以上Labels注解,然后 单击更新。
- 7. 在容器组页签目标应用右侧单击日志,查看ack-onepilot的Pod日志是否报STS错误,即提示 "Messag e":"STS错误"。
  - 如果报STS错误,则需为应用所在集群授权,并重启应用所在Pod。具体操作,请参见为容器服务 Kubernetes版授权。
  - 如果未报STS错误,则联系ARMS钉钉服务账号: arms160804。
- 8. 在容器组页签目标应用右侧单击编辑,在编辑YAML对话框中查看YAML文件中是否存在以 下javaagent参数。

-javaagent:/home/admin/.opt/ArmsAgent/arms-bootstrap-1.7.0-SNAPSHOT.jar

○ 如果存在,则单击容器组页签右侧的终端进入命令行页面,执行以下命令查看是否存在以.log为后缀

的日志文件,然后联系ARMS钉钉服务账号: arms160804。

cd /home/admin/.opt/ArmsAgent/logs

○ 如果不存在,则联系ARMS钉钉服务账号: arms160804。

#### [回到顶部]

容器服务K8s集群中的应用安装Agent失败怎么处理?

可能原因:目标集群中不存在ARMS Addon Token。

解决方案:

步骤一: 查看集群是否存在ARMS Addon Token。

- 1. 登录容器服务管理控制台,在集群列表页面,单击目标集群名称进入集群详情页。
- 2. 在左侧导航栏选择配置管理 > 保密字典。
- 3. 在顶部选择命名空间为kube-system, 查看addon.arms.token是否存在。

所有集群	/ 集群: 📰 🔳 -01 / 命名空的	间: kube-system 🗸 o / 保密	字典			
保密	客典 Secret				创建使用YAML创建资	æ
名称	✓ Q 请输入					Ó
	名称	标签	类型	创建时间	操	ſF
	ack-		kubernetes.io/	2021年6月16日 11:41:40	编辑 YAML编辑 删	除
	acr- 38173632e 77a2be46		kubernetes.io/	2021年6月16日 11:42:50	编辑   YAML编辑   删	除
	acr- bbd88cdc 91a39a86		kubernetes.io/	2021年6月16日 11:42:51	编辑   YAML 编辑   删	除
	acr- dcb4112157 e7ef6512		kubernetes.io/	2021年6月16日 11:42:50	编辑   YAML 编辑   删	除
	acr- 🗰 🖬 🖬 b628e2838 9a5eff36		kubernetes.io/ 🔳 🔳 🔳	2021年6月16日 11:42:49	编辑 YAML编辑 翻	除
	acr- ab48b6f52 d33a82		kubernetes.io/	2021年6月16日 11:42:52	编辑   YAML 编辑   删	R
	addon.arms.token		Opaque	2021年6月16日 11:35:26	编辑   YAML 编辑   删I	除

步骤二:为容器服务Kubernetes版授予ARMS资源的访问权限。

- 1. 登录容器服务管理控制台。
- 2. 在左侧导航栏选择集群,在集群列表页面,单击目标集群名称或其右侧操作列的详情。

	1818 • 🕑 ±18		Q IBRRE, IMME	s, api, Nirjinkon	8. 8	# I#	备账 企业	2 文持	吉岡 🖸	٥.	Ä	() ()	0
容器服务 - Kubernetes -	集群列表						<b>放石</b> 台	的推荐与节点者	- 805 -	lłf	1108980	080	4:29
概范	●●●●● ARMS 应用监控器和放重数没布,有研	计每天仅靠0.7元											
RET Country Will	88 v WE											@ N	和文档
Serveriess and	编制+新制/ID	经签 集制类型	地域 (全部) •	集制化态	节点数	使用量	创建时间	版本⑦					9901
▼ 市场	cd2856	<ul> <li>Nati</li> </ul>	48:152 (北州)	●运行中	3	c	2020-07-08 15:01:49	1.16.9-ally	un.1	3	<b>村橋</b> 皮用物	11月1日 - 11月1日 11月1日 - 11月1日 11月1日 - 11月1日 - 11月1日 11月1日 - 11月1日 - 11月1日 - 11月1日 - 11月1日 11月1日 - 11月1日 - 11月1日 - 11月1日 - 11月1日 - 11月1日 11月1日 - 11月1日 - 11月11日 - 11月111日 - 11月111日 - 11月111日 - 11月111日 - 11月1111 - 111111 - 111111 - 111111 - 111111	38 88 -

3. 在目标集群的集群信息页面上单击集群资源页签,然后单击Worker RAM角色右侧的链接。

	anan - E 211	Q HORIZAL ESHEL AFL, NORTHHOLIZ 👼 II IN MAR GAL ZIF NN 🖾 🎝 👾 🕐 NA 🍥
<	集群:	和約 #Bit Clourshell 验證#期
集群信息	▲ 概范 基本信息 近接信息 编数资源 集新日志	
日点管理	HIPMAN ROS	10;
工作负载	虚拟导教网络 VPC 节点虚拟交换机	чр- чж
服务	安全相	82-10-10-10-10-10-10-10-10-10-10-10-10-10-
路由	Worker RAM 角色	KubernetesWorkerRole
	Noire Indress SLR	2- martine - C. Santon and

- 4. 在访问控制RAM控制台的RAM角色管理页面上,单击权限管理页签上的权限策略名称。
- 5. 在策略内容页签上单击修改策略内容,并将以下内容添加到策略内容中,最后单击确定。

