



让智能运维成为每个企业的标配



# 一站式智能观测平台

北京快猫星云科技有限公司



由开源项目夜莺  
的核心开发团队组成



创始团队均来自  
国内一线互联网公司



累计为上百家企业  
提供服务



由国内顶级投资机构  
连续投资



Nightingale 夜莺是一个 All-in-One 的开源监控工具，最早由滴滴开发并开源，是中国计算机学会接受捐赠并托管的第一个开源项目，在 GitHub 上有超过 11000 颗星，是广受关注和使用的开源监控工具。快猫星云技术团队是夜莺开源项目的重要开发力量。



Flashcat 是快猫星云以开源夜莺为核心打造的一体化观测平台，支持指标、日志、链路追踪数据的统一采集、存储、监控告警、可视化分析，只需一个 Flashcat 平台即可全面覆盖云上、云下、Kubernetes 的观测需求。Flashcat 预置了行业领先的故障发现定位最佳实践，并深度使用 AI 加速故障的分析过程，大幅缩短故障恢复时间。



Flashduty 是一站式告警事件响应平台，支持告警聚合、降噪、排班、协作，内置飞书/钉钉/企微/短信/电话等多种通知方式，在 IM 和 App 中响应和处理告警，全生命周期管理告警事件、分析告警数据，提升 OnCall 体验，缩短 MTTA/MTTR。



# 快猫星云 ❤️ Nightingale

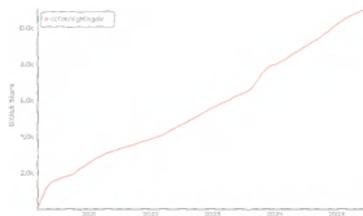
## CNCF landscape project



11000+

GitHub Star

是最受关注的开源监控项目之一



150

Release

150多个版本发布，社区活跃

140

代码贡献者

上百位社区贡献者参与其中，群策群力

10000+

企业用户

政企、金融、科技、互联网，众多公司信赖之选



中国计算机学会接受捐赠并托管的第一个开源项目

# Flashcat 解决什么问题



## 一体化的观测平台

组织好数据

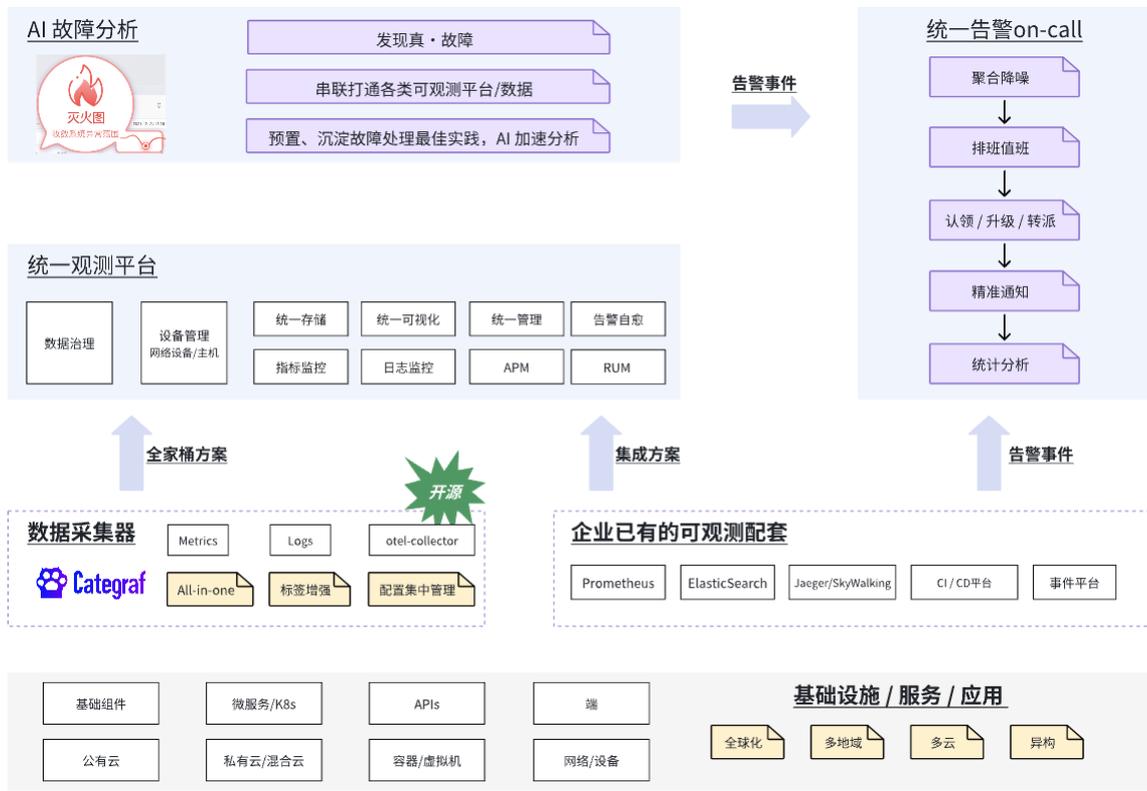
- ✓多对象：  
物理机 / 微服务 / 网络 / Kubernetes
- ✓多平台：  
公有云、私有云、多云
- ✓多维度：  
Metrics、Logs、APM、Events、RUM

## 故障发现定位的智能助手

用好数据

- ✓发现故障：  
快速准确发现业务故障
- ✓定位故障：  
一站式、引导式完成分析和定位
- ✓最佳实践：  
持续积累和进化的稳定性保障实践
- ✓智能化：  
深度使用 AI 加速问题分析和修复过程

Flashcat 旨在构建一个一体化的观测平台，并致力于解决服务稳定性保障过程中的核心难题



## 统一采集

采用插件化, 内置集成上百种采集插件, GPU、服务器、网络设备、中间件、数据库、应用、业务, 云上云下, 均可监控, 开箱即用。

## 统一集成

可集成企业内部已有的、云上云下的可观测配套系统, 支持对接数十种数据源, 已有工具无需推倒重来, 充分利旧, 减少落地阻力, 快速见效, 串联打通数据, 发挥协同分析的价值。

## 统一观测平台

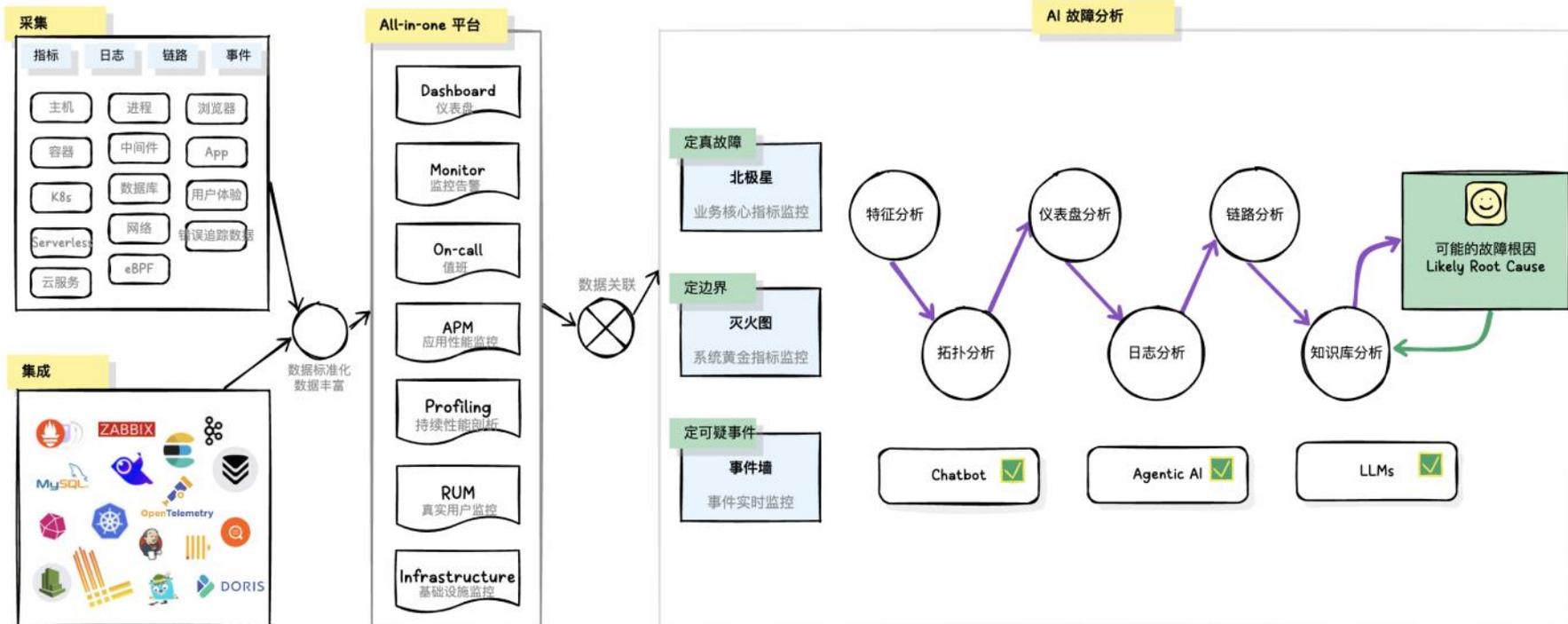
将Metrics、Logs、Traces、Events、Profiling 等多种可观测性数据拉通, 一个平台集中观测, 具备完整的平台能力。