"	Antion II. Ioma MI
	ACCION: Mains: A.,
"F	Resource": "*",
	200 1 1 1 1 1
1	Sfiect": "Allow"
}	
,	
- 编	aga 1997 - 1997 - 1997 - 1997 - 1997 - 1997 - 1997 - 1997 - 1997 - 1997 - 1997 - 1997 - 1997 - 1997 - 1997 - 19
10	
● 您正	左使用个关设计的规范失败治理器,关系改为于21月空以关致再同楼目用,我们比学贵炯级的使用后续, 建生活我们级的相注
	在12月王新攻江的12股東始編編編,新版40月丁江日定又東始史易運易用。23月1年帝里规念的使用反映,第百杯34月念的想法。
可视化纲	扁辑 Beta 脚本编辑
策略文档	长度 924 个字符
30	"Effect": "Allow"
31	1
	1,
32 -	
32 • 33 • 34	7, { "Action": [
32 • 33 • 34 35	I. [ "Action": [
32 • 33 • 34 35 36	z, { "Action": [
32 * 33 * 34 35 36 37	z, f "Action": [
32 • 33 • 34 35 36 37 38	), {
32 * 33 * 34 35 36 37 38 39 *	), ( "Action": [ ], "Resource": [
32 * 33 * 34 35 36 37 38 39 * 40 41	), {
32 * 33 * 34 35 36 37 38 39 * 40 41 42	<pre> /,</pre>
32 * 33 * 34 35 36 37 38 39 * 40 41 42 43	<pre> // // // // // // // // // // // // //</pre>
32 * 33 * 34 35 36 37 38 39 * 40 41 42 43 44 *	<pre> / / / / / / / / / / / / / / / / / / /</pre>
32 * 33 * 34 35 36 37 38 39 * 40 41 42 43 44 * 45	<pre> /, / / / / / / / / / / / / / / / / / /</pre>
32 * 33 * 34 35 36 37 38 39 * 40 41 42 43 44 * 45 46 47	<pre> /, / / / / / / / / / / / / / / / / / /</pre>
32 * 33 * 34 35 36 37 38 39 * 40 41 42 43 44 * 45 46 47 48	<pre> } {</pre>
32 * 33 * 34 35 36 37 38 39 * 40 41 42 43 44 * 45 46 47 48 49	<pre> /, / / / / / / / / / / / / / / / / / /</pre>

#### [回到顶部]

# 为什么开源K8s集群中JDK11版本的Java应用安装Agent后,应用监控没有数 据呢?

可能原因:应用的YAML文件中未添加JDK版本参数。

解决方案: 在YAML文件中的 spec > template > metadata > labels 层级下添加以下内容。

one-agent.jdk.version: "OpenJDK11"

#### [回到顶部]

#### ACK集群如何跨区域上报数据?

- 1. 在容器集群arms-pilot Namespace下的Deployment中添加ARMS_REPORT_REGION环境变量,值为 ARMS所支持的RegionId(例如cn-hangzhou、cn-beijing)。
- 2. 重启现有应用或部署一个新的应用,完成跨区域上报。

⑦ 说明 当添加环境变量后,该集群下所有应用均会跨区域上报到步骤一中指定的地域。

# 应用无法正常启动怎么办?

请执行以下命令查看ack-onepilot的日志信息来排查问题。

```
k logs -f ack-onepilot-ack-onepilot-XXXX-XXXX -n ack-onepilot
```

⑦ 说明 XXXX-XXXX为随机信息,您可通过执行命令 k get pod -n ack-onepilot 来获取。

#### [回到顶部]

#### 如何查看Agent日志?

在Kubernetes集群Worker机上查看/home/admin/.opt/ArmsAgent/logs/xxxx.log。

#### [回到顶部]

#### 以通用方式安装Agent的普通Java应用如何更改应用名称?

普通Java应用是指除了部署在阿里云ECS实例上的应用以外的Java应用。如果您是以通用方式安装Agent的,则Agent目录就是您自定义的位置。

您可以在Agent目录的Version文件中查看Agent版本。例如, 2.5.8_cf020486_20190816150025 表示 Agent版本为2.5.8, Agent发布时间为2019年8月16日。

- 如果您的Agent版本低于2.5.8.1,请先卸载Agent再重新安装,并在重新安装时使用新的应用名称。
  - 手动安装的Agent的卸载方法,请参见如何卸载以通用方式安装的Agent。
  - 。 以脚本自动安装的Agent的卸载方法,请参见如何卸载以快速方式安装的Agent。
  - 。 ECS应用的Agent卸载方法,请参见如何卸载ECS中Java应用的Agent。
- 如果您的Agent版本为2.5.8.1或以上,则可在不重装Agent的情况下更改Java应用名称,请参见以下操作步骤:

⑦ 说明 2019年8月20日之后下载的Agent默认都支持该功能。

1. 请使用以下方法之一下载并解压Supervisor。

⑦ 说明 请根据您的应用所在的地域替换下载地址中的地域。

。 公网地址

```
wget http://arms-apm-hangzhou.oss-cn-hangzhou.aliyuncs.com/ArmsSupervisor.zip -0 Arms
Supervisor.zip
unzip ArmsSupervisor.zip
```

○ VPC地址(公网地址无法下载时使用)

```
wget http://arms-apm-hangzhou.oss-cn-hangzhou-internal.aliyuncs.com/ArmsSupervisor.zi
p -0 ArmsSupervisor.zip
unzip ArmsSupervisor.zip
```