## 统一告警Oncall

支持指标告警、日志告警、智能告警, 支持收集数十种各类监控系统的告警事件, 进行统一的告警收敛、降噪、排班、认领、升级、协同, 大幅提升告警处理效率, 是各技术团队日常协同处理告警的统一入口。

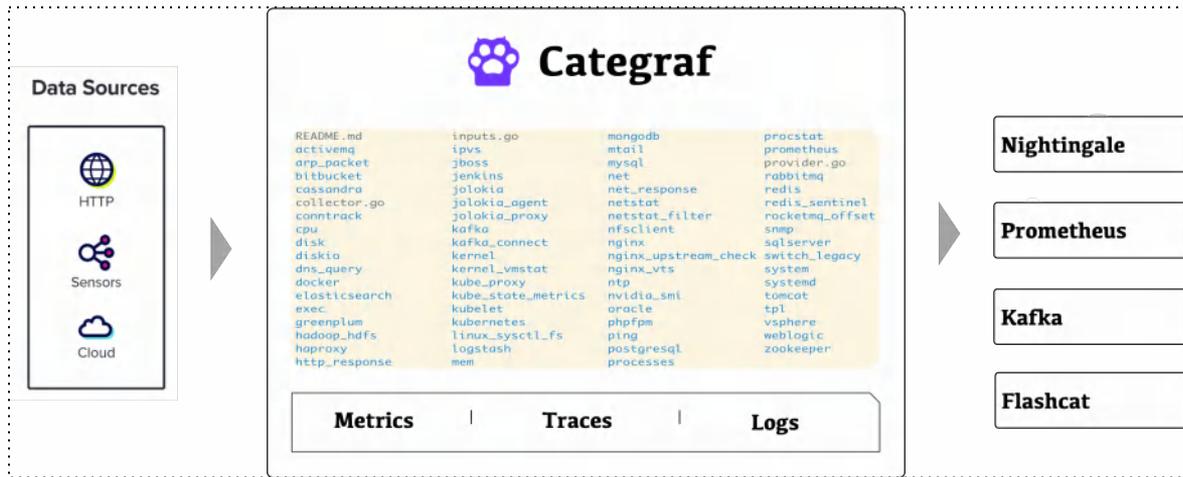
## AI 故障分析

预置稳定性保障行业最佳实践, 既提供全局业务视角, 又具备层层下钻、引导式、智能化的故障定位能力, 有效缩短故障发现和定位时间, 是各技术团队稳定性保障的统一入口。



# Categraf | All-in-one的采集器

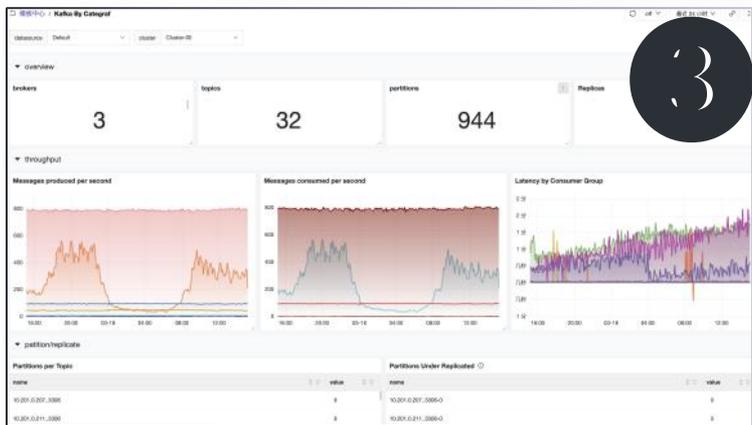
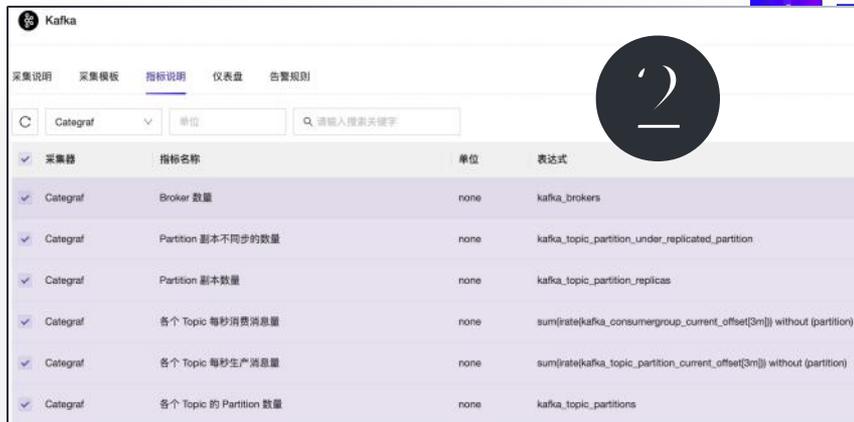
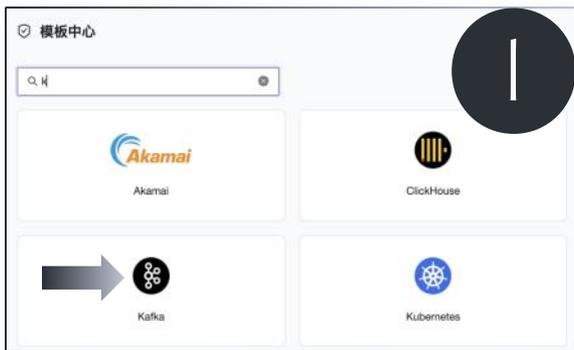
# 开源 / All-in-one / 开箱即用



- 1、中心端集中管理所有采集点配置
- 2、支持agent自升级
- 3、支持标签增强
- 4、支持agent模式、Proxy模式运行
- 5、支持K8s部署
- 6、支持网络拨测
- 7、支持Pingmesh
- 8、支持持续性能剖析

release v0.3.82 docker pulls 1.3M Stars 860 Forks 261 contributors 73 license MIT Powered By Flashcat

**100多种插件，开箱即用**  
**超过130万次下载，广受信赖**



# 统一的数据源集成

# 多数据源集成



### 指标数据源

可用于 告报警, 防火墙, 仪表盘, 告警管理

 Prometheus Like 添加	 MySQL 添加	 InfluxDB 添加	 Oracle 添加
 Zabbix 添加	 PostgreSQL 添加	 ClickHouse 添加	 SQL Server 添加
 JSON API 添加	 MongoDB MongoDB 添加		

### 链路跟踪数据源

可用于 链路分析

 Zipkin 添加	 Jaeger 添加	 Skywalking Skywalking 添加	 自定义追踪 添加
 Elastic APM 添加	 SLS Trace 添加	 阿里云 OpenTelemetry 添加	 腾讯云 APM 添加
 Arms Trace 添加	 Tempo 添加	 OpenTelemetry 添加	 Pinpoint 添加

### 日志数据源

可用于 日志分析, 告警管理, 仪表盘

 kafka 添加	 Elasticsearch 添加	 阿里云SLS 添加	 ClickHouse 添加
 腾讯云CLS 添加	 OpenSearch 添加	 Loki 添加	 Doris 添加

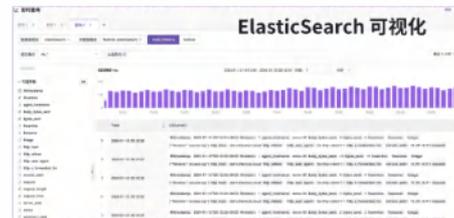
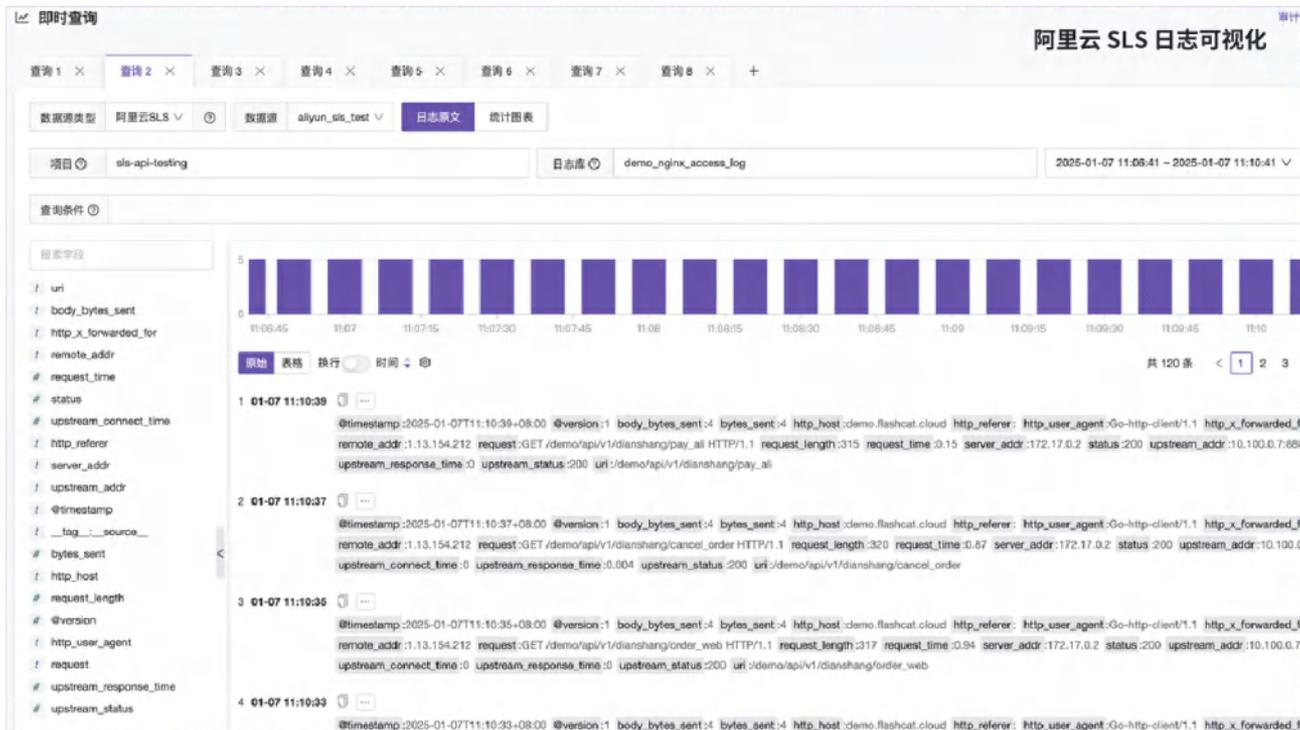
### 事件数据源

自定义事件

 自定义事件 添加	 Jira 添加	 Kubernetes 添加	 Jenkins 添加
 自定义事件 添加	 Prometheus 添加	 Zabbix 添加	 Nightingale 添加
 Open-Falcon 添加	 腾讯云监控 CM 添加		

对接40多种数据源  
与企业已有的可观测性配  
套, 深度集成融合, 对其  
进行统一分析

# 日志统一可视化



# 日志转换和报表



选择日志源 添加日志源

go-skywalking-nginx-log

消息子集

日志样例 (0) (from beginning)

```
{
  "message": "{@timestamp}"/"2024-09-12T18:34:36488:00V", "version": "1", "server_addr": "172.17.0.2", "remote_addr": "172.17.0.2", "http_host": "localhost", "uri": "/roll", "body_bytes_sent": 81, "upstream_response_time": 0.892, "upstream_status": 200, "upstream_connect_time": 0.891, "request": "GET /roll HTTP/1.1", "request_length": 1480, "request_time": 0.892, "status": "200", "http_referer": "", "http_x_forwarded_for": "", "trace_id": "F40b607bf7e5e_user_agent": "Go-http-client/1.1", "status": "info", "timestamp": "2024-09-12T18:34:36488:00V", "agent_hostname": "demo-03", "agent_namespace": "skywalking-nginx", "service": "skywalking-nginx", "resource": "demo03", "tags": "{@filename:access.log_format", "topic": "go-skywalking-demo-nginx", "msg_key": "demo-03"}
}
```

JSON反序列化

message	status	timestamp	agent	key_value提取
message				msg_key

复制文本提取

message

日志分析 默认主题

接口维度 域名维度 接口+refer remote\_addr 来源 hostname hostname-upstream

http\_host request 重置

筛选条件

流量(req/min) 成功率(%)

http_host	request	流量(req/...	成功率(%)	50分位
demo.flashcat.cloud	/demo/api/v1/zhibo/speak	212	100	1.00
demo.flashcat.cloud	/demo/api/v1/zhibo/hotlink	209	100	0.00
demo.flashcat.cloud	/demo/api/v1/dianshang/order_app	206	100	5.00
demo.flashcat.cloud	/demo/api/v1/dianshang/order_web	198	100	0.00
demo.flashcat.cloud	/demo/api/v1/chuxing/price_estimate	190	99	1.00
demo.flashcat.cloud	/demo/api/v1/zhibo/user_online	170	99	0.00

关联的日志源 添加日志源

日志源 demo-kafka

```
{@tags: "filename:access.log", "message": "{@timestamp}"/"2022-10-09T18:44:21+08:00V", "version": "1", "server_addr": "172.17.0.2", "remote_addr": "129.211.209.151", "http_host": "demo.flashcat.cloud", "uri": "/demo/api/v1/chuxing/price_estimate", "body_bytes_sent": 4, "body_bytes_recv": 10, "upstream_addr": "10.100.0.7:8088", "upstream_response_time": 0.808, "upstream_status": 200, "upstream_connect_time": 0.808, "request": "GET /demo/api/v1/chuxing/price_estimate HTTP/1.1", "request_length": 326, "request_time": 0.808, "status": "200", "http_referer": ""}
```

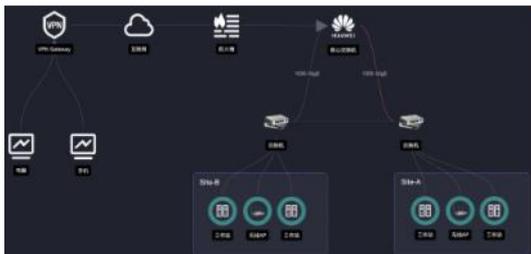
关联规则

```
{@tags: "filename:access.log", "message": "{@timestamp}"/"2022-10-09T18:44:21+08:00V", "version": "1", "server_addr": "172.17.0.2", "remote_addr": "129.211.209.151", "http_host": "demo.flashcat.cloud", "uri": "/demo/api/v1/chuxing/price_estimate", "body_bytes_sent": 4, "body_bytes_recv": 10, "upstream_addr": "10.100.0.7:8088", "upstream_response_time": 0.808, "upstream_status": 200, "upstream_connect_time": 0.808, "request": "GET /demo/api/v1/chuxing/price_estimate HTTP/1.1", "request_length": 326, "request_time": 0.808, "status": "200", "http_referer": ""}
```