#### 2. 运行以下命令更改应用名称。

```
cd ArmsSupervisor
./attach.sh </path/to/ArmsAgent/arms-bootstrap-1.7.0-SNAPSHOT.jar> <PID> <NewAppName>
<LicenseKey>
```

- 。 </path/to/ArmsAgent/arms-bootstrap-1.7.0-SNAPSHOT.jar>: 指向Agent下同名文件的路径。
- o <PID>: 目标进程ID, 可通过 jps/ps 命令获取。
- <NewAppName>: 新的应用名称。

○ <LicenseKey>: ARMS应用监控中应用的LicenseKey, 可从控制台获取。

如果应用的标准输出打印以下日志,则说明应用名称更改成功。

19-88-20 21:48:81 [WAW] [com.navercorp.pinpoint.bootstrap.PinpointBootStrap] rename appliane from chao leng to 19-88-20 21:48:84 [WAW] [com.navercorp.pinpoint.bootstrap.PinpointBootStrap] [WBU] applici 19-88-20 21:48:44 [WAW] [com.navercorp.pinpoint.bootstrap.PinpointBootStrap] [rename shortstap already started, skipping appent loading.

[回到顶部]

#### 以快速方式安装Agent的普通Java应用如何更改应用名称?

如果您是以快速方式安装Agent的,则Agent目录为~/*.arms/supervisor/agent*。注意,使用的账号必须与应用相同。

请按照以下步骤更改应用名称。

1. 登录应用所在的机器,并运行以下命令。

```
cd ~/.arms/supervisor
./cli.sh <LicenseKey> <NewAppName>
```

- <LicenseKey>: ARMS应用监控中应用的LicenseKey, 可从控制台获取。
- <NewAppName>: 新的应用名称。
- 2. 从程序列出的所有Java进程中选择正确的进程。
  - ⑦ 说明 如果只有一个应用进程则默认选中该进程。
- 3. 打开~/.arms/attach.info文件, 将需要修改的旧应用名称修改为新应用名称并保存文件。

警告 修改文件时切勿多加空格等内容,否则会因为与前述步骤中修改的新应用名称不符而导致修改失败。

应用名称更改成功后稍等片刻,旧名称的应用下将不再有监控数据上报,且新名称的应用下将有监控数 据上报。

#### 部署在ECS中的Java应用如何更改应用名称?

如果是部署在ECS实例中的Java应用,则Agent目录为/.arms/agent。

#### 请按照以下步骤更改应用名称。

1. 登录应用所在的机器,并使用root账号运行以下命令。

```
su <USER> -c "./attach.sh /.arms/agent/arms-bootstrap-1.7.0-SNAPSHOT.jar <PID> <NewAp
pName> <LicenseKey>"
```

- <PID>: 目标进程ID, 可通过 jps/ps 命令获取。
- <NewAppName>: 新的应用名称。
- <LicenseKey>: ARMS应用监控中应用的LicenseKey, 可从控制台获取。
- 2. 打开~/.arms/attach.info文件,将需要修改的旧应用名称修改为新应用名称并保存文件。

警告 修改文件时切勿多加空格等内容,否则会因为与前述步骤中修改的新应用名称不符而导致修改失败。

如果应用的标准输出打印以下日志,则说明应用名称更改成功。

#### 部署在容器服务K8s集群中的Java应用如何更改应用名称?

您可以在Agent目录的Version文件中查看Agent版本。例如, 2.5.8_cf020486_20190816150025 表示 Agent版本为2.5.8, Agent发布时间为2019年8月16日。

- 如果您的Agent版本低于2.5.8.1,请先卸载Agent再重新安装,并在重新安装时使用新的应用名称。
  - 手动安装的Agent的卸载方法,请参见如何卸载以通用方式安装的Agent。
  - 。 以脚本自动安装的Agent的卸载方法,请参见如何卸载以快速方式安装的Agent。
  - 。 ECS应用的Agent卸载方法,请参见如何卸载ECS中Java应用的Agent。
- 如果您的Agent版本为2.5.8.1或以上,则可在不重装Agent的情况下更改Java应用名称,请参见以下操作步骤:

⑦ 说明 2019年8月20日之后下载的Agent默认都支持该功能。

修改Deployment内的armsPilotCreateAppName参数并重启Pod即可。

应用名称更改成功后稍等片刻,旧名称的应用下将不再有监控数据上报,且新名称的应用下将有监控数据上报。

#### [回到顶部]

#### 部署在EDAS上的Java应用如何更改应用名称?

目前不支持为部署在EDAS上的Java应用更改应用名称。

[回到顶部]

#### 如何卸载以通用方式安装的Agent?

以通用方式安装Agent是指使用手动方式为Java应用安装Agent,具体操作,请参见为Java应用手动安装 Agent。卸载以通用方式安装的Agent的操作步骤如下所示:

- 1. 当您不需要使用ARMS监控您的Java应用时,删除上述安装文档步骤8添加的AppName、LicenseKey相关的所有参数。
- 2. 重启Java应用。

#### [回到顶部]

#### 如何卸载以快速方式安装的Agent?

以快速方式安装Agent是指使用一键接入脚本为Java应用安装Agent,具体操作,请参见使用脚本为Java应用快 速安装探针。卸载以快速方式安装的Agent的操作步骤如下所示:

1. 当您不需要使用ARMS监控您的Java应用时,执行 jps -1 命令查看所有进程,并在执行结果中找到 c om.alibaba.mw.arms.apm.supervisor.daemon.Daemon 对应的进程号。