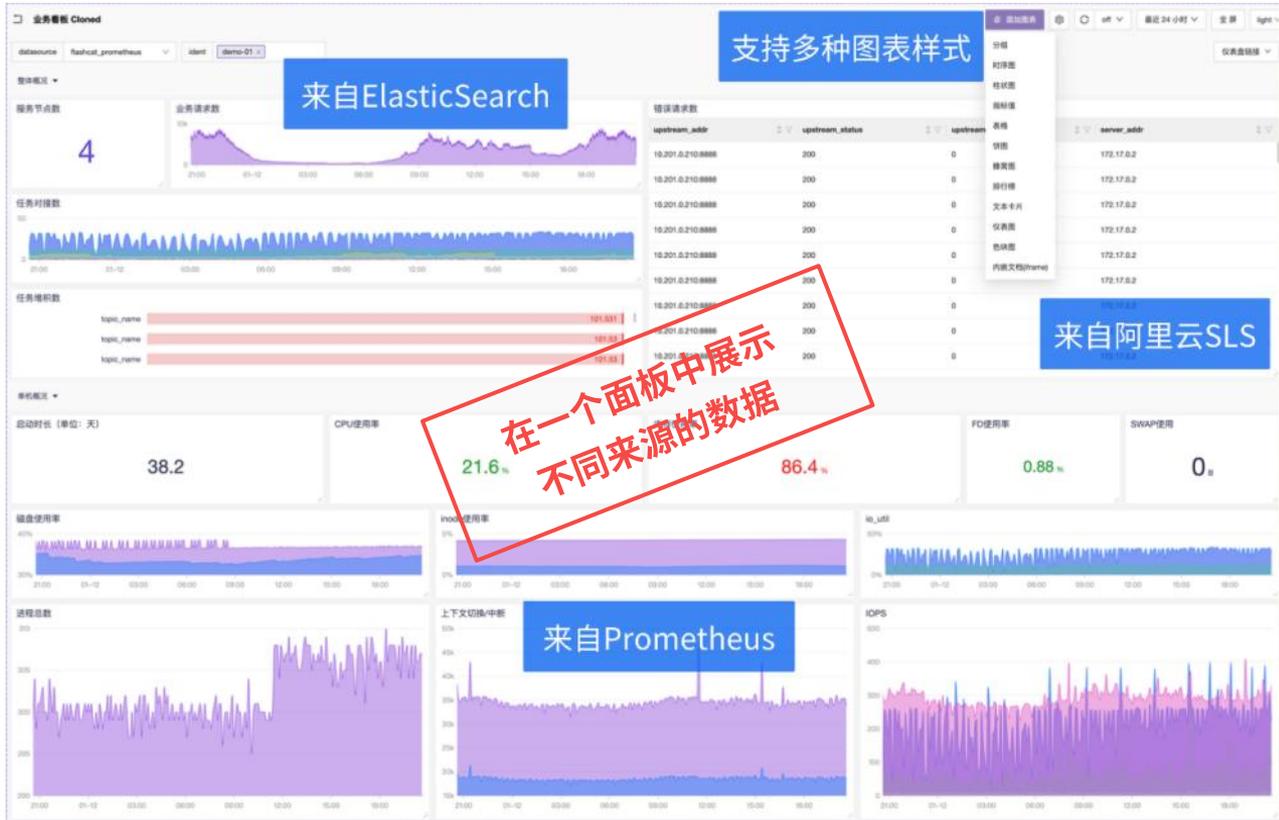
# 统一的仪表盘



自定义绘制拓扑图



支持导入Grafana仪表盘



# 针对各种数据源集中配置和管理告警策略



## 多集群Prometheus监控告警

规则配置

Metric Host Log Anomaly **Pro**

数据源类型 **关联数据源**

Prometheus **flashcat\_prometheus** 北京机

告警条件  级别限制

PromQL `elasticsearch_cluster_health_status(color="yellow") == 1`

触发告警:  一级报警  二级报警  三级报警

[数据预览](#)

PromQL `rate(cpu_usage_idle(ident="demo-01")[2m]) < 0.01`

触发告警:  一级报警  二级报警  三级报警

[数据预览](#)

执行频率 (s)  持续时长 (s)

生效配置

立即启用

生效时间        开始时间  结束时间

服务日历

## 阿里云SLS监控告警

告警策略配置

告警策略名称

告警策略类型

告警策略描述

告警策略表达式

告警策略配置

告警策略名称

告警策略类型

告警策略描述

告警策略表达式

告警策略配置

告警策略名称

告警策略类型

告警策略描述

告警策略表达式

## ElasticSearch日志监控告警

告警策略配置

告警策略名称

告警策略类型

告警策略描述

告警策略表达式

告警策略配置

告警策略名称

告警策略类型

告警策略描述

告警策略表达式

智能  
监控



规则名称: A slow query has occurred in Mysql within the last minute - catalog | 附加标签: alertname=MysqlSlowQueries | Flashcat-自运维

备注: MySQL server mysql has some new slow query

### 1 针对不同数据源，设置监控规则

规则配置 使用说明

数据源选择: 全部数据源 | 多Prometheus实例

规则模式:  普通模式 |  高级模式

告警条件: 启用变量 | 内置指标: `increase(mysql_global_status_slow_queries[1m]) > 0`

触发告警:  一级报警 (Critical) |  二级报警 (Warning) |  三级报警 (Info)

辅助配置: 数据预览

多数数据源类型

Cluster Health 健康度状态码

- Cluster Health delayed innodb页故障
- Cluster Health Pending task 阻塞
- Cluster Health aborting task 阻塞
- Cluster Health unassigned IO 阻塞
- Cluster Health 数据库状态告警

推送 Flashduty | 多种通知渠道和模板 | 3 设置告警通知方式

当前已配置了全民 Flashduty 推送 | 通知简介 使用文档

告警接收组: dingtalk wecom feishu mm telegram email feishucard tx-sms tx-voice ali-sms ali-voice lark larkcard xx内部-电话 xx内部-短信

应用恢复通知:  启用

报警时长 (秒): 0 | 重复通知间隔 (分钟): 60 | 最大发送次数: 0

告警自愈:  创建自愈模板 | 告警自愈

自愈模板: 选择模板 | 执行机器: 默认可留空, 也从事件中的 label 标签选择要执行的机器

Webhook

附加信息: recovery\_prompt elasticsearch\_cluster\_health\_status\_code

事件 relabel: 针对告警事件进行relabel | 2 自定义监控生效周期

生效配置 使用说明

立即应用

生效时间: 开始时间: 00:00 | 结束时间: 23:59

生效日期: 周一-周二 | 周三-周四 | 周五 | 周六-周日

服务日历: 第一组服务日历

仅在本业务组生效

精确告警事件中的 idset 归属关系判断, 即: 如果告警事件中有 idset 标签且 idset 对应的机器不属于该业务组, 则丢弃此告警事件

通知升级 | 告警升级和聚合

通知升级:  启用

持续时间超过 (分钟): 60 | 重复通知间隔 (分钟): 60 | 最大发送次数: 0

重新定义告警级别:  一级报警 (Critical) |  二级报警 (Warning) |  三级报警 (Info)

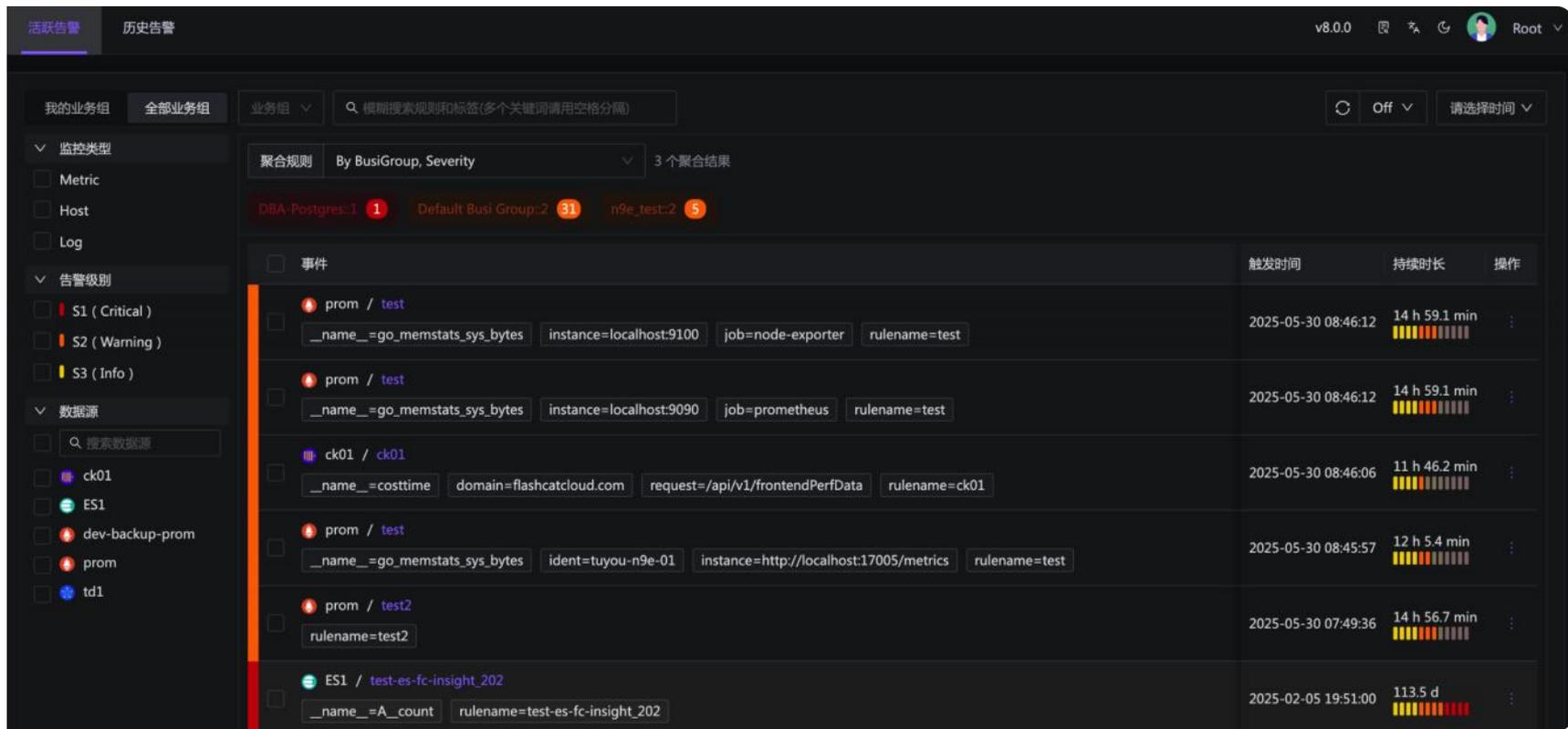
重新定义通知媒介:  dingtalk | wecom | feishu | mm | telegram | email | feishucard | tx-sms | tx-voice | ali-sms | ali-voice | lark | larkcard | xx内部-电话 | xx内部-短信

告警接收组

通知聚合:  启用

聚合发生等待时间 (分钟): 2

网络设备高级配置



# 监控告警大屏看板



# Flashduty | 统一告警On-call平台

# 告警 On-call 领域面临的挑战



01

## 告警散落各处 告警多

云上、云下多套监控系统，告警散落各处，没有集中的分派通知策略，缺少全局视图

03

## 告警遗漏

重要告警容易被淹没、遗漏；告警责任不到人，没有值班、备份、升级等机制，告警长时间无人响应

02

## 告警风暴

底层故障经常导致告警风暴，手机震动不止，影响故障处理，夜间尤其痛苦，工程师体验差

04

## 处理效率低 协同差

告警处理缺乏协同，过程不透明，信息难以共享，知识难以沉淀。不方便移动端查看处理

用户往往先于技术团队发现故障，久而久之，对 IT 的满意度持续走低

# 大厂 On-call 体验比拼



前蚂蚁金服员工[2]  

6 

当年去度蜜月全程背着15寸的电脑，以便随时响应oncall，登机起飞前的几分钟还被拉了个钉钉会议。回来后干了两件事，第一就是申请把电脑换成13寸的，15寸真的重；第二开始刷题换工作。

Amazon员工  

oncall到怀疑人生

Google员工  

按小时计费！

百度员工  

晚上接一个oncall电话，起来处理十几分钟，然后就睡不着了，第二天一天都废了，每次都这样，感觉身体受不了了

字节跳动员工  

真想请假写一天代码，每天被oncall搞死

Facebook员工 

要oncall，破事儿还贼多

微软员工  

4 

回复带带dai师兄：就这样啊，微软没有运维，所有运维都是工程师自己弄的，很多组都有

724oncall值班

蚂蚁金服员工[3]  

14 

现在理解什么是干电池被用尽了吧，我出去旅游什么都可以不带，但是电脑一定得带，在景区的时候都在应急。钉钉一响，整个人就跟惊弓之鸟一样

Google员工[3]  

2 

有的组专门招sre来oncall，但近几年sre的预算卡的紧，很多新的service都是swe在oncall。不过一般oncall都没什么事，还给双倍工资。。所以大家都抢着oncall 🤔

# 一站式告警On-call平台，加速企业告警响应



**Step 1**

### 告警集成

将各种不同监控工具发出的告警，实时的收集到统一的平台中，开箱即用

**Step 2**

### 标签增强

与CMDB等元数据中心打通，实时丰富告警标签和告警上下文

**Step 3**

### 聚合降噪

对相似的告警进行聚合、对频发的告警进行收敛，显著降低告警数量

**Step 4**

### 排班值班

灵活排班，满足日常值班、节假日值班、调班等场景，避免整个团队被频繁打断

**Step 5**

### 告警分派/认领/升级

集中管理告警分发策略，按时段、按标签灵活分发告警到不同的对象

**Step 6**

### 转派/协同

正确的时间通知需要的人，共享告警处理上下文信息，避免告警漏处理

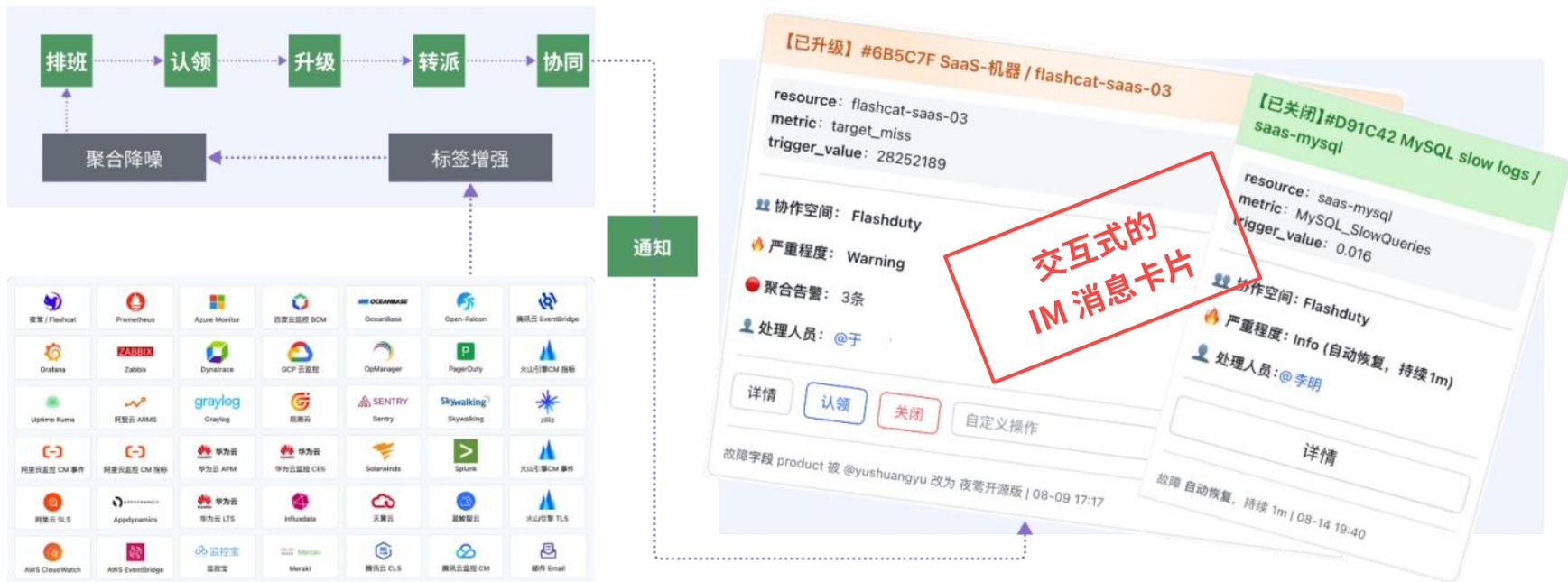
**Step 7**

### 告警通知

内置多种告警通知方式，及时、准确的通知相关人员，在IM中实时交付响应告警



# 一站式告警On-call平台，加速企业告警响应





# 对接各种告警事件



选择数据源

获取告警推送地址

修改 AlertManager Webhook 地址

集中查看和处理告警

```
receivers:  
- name: 'flashcat'  
  webhook_configs:  
  - url: '-替换为prometheus集成推送地址'  
    send_resolved: true  
    http_config:  
      proxy_url: 'http://proxyserver:port'
```

```
route:  
  ...  
  receiver: 'flashcat'
```

```
annotations:  
  timestamp: '{{ with query "time()" }}{{ . | first | value }}{{ end }}''  
  ...
```