在本示例中, com.alibaba.mw.arms.apm.supervisor.daemon.Daemon 对应的进程号为: 62857。



```
2. 执行命令 kill -9 <进程号> 。
```

例如: kill -9 62857 。

- 3. 执行 rm -rf /.arms /root/.arms 。
- 4. 重启您的应用。

[回到顶部]

#### 如何卸载ECS中Java应用的Agent?

1. 当您不需要使用ARMS监控您的Java应用时,执行 jps -1 命令查看所有进程,并在执行结果中找到 c om.alibaba.mw.arms.apm.supervisor.daemon.Daemon 对应的进程号。

在本示例中, com.alibaba.mw.arms.apm.supervisor.daemon.Daemon 对应的进程号为: 62857。



2. 执行命令 kill -9 <进程号> 。

例如: kill -9 62857 。

- 3. 执行 rm -rf /.arms /root/.arms 。
- 4. 重启您的应用。

#### [回到顶部]

## 如何卸载容器服务K8s集群中Java应用的Agent?

当您不需要使用ARMS监控容器服务Kubernetes版集群中的Java应用时,可按照以下步骤卸载Agent:

- 1. 登录容器服务管理控制台。
- 2. 在左侧导航栏单击集群,在集群列表页面上的目标集群右侧操作列单击应用管理。
- 3. 在左侧导航栏选择应用 > Helm。
- 4. 在Helm页面, ARMS Agent对应的发布名称arms-pilot右侧操作列, 单击删除。
- 5. 在删除应用对话框单击确定。
- 6. 重启您的业务Pod。

#### [回到顶部]

#### 如何卸载开源K8s环境中Java应用的Agent?

1. 当您不需要再监控开源Kubernetes环境中的Java应用时,可执行以下命令卸载arms-pilot。

helm del --purge arms-pilot

2. 重启您的业务Pod。

#### [回到顶部]

#### 如何卸载Docker中Java应用的Agent?

1. 当您不需要再监控Docker集群中的Java应用时,可以删除安装文档的步骤1所编辑的Dockerfile内容。

- 2. 运行docker build命令来构建镜像。
- 3. 运行docker run命令来启动镜像。

#### [回到顶部]

#### 使用OpenFeign组件的应用在ARMS中数据不完整怎么办?

若您的OpenFeign应用接入ARMS应用监控后,出现数据不完整、看不到下游应用的数据等情况,可能的原因是OpenFeign组件默认开启了使用RxJava异步框架的Hystrix,而ARMS不支持异步框架。

⑦ 说明 本文仅限于ARMS应用监控Java Agent版本低于2.6.0的场景, 2.6.0及以上版本已支持异步框架。

您可以通过关闭Hystrix并配置OkHttp请求类来解决此类问题:

1. 在 pom.xml 文件中添加以下依赖。

```
<!-- OKHttp对Feign支持 -->
<dependency>
<groupId>io.github.openfeign</groupId>
<artifactId>feign-okhttp</artifactId>
</dependency>
```

#### 2. 在SpringCloud配置文件中添加以下配置。

feign.okhttp.enabled: true
feign.hystrix.enabled: false

#### 3. 配置OkHttp请求类。

```
@Configuration
@ConditionalOnClass(Feign.class)
@AutoConfigureBefore(FeignAutoConfiguration.class)
public class FeignClientOkHttpConfiguration {
    @Bean
    public OkHttpClient okHttpClient() {
       return new OkHttpClient.Builder()
               // 连接超时
                .connectTimeout(20, TimeUnit.SECONDS)
               // 响应超时
               .readTimeout(20, TimeUnit.SECONDS)
               // 写超时
                .writeTimeout(20, TimeUnit.SECONDS)
               // 是否自动重连
                .retryOnConnectionFailure(true)
               // 连接池
                .connectionPool(new ConnectionPool())
                .build();
    }
```

#### [回到顶部]

# 8.故障排除

# 8.1. ARMS Agent安装成功,为什么控制台 上仍无监控数据?

#### Condition

ARMS Agent安装成功,控制台上仍无监控数据。

#### Cause

应用无持续的外部请求访问、应用所属地域和ARMS Agent所属地域不一致、或者ARMS 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。

#### 操作步骤

- 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ArmsAgent的支持范围内。关于 ARMS应用监控对第三方组件和框架的支持情况,请参见ARMS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15分钟,然后再次确认是否有监控数据。
- 3. 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}/ArmsAgent/arms-bootstrap-1.7.0-SNAPSHOT.jar -Darms.li censeKey=xxx -Darms.appName=xxx -jar demoApp.jar

- 4. 如果ArmsAgent /log/下的日志中出现LicenseKey is invalid.异常,请您检查应用所属地域与Agent所属 地域是否一致。
- 5. 如果应用启动之后ArmsAgent目录下无log子目录,是由于arms-bootstrap-1.7.0-SNAPSHOT.jar未被成功加载导致的,请您检查ArmsAgent安装目录的权限是否正确。
- 6. 若应用启动时出现以下报错,请您检查arms-bootstrap-1.7.0-SNAPSHOT.jar软件包及相应权限是否正确。

Error opening zip file or JAR manifest missing : /root/ArmsAgent/arms-bootstrap-1.7.0-SNAPSHOT.jar Error occurred during initialization of VM agent library failed to init: instrument

- 7. 如果仍无监控数据,请打包Java探针日志(路径: ArmsAgent/log),并联系钉钉服务账号 arms16080
   4 解决问题。
- 8. 检查JDK版本。