# 数据增强，丰富告警上下文



## 历史变更事件

70% 的故障由变更导致



## CMDB 元数据

资产关系依赖映射



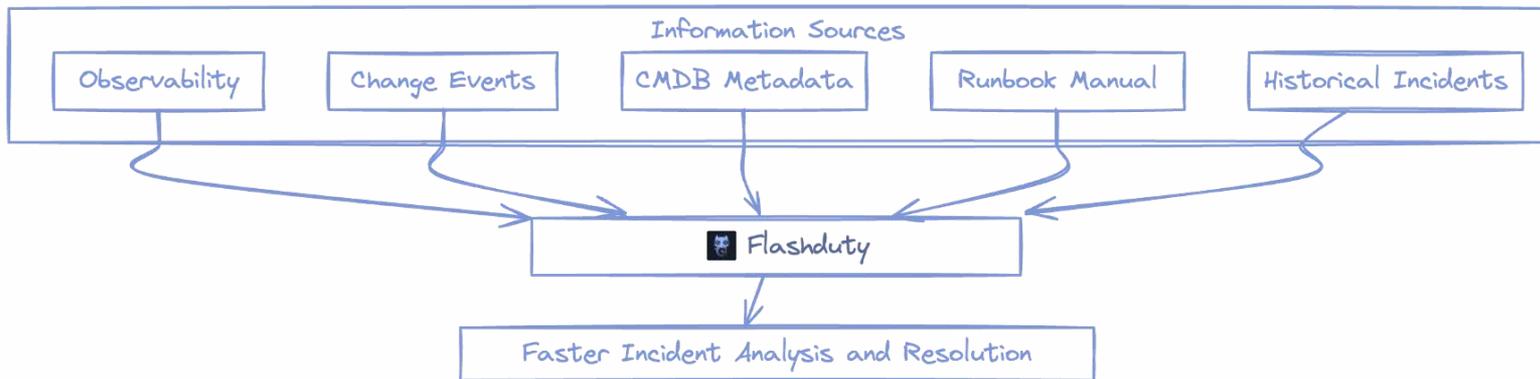
## 知识库和 SOP

在故障信息中展示 SOP



## 历史故障记录

参考相似故障的解决办法



# 告警降噪，显著降低告警数量



## 对相似告警进行聚合

- 事件 => L1 告警 => L2 故障
- 减少通知，避免告警风暴
- 规则聚合、智能聚合，至多降噪90%+

## 对频发告警进行收敛

- 避免狼来了效应
- 避免频繁被打断



## 故障收敛

60

分钟内，相同的故障发生

2

次及以上则进入抖动状态，未来

300

分钟内不再发送新的故障通知

# 排班值班，避免整个团队被频繁打断



1

## 满足各类场景

日常、节假日、调班、限时、公平轮换

2

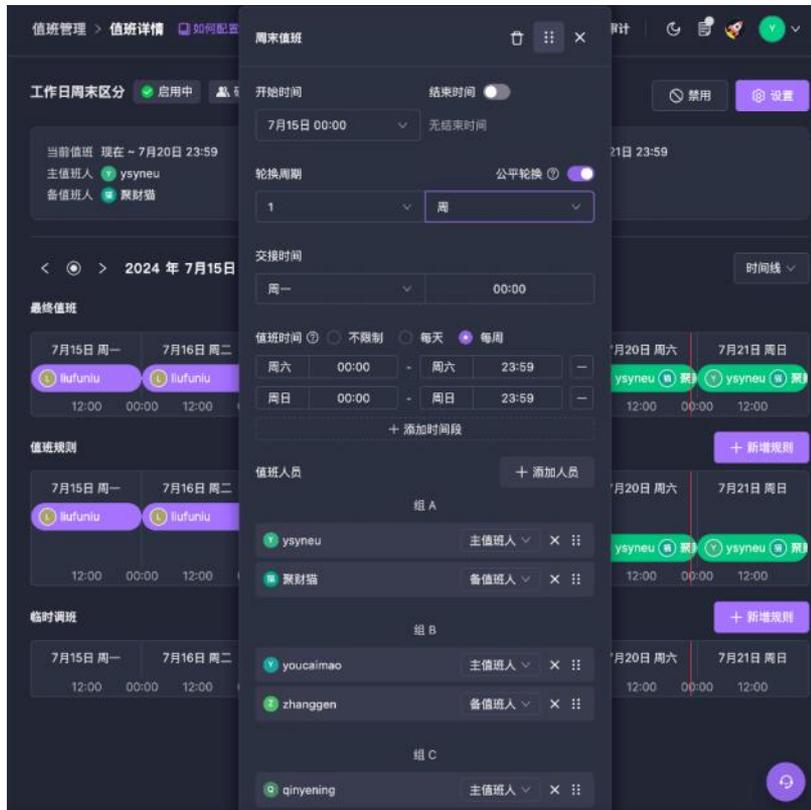
## 建立主备值班机制

支持多人同时，按角色值班

3

## 分派告警到值班人

不要随机分派，更不要分派到整个团队



# 告警升级和转派，正确的时间通知需要的人



- 1 制定升级路线**  
为故障制定清晰的升级路线
- 2 A: 确保问题得到响应**  
不因漏处理告警而引发更大损失
- 3 B: 从容应对紧急情况**  
发生紧急情况不慌张，有协作

**3 分派配置**  
分派策略支持设置多个环节，满足上下级、一二线之间的自动升级需求。升级行为等同于重新分派。

**环节1**

值班表 > 演示 + 团队 个人

以单聊渠道通知   遵循个人偏好  遵循统一设置 请确认已完成设置

以群聊渠道通知

循环通知设置

超过 30 分钟后如果故障  未关闭  未关闭且未认领，则升级到下一环节。

↓

**环节2**

值班表 团队 个人 > laimei

以单聊渠道通知   遵循个人偏好  遵循统一设置 请确认已完成设置

以群聊渠道通知

循环通知设置

超过 30 分钟后如果故障  未关闭  未关闭且未认领，则升级到下一环节。

业务线2 / 商品实时下单量-总量-偏离阈值(指标>0连续2次)

Critical 待处理 ID DB81FF 13小时1分 huaming 认领 关闭

故障概览 关联告警 时间线 历史变更

点击编辑内容，支持 Markdown 语法

qinxiaohui 发起评论 2分33秒前

HI, 数据库服务器的CPU有偶发性增高, 如图所示:

系统 触发了通知, 详情如下 13小时1分前

- 钉钉应用 通知到群组 告警测试(失败)
- 钉钉机器人 通过 65ab4b1598544dc904c3b6f8c707b395d2b4238f074601a26266ddac1a285f5 通知到群组

系统 通过分派策略 Default 的环节 1 分派处理人员为 huaming 13小时1分前

系统 触发新故障 业务线2 / 商品实时下单量-总量-偏离阈值(指标>0连续2次), 严重程度为 Critical 13小时1分前

# 通知，及时精准高效的触达干系人



手机App(安卓、iOS)

常见的接收告警通知方式都支持

- 默认告警模板 ● 启用中  
系统预警模板，不可更改
- > 飞书应用 (已设置)
- > 钉钉应用 (已设置)
- > 企业微信应用 (已设置)
- > 飞书机器人 (已设置)
- > 钉钉机器人 (已设置)
- > 企业微信机器人 (已设置)

> Telegram 机器人 (已设置)

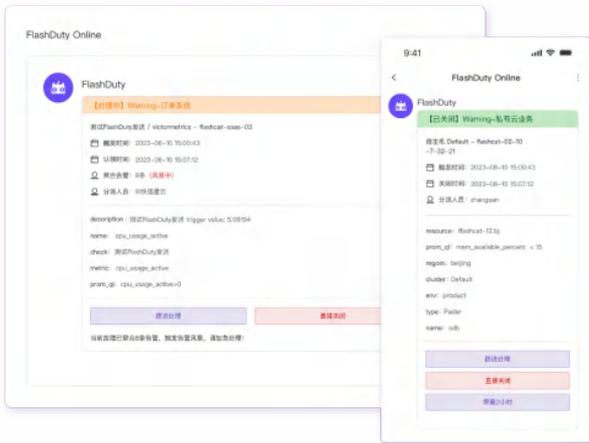
> Slack 机器人 (已设置)

```
1. <{{ifreason .}}ZMC <{{.DetailUrl}}#{{.Num}}> <{{tohtml .Title}}>
2. -----
3. 协作空间: {{if .ChannelName}}<.ChannelName>{{else}}无{{end}}
4. 严重程度: <{{.IncidentSeverity}}>
5. 触发时间: <{{date "2006-01-02 15:04:05" .StartTime}}>
6. 持续时间: <{{ago .StartTime}}>{{if gt .AlertCnt 1}}
7. 聚合告警: <{{AlertCnt}}条{{end}}<{{if Labels.resource}}>
8. 告警对象: <{{Labels.resource}}>{{end}}<{{if .Description}}>
9. 故障描述: <{{.Description}}>{{end}}<{{if gt (len .Responders) 0}}>
10. 分派人员: <{{range .Responders}}<{{.PersonName}}> <{{end}}>{{end}}
11. -----
12. </hr>{{.DetailUrl}} | 详情 <{{.DetailUrl}}> | 认领 </hr>
```

## 自定义消息的字段和形式

- > 短信 (已设置)
- > 邮件 (已设置)

## 交互式的 IM 消息卡片



# 自定义操作，集成 workflows



## API 集成

以 **按钮** 形式集成到控制台、IM 消息卡片

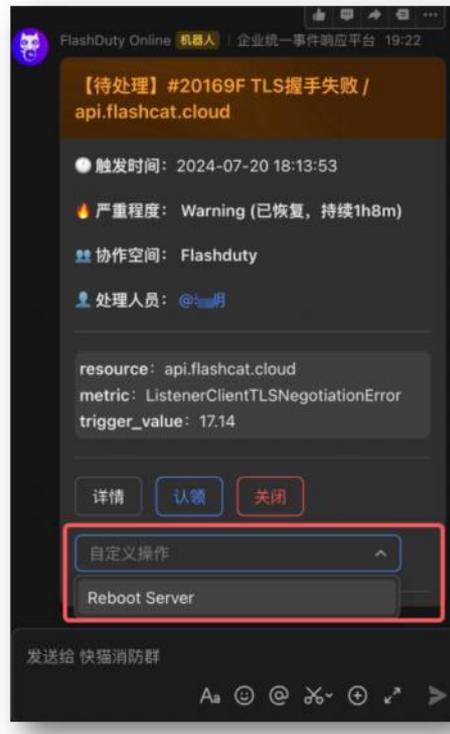
## 自动化流程

集成任何自动化、SOP 流程

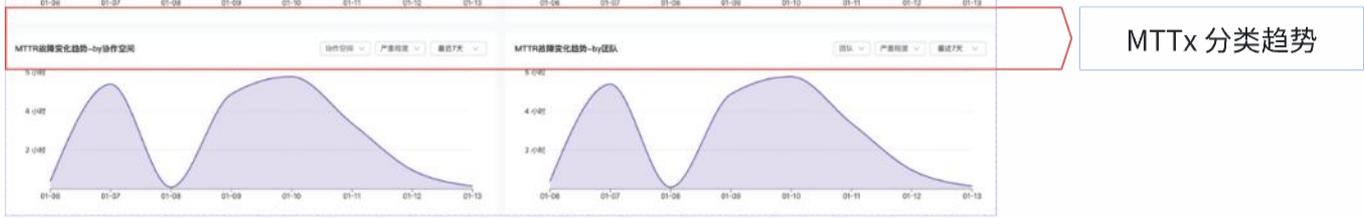
## 典型场景

- 重启主机
- 回滚变更
- AI 根因分析
- 一键拉群
- 发布 Status Page

IM(飞书、企微、钉钉、手机App)



# 数据驱动，推动告警处理流程持续优化



名称	数量	名称	数量
binlog同步延迟	226	PRC ipchat-04-02_RL_N_45	178
SaaS-HTTP请求失败	128	customers-ign-01	144
SystemDefaultDomainController_CPUUtilization	88	dev-flashops-02/03_R6.237129	77
连接mysql数据库超时	62	PRC ipchat-04-02_RL_N_43	48
请求数异常	44	api-.....-3rd	44

TopN 告警统计

姓名	接收告警数	认领告警数	关闭告警数	MTTA	MTTR
guyou	39	0	0	0秒	0秒
lining	46	44	43	7分40秒59	80分44分59
yuahu	79	4	2	14分19秒	26分42秒
WANG...	23	0	0	0秒	0秒