如果JDK版本为1.8.0_25或者1.8.0_31,可能会出现无法安装探针的情况,建议您升级对应的JDK版本, 或咨询钉钉服务账号 arms160804 。

# 8.2.为什么容器服务Kubernetes版集群中的Java应用安装Agent后,应用监控没有数据呢?

#### Condition

阿里云容器服务Kubernetes版集群中的Java应用安装Agent后,应用监控没有数据。

#### Cause

应用所在Pod未被注入arms-init-container、应用YAML文件中无Labels注解以及STS服务未正确授权等原因都可能导致应用监控无数据。

#### Remedy

#### 操作步骤

- 1. 登录容器服务管理控制台。
- 2. 在左侧导航栏选择集群,在集群列表页面上的目标集群右侧操作列单击应用管理。
- 3. 在容器组页签顶部选择您的应用所在的命名空间,然后单击目标应用右侧单击编辑。
- 4. 在编辑YAML对话框中查看YAML文件中是否存在init Containers。

<pre>94</pre>	巃 YAM	L	
<pre>preadOnly: true preadOnly: true preadOnly</pre>	94	name: default-token-5cm96	
<pre>96</pre>	95	readOnly: true	
<pre>97   name: arms-agent-vol 98   name: arms-podinfo 99   name: arms-podinfo 99   name: volume-tart:/pt/arms-debug/share 99   dnsPolicy: ClusterFirst 90   initContainers: 91   args: 92   args: 93   - args: 94   initContainers: 95   - args: 96   - '-pid-a 97   - name: ARMS_PILOT_INIT_REGIONID 99   - name: ARMS_PILOT_INIT_REGIONID 99   value: cn-hangzhou 99   mame: arms-pilot-init:v1.0.1 91   image: &gt;- 91   registry-vpc.cn-hangzhou.aliyuncs.com/arms-docker-repo/arms-pilot-init:v1.0.1 91   image: &gt;- 92   registry-vpc.cn-hangzhou.aliyuncs.com/arms-docker-repo/arms-pilot-init:v1.0.1 93   image: s- 94   reminationNessagePath: /dev/termination-log 95   reminationNessagePath: /dev/termination-log 96   terminationNessagePath: /dev/termination-log 97   restartPolicy: Always 98   restartPolicy: Always 99   restartPolicy: Always 91   restartPolicy: Always 92   restartPolicy: Always 93   restartPolicy: Always 94   schedulerName: default 95   serviceAccount: default 95   serviceAccount: default 96   terminationNessePeriodSeconds; 30</pre>	96 -	- mountPath: /home/admin/.opt/	
<pre>99</pre>	97	name: arms-agent-vol	
<pre>99   name: arms-podinfo 100 - mountPath: /opt/arms-debug/share 111 name: volume-tmp-share 112 dnsPolicy: ClusterFirst 113 enableServiceLinks: true 114 initContainers: 115 - args: 116 - '-pid= 117 registry-vpc.cn-hangzhou.aliyuncs.com/arms-docker-repo/arms-pilot-init:v1.0.1 118 imagePullPolicy: Always 114 name: arms-pilot-initcontainer 115 registry-vpc.cn-hangzhou.aliyuncs.com/arms-docker-repo/arms-pilot-init:v1.0.1 118 imagePullPolicy: Always 114 name: arms-pilot-initcontainer 115 resources: {} 116 terminationMessagePath: /dev/termination-log 117 terminationMessagePath: /dev/termination-log 118 volumeMounts: 119 - noutPath: /home/admin/.opt/ 110 nume: arms-agent-vol 111 nodeName: default 122 serviceAccount! default 123 serviceAccount! default 124 terminationGracePeriodSeconds: 30</pre>	98 -	- mountPath: /etc/podinfo	
100       - mountPath: /opt/arms-debug/share         101       name: volume-tmp-share         102       dnsPolicy: ClusterFirst         103       - args:         104       initContainers:         105       - '-java=true'         106       - '-java=true'         107       - '-java=true'         108       - '-java=true'         109       - '-java=true'         101       name: ARMS_PILOT_INIT_REGIONID         102       value: cn-hangzhou         103       image: b;         104       resulto: cn-hangzhou         105       name: arms-pilot-initcontainer         106       terminationMessagePath: /dev/termination-log         107       terminationMessagePath: /dev/termination-log         108       volumeMounts:         109       - mountPath: /home/admin/.opt/         119       - mountPath: /home/admin/.opt/         121       name: arms-agent-vol         122       priority: 0         123       restartPolicy: Always         124       securityContext: {}         125       securityContext: {}         126       serviceAccountName: default         127       serviceAccountName: default	99	name: arms-podinfo	
101       name: volume-tmp-share         102       dnsPolicy: ClusterFirst         103       enableServiceLinks: true         104       initContainers:         105       - args:         106       - '-pid-         107       - '-pid-         108       env:         109       - name: ARMS_PILOT_INIT_REGIONID         109       value: cn-hangzhou         110       value: cn-hangzhou         111       inagePullPolicy: Always         112       registry-vpc.cn-hangzhou.aliyuncs.com/arms-docker-repo/arms-pilot-init:v1.0.1         113       inagePullPolicy: Always         114       name: arms-pilot-initcontainer         resources: {}       terminationNessagePolicy: File         volumeMounts:       - noutPath: /home/admin/.opt/         113       nodeName: cn-hangzhou.         114       priority: 0         115       restartPolicy: Always         116       schedulerName: default         117       serviceAccount: default         118       volumeAuto: scheduler         119       - mourtPath: /home/admin/.opt/         111       nodeName: default         112       schedulerName: default         113	100 -	- mountPath: /opt/arms-debug/share	
102       dnsPolicy: ClusterFirst         103       enableServiceLinks: true         104       initContainers:         105       - args:         106       - '-pid         107       - name: ARMS_PILOT_INIT_REGIONID         108       env:         109       - name: ARMS_PILOT_INIT_REGIONID         109       value: cn-hangzhou         111       image: >-         112       registry-vpc.cn-hangzhou.aliyuncs.com/arms-docker-repo/arms-pilot-init:v1.0.1         113       imagePullPolicy: Always         114       name: arms-pilot-initcontainer         115       resources: {}         116       terminationMessagePath: /dev/termination-log         117       terminationMessagePolicy: File         volumeMounts:       -         119       -         119       modeName: cn-hangzhou.         111       restartPolicy: Always         112       priority: 0         restartPolicy: Always         117       serviceAccount: default         118       serviceAccount: default         119       serviceAccount: default         120       serviceAccountName: default         121       terminationGracePeriodSeconds: 30 <td>101</td> <td>name: volume-tmp-share</td> <td></td>	101	name: volume-tmp-share	
103       enableServiceLinks: true         104       initContainers:         - args:       - '-java=true'         06       - '-java=true'         07       - '-java=true'         08       - ''-pid=         09       - name: ARMS_PILOT_INIT_REGIONID         101       value: cn-hangzhou         102       registry-vpc.cn-hangzhou.aliyuncs.com/arms-docker-repo/arms-pilot-init:v1.0.1         103       imagePulPolicy: Always         104       terminationMessagePath: /dev/termination-log         105       resources: {}         106       terminationMessagePolicy: File         107       name: arms-agent-vol         108       volumeMounts:         109       restartPolicy: Always         119       schedulerName: default-scheduler         120       serviceAccount: default         121       serviceAccount: default         122       terminationGracePeriodSeconds: 30	102	dnsPolicy: ClusterFirst	
<pre>initContainers: - args: - '-java-true' - '-jid= env: - name: ARMS_PILOT_INIT_REGIONID value: cn-hangzhou image: &gt;- registry-vpc.cn-hangzhou.aliyuncs.com/arms-docker-repo/arms-pilot-init:v1.0.1 imagePullPolicy: Always name: arms-pilot-initcontainer resources: {} terminationMessagePath: /dev/termination-log terminationMessagePolicy: File volumeMounts: - moutPath: /home/admin/.opt/ name: arms-agent-vol nodeName: cn-hangzhou. priority: 0 restartPolicy: Always schedulerName: default-scheduler securityContext: {} securityContext: {} securityContext: {} is erviceAccount! default serviceAccount! default is erviceAccount! affault is terminationGracePeriodSeconds: 30</pre>	103	enableServiceLinks: true	
105       - args:         106       - '-java=true'         107       - name: ARMS_PILOT_INIT_REGIONID         108       env:         109       - name: ARMS_PILOT_INIT_REGIONID         110       image: >         registry-vpc.cn-hangzhou.aliyuncs.com/arms-docker-repo/arms-pilot-init:v1.0.1         111       image/ullPolicy: Always         112       registry-vpc.cn-hangzhou.aliyuncs.com/arms-docker-repo/arms-pilot-init:v1.0.1         113       image/ullPolicy: Always         114       name: arms-pilot-initcontainer         115       resources: {}         116       terminationMessagePolicy: File         volumeMounts:       -         - mame: arms-agent-vol         111       nodeName: cn-hangzhou.         112       priority: 0         113       restartPolicy: Always         114       schedulerName: default         115       securityContext: {}         116       serviceAccount! default         117       serviceAccountName: default         118       terminationGracePeriodSeconds: 30	104 -	initContainers:	
106       - '-java-true'         107       - 'name: ArMS_PILOT_INIT_REGIONID         108       - name: ArMS_PILOT_INIT_REGIONID         109       - value: cn-hangzhou         110       image: >-         registry-vpc.cn-hangzhou.aliyuncs.com/arms-docker-repo/arms-pilot-init:v1.0.1         113       image/pullPolicy: Always         114       name: arms-pilot-initcontainer         resources: {}       terminationMessagePolicy: File         volumeMounts:       - name: arms-agent-vol         110       nodeName: cn-hangzhou.         111       nodeName: cn-hangzhou.         112       priority: 0         restartPolicy: Always         114       serviceAccount: default         115       serviceAccountMame: default         116       serviceAccountName: default         117       serviceAccountName: default         118       terminationGracePeriodSeconds: 30	105 -	- args:	
107      plat- env:         109       - name: ARMS_PILOT_INIT_REGIONID         100       value: cn-hangzhou         111       image: >- registry-vpc.cn-hangzhou.aliyuncs.com/arms-docker-repo/arms-pilot-init:v1.0.1         113       imagePullPolicy: Always         114       name: arms-pilot-initcontainer         resources: {}       terminationMessagePath: /dev/termination-log         118       volumeMounts:         119       - mountPath: /home/admin/.opt/ name: arms-agent-vol         110       restartPolicy: Always         122       priority: 0         123       restartPolicy: Always         124       schulerName: default-scheduler         125       securityContext: {}         126       serviceAccounti default         127       serviceAccounti default         128       terminationGracePeriodSeconds: 30	105	Java=true	
109       - name: ARMS_PILOT_INIT_REGIONID         110       value: cn-hangzhou         111       image: >-         registry-vpc.cn-hangzhou.aliyuncs.com/arms-docker-repo/arms-pilot-init:v1.0.1         113       imagePullPolicy: Always         114       registry-vpc.cn-hangzhou.aliyuncs.com/arms-docker-repo/arms-pilot-init:v1.0.1         113       imagePullPolicy: Always         114       resources: {}         115       reminationMessagePolicy: File         volumeMounts:       - moutPath: /home/admin/.opt/         119       - moutPath: /home/admin/.opt/         120       name: arms-agent-vol         121       restartPolicy: Always         122       schedulerName: default-scheduler         123       restartPolicy: Always         124       schedulerName: default         125       securityContext: {}         126       serviceAccount! default         127       serviceAccountName: default         128       terminationGracePeriodSeconds: 30	107	- '-pid=;	
199       - name: arms-pillol_INIT_REGUMUD         110       walue: cn-hangzhou         111       image: >-         registry-vpc.cn-hangzhou.aliyuncs.com/arms-docker-repo/arms-pilot-init:v1.0.1         113       image/pullPolicy: Always         114       name: arms-pilot-initcontainer         resources: {}       terminationMessagePath: /dev/termination-log         117       terminationMessagePolicy: File         volumeMounts:       -         - montPath: /home/admin/.opt/         120       name: arms-agent-vol         121       nodeName: cn-hangzhou.         122       priority: 0         restartPolicy: Always         123       restartPolicy: Always         124       schedulenHame: default-scheduler         125       securityContext: {}         126       serviceAccountName: default         127       serviceAccountName: default         128       terminationGracePeriodSeconds: 30	108 -	env:	
111       Value: cn-nangznou         111       image: >-         registry-vpc.cn-hangzhou.aliyuncs.com/arms-docker-repo/arms-pilot-init:v1.0.1         113       imagePullPolicy: Always         114       name: arms-pilot-initcontainer         resources: {}       terminationMessagePath: /dev/termination-log         115       terminationMessagePolicy: File         116       terminationMessagePolicy: File         117       nodeMame: arms-agent-vol         118       volumeMounts:         119       - mountPath: /home/admin/.opt/         120       priority: 0         122       priority: 0         123       restartPolicy: Always         124       schedulerName: default-scheduler         125       securityContext: {}         126       serviceAccount! default         127       serviceAccount? default         128       terminationGracePeriodSeconds: 30	109 -	- name: ARMS_PILOI_INII_REGIONID	
111       「mage: >- registry-vpc.cn-hangzhou.aliyuncs.com/arms-docker-repo/arms-pilot-init:v1.0.1         113       imagePullPolicy: Always         114       name: arms-pilot-initcontainer         115       resources: {}         116       terminationMessagePath: /dev/termination-log         117       terminationMessagePolicy: File         volumeMounts:       -         119       -         119       -         119       -         119       -         119       -         119       -         119       -         119       -         119       -         1110       name: arms-agent-vol         1121       nodeName: cn-hangzhou.         1122       priority: 0         1123       restartPolicy: Always         124       schedulerName: default-scheduler         125       securityContext: {}         126       serviceAccount! default         127       serviceAccount! default         128       terminationGracePeriodSeconds: 30	110	Value: cn-nangznou	