工作量统计

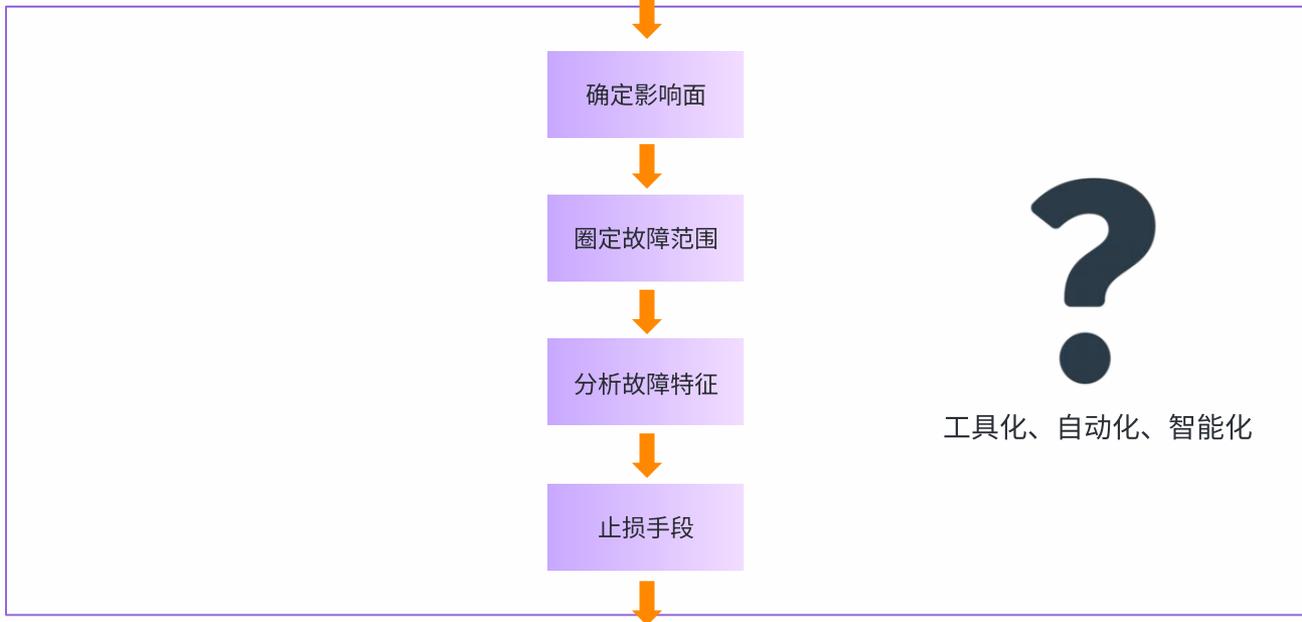
数据统计，通过数据驱动不断推动告警治理和优化

# Flashcat | 智能的故障发现定位平台

# 故障发现定位模型



主动/被动发现业务受损



流量调度

Traffic Control

单点切换

Single Point Switch Over

服务降级

Service Degradation

限流

Throttle Control

自动熔断

Auto Circuit Breaker

一键回滚

One-button Rollback

一键重启

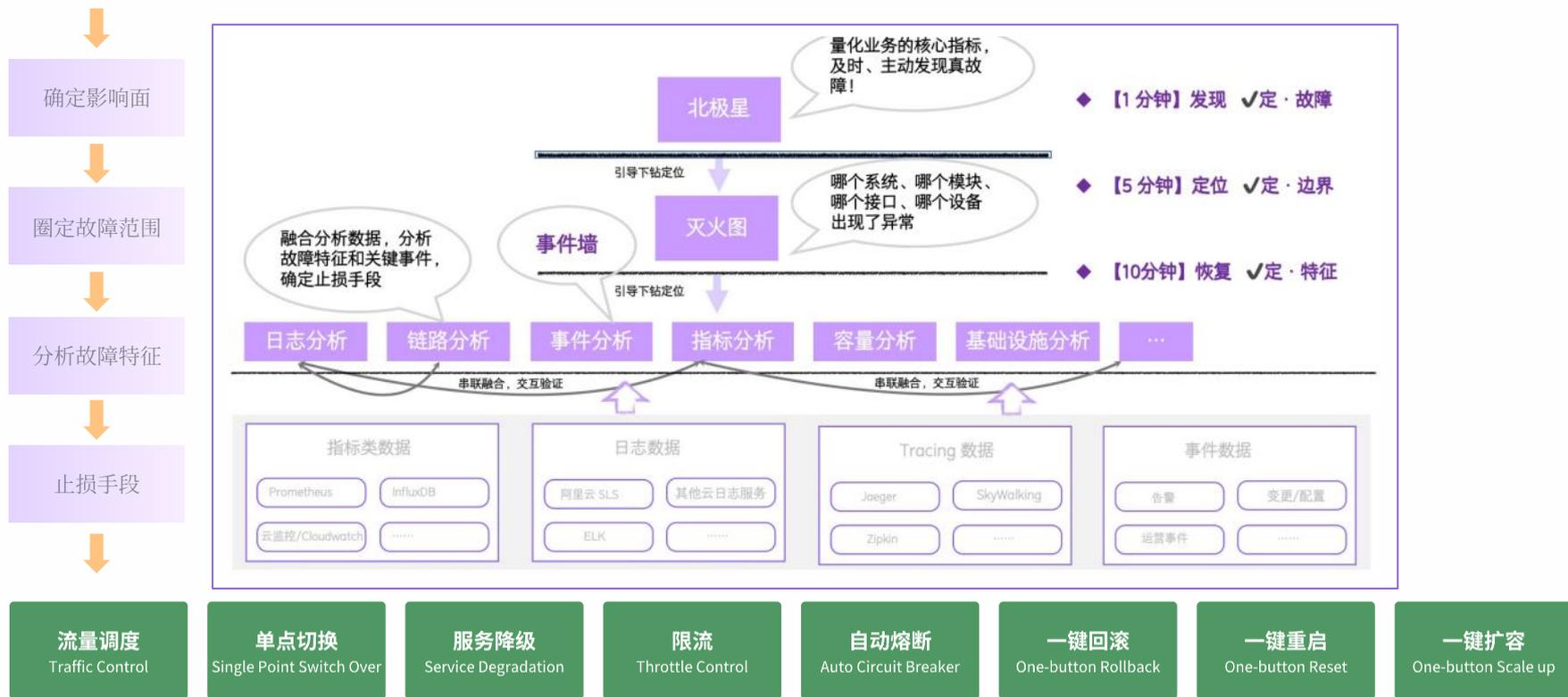
One-button Reset

一键扩容

One-button Scale up

# 故障发现定位模型

主动/被动发现业务受损



# Flashcat北极星——发现真故障，确定影响面和程度



✓ 北极星指标必须是公司上下人人都容易直观理解的，含义和重要性不言自明的。

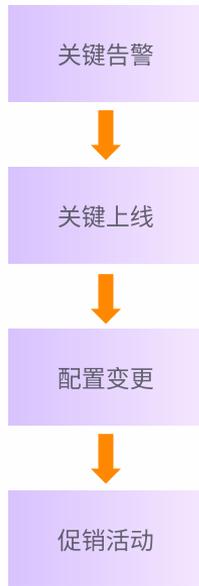
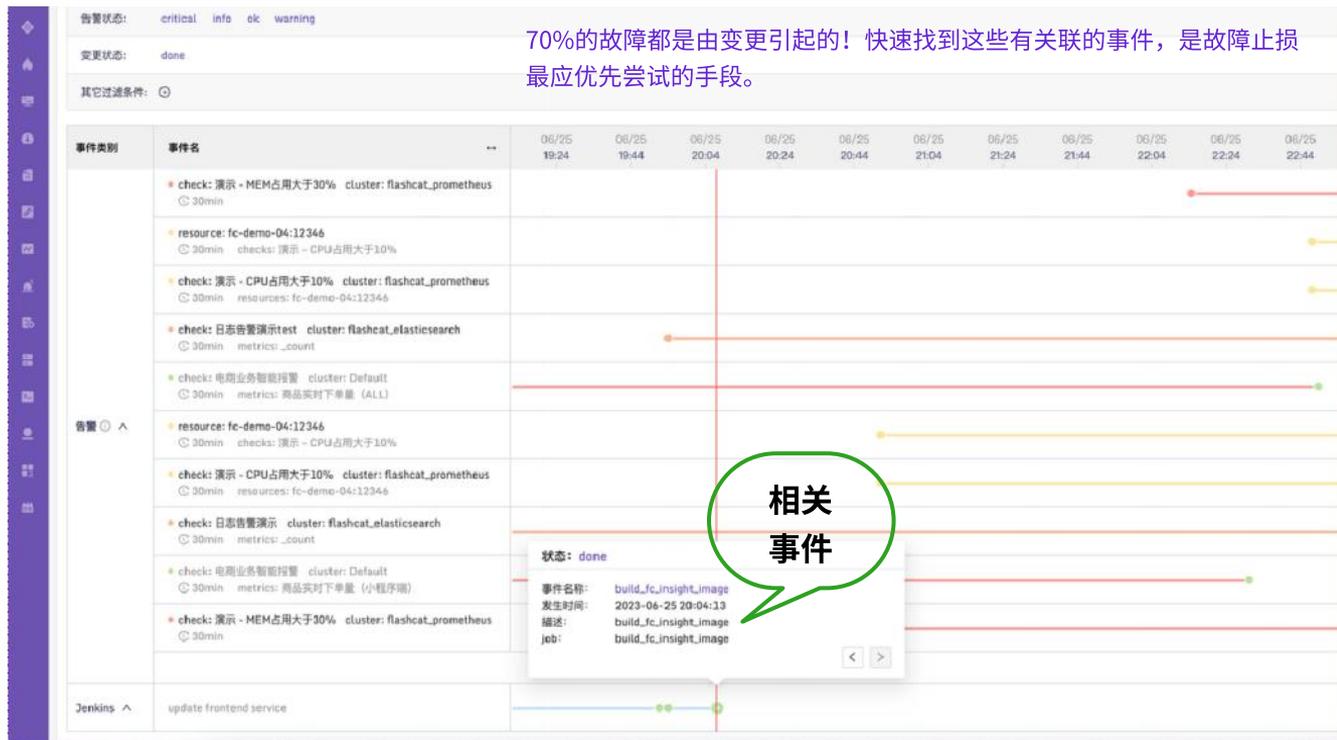
✓ 北极星指标的配置和生成应该是非常简便易得的。

✓ 北极星指标必须是实时的，这样才能第一时间发现业务受损的情况。

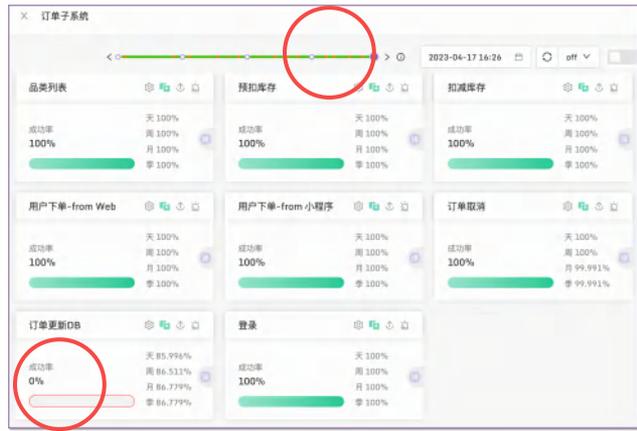
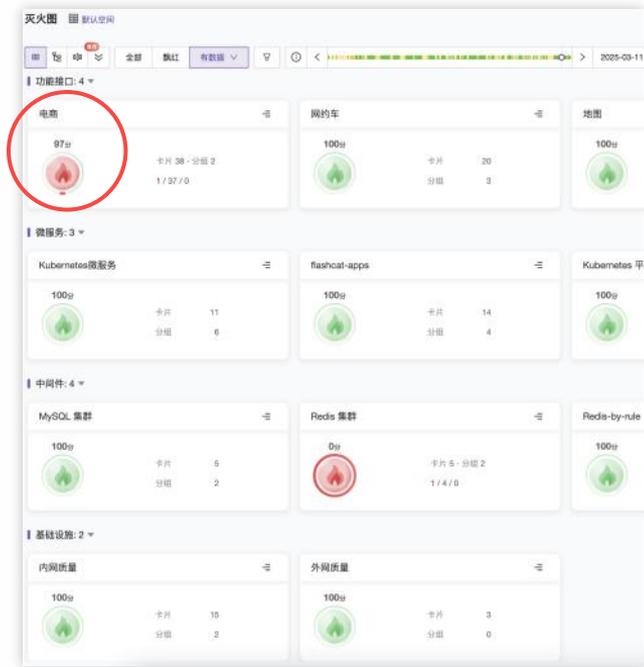
✓ 针对北极星指标的异常波动检测是要非常准确的，否则就是狼来了。

✓ 针对北极星告警，需制定明确的应急响应流程。

# Flashcat事件墙——快速确定或者排除相关事件



# Flashcat灭火图——圈定故障范围和对象



- Metrics
- Logs
- Traces
- Events
- Profiling
- Dashboard

- 1、不断的将更多的经验、知识、能力，沉淀固化到下钻链路上，让每位工程师都能拥有轻松调查故障的能力。
- 2、在一个平台上完成指标、日志、链路追踪数据的统一存储、可视化与分析。

# RUM —— 真实用户体验监控





# AI 分析



### 核心问题

Redis服务不可达: 日志及Trace指向Redis服务 10.99.1.4:6380 连接失败, 导致用户登录接口 /api/user/login 出现HTTP 500错误, 最终引发服务成功率暴跌至45.81%。

### 直接证据

1. 日志层:
  - account\_svr