111       registry-vpc.th-nang2nou.aryuncs.com/arms-docker-rep0/arms-pilot-init:vi.0.1         113       image/pullPolicy: Always         114       name: arms-pilot-initcontainer         115       resources: {}         116       terminationMessagePolicy: File         volumeMounts:	111 *	image: >-	
113       ImagePullPolicy: Always         114       name: arms-piloto-initcontainer         115       resources: {}         116       terminationMessagePath: /dev/termination-log         117       terminationMessagePolicy: File         118       volumeMounts:         119       - mountPath: /home/admin/.opt/         120       nodeName: cn-hangzhou.         121       nodeName: cn-hangzhou.         122       priority: 0         123       restartPolicy: Always         124       schedulerName: default-scheduler         125       securityContext: {}         126       serviceAccount! default         127       serviceAccount? default         128       terminationGracePeriodSeconds: 30	112	registry-vpc.cn-nangznou.aiiyuncs.com/arms-docker-repo/arms-piiot-init:vi.0.i	
114       name: arms-pirot-initcontailer         115       resources: {}         116       terminationMessagePath: /dev/termination-log         117       terminationMessagePolicy: file         118       volumeMounts:         119       - mountPath: /home/admin/.opt/         120       name: arms-agent-vol         121       nodeName: cn-hangzhou.         122       priority: 0         123       restartPolicy: Always         124       schedulerName: default-scheduler         125       securityContext: {}         126       serviceAccounti default         127       serviceAccounti default         128       terminationGracePeriodSeconds: 30	115	imagerdiffolity: Always	
113       resources. ()         116       terminationMessagePath: /dev/termination-log         117       terminationMessagePolicy: File         118       volumeMounts:         119       - mountPath: /home/admin/.opt/         120       name: arms-agent-vol         121       nodeName: cn-hangzhou.         122       priority: 0         123       restartPolicy: Always         124       schedulerName: default-scheduler         125       securityContext: {}         126       serviceAccount: default         127       serviceAccountName: default         128       terminationGracePeriodSeconds: 30	114	name: arms-priot-initiontainer	
110     terminationWessagerait. /uev/terminationFlog       117     terminationWessagerait. /uev/terminationFlog       117     terminationWessagerait. /uev/terminationFlog       118     volumeMounts:       119     - mountPath: /home/admin/.opt/       120     nodeName: cn-hangzhou.       121     nodeName: cn-hangzhou.       122     priority: 0       123     restartPolicy: Always       124     schedulerName: default-scheduler       125     securityContext: {}       126     serviceAccount; default       127     serviceAccountName: default       128     terminationGracePeriodSeconds; 30	115	resources: {}	
111       volumeSsagent(), File         113       - mountPath: /home/admin/.opt/         119       - mountPath: /home/admin/.opt/         119       - mountPath: /home/admin/.opt/         120       name: arms-agent-vol         121       nodeName: cn-hangzhou.         122       priority: 0         123       restartPolicy: Always         124       schedulerName: default-scheduler         125       securityContext: {}         126       serviceAccount! default         127       serviceAccountName: default         128       terminationGracePeriodSeconds: 30	117	terminationlessagerath. / vev/termination-rog	
119     - mountPath: /home/admin/.opt/       120     - mountPath: /home/admin/.opt/       121     name: arms-agent-vol       122     priority: 0       123     restartPolicy: Always       124     schedulerName: default-scheduler       125     securityContext: {}       126     serviceAccount! default       127     serviceAccountName: default       128     terminationGracePeriodSeconds: 30	118 -	volumeMounts:	
120     name: arms-agent-vol       121     nodeName: cn-hangzhou.       122     prionity: 0       123     restartPolicy: Always       124     schedulerName: default-scheduler       125     securityContext: {}       126     serviceAccount: default       127     serviceAccountName: default       128     terminationGracePeriodSeconds: 30	110 -		
121     nodeName: cn-hangzhou.       122     priority: 0       123     restartPolicy: Always       124     schedulerName: default-scheduler       125     securityContext: {}       126     serviceAccount! default       127     serviceAccountName: default       128     terminationGracePeriodSeconds: 30	120	name: arms_agent_vol	
122       priority: 0         123       restartPolicy: Always         124       schedulerName: default-scheduler         125       securityContext: {}         126       serviceAccount: default         127       serviceAccountName: default         128       terminationGracePeriodSeconds: 30	121	nodeName: cn-hangzhou.	
123       restartPolicy: Always         124       schedulerName: default-scheduler         125       securityContext: {}         126       serviceAccount: default         127       serviceAccountName: default         128       terminationGracePeriodSeconds: 30         更新       下载       另存为       取消	122	priority: 0	
124       schedulerName: default-scheduler         125       securityContext: {}         126       serviceAccount: default         127       serviceAccountName: default         128       terminationGracePeriodSeconds: 30         更新       下载       另存为       取消	123	restartPolicy: Always	
125       securityContext: {}         126       serviceAccount: default         127       serviceAccountName: default         128       terminationGracePeriodSeconds: 30         更新       下载       另存为       取消	124	schedulerName: default-scheduler	
126       serviceAccount: default         127       serviceAccountName: default         128       terminationGracePeriodSeconds: 30         更新       下线       另存为       取消	125	<pre>securityContext: {}</pre>	
127       serviceAccountName: default         128       terminationGracePeriodSeconds: 30         更新       下载       另存为       取消	126	serviceAccount: default	
128 terminationGracePeriodSeconds: 30 更新 下载 另存为 取消	127	serviceAccountName: default	
更新 下载 另存为 取消	128	terminationGracePeriodSeconds: 30	
更新 下载 另存为 取消			
		更新下载	取消

- 如果不存在,则说明未被注入arms-init-container,执行步骤5。
- 如果存在,则说明已被注入arms-init-container,执行步骤8。
- 5. 在容器组页签顶部选择命名空间为ack-onepilot。查看Pod列表中是否存在名称前缀为ack-onepilot的Pod。
  - 如果存在,则执行步骤6。
  - 如果不存在,则在应用市场中安装ack-onepilot。具体操作,请参见安装ARMS应用监控组件。
- 6. 在无状态或有状态页签目标应用右侧操作列中选择更多 > 查看Yaml, 在编辑YAML对话框查看YAML 文件中是否存在以下Labels注解。

labels: armsPilotAutoEnable: "on" armsPilotCreateAppName: "<your-deployment-name>" //请将<your-deployment-name>替换为 您的应用名称。 one-agent.jdk.version: "OpenJDK11" //如果应用的JDK版本是JDK11,则需要配置此参数。

- 如果存在,则执行步骤7。
- 如果不存在,则在编辑YAML对话框中的*spec > template > metadata*层级下添加以上Labels注解, 然后单击更新。
- 7. 在容器组页签目标应用右侧单击日志,查看ack-onepilot的Pod日志是否报STS错误,即提示 "Messag e":"STS错误" 。
  - 如果报STS错误,则需为应用所在集群授权,并重启应用所在Pod。具体操作,请参见为容器服务 Kubernetes版授权。
  - 如果未报STS错误,则联系ARMS钉钉服务账号: arms160804。
- 8. 在容器组页签目标应用右侧单击编辑,在编辑YAML对话框中查看YAML文件中是否存在以下javaagent参数。

-javaagent:/home/admin/.opt/ArmsAgent/arms-bootstrap-1.7.0-SNAPSHOT.jar

如果存在,则单击容器组页签右侧的终端进入命令行页面,执行以下命令查看是否存在以.log为后缀的日志文件,然后联系ARMS钉钉服务账号: arms160804。

cd /home/admin/.opt/ArmsAgent/logs

○ 如果不存在,则联系ARMS钉钉服务账号: arms160804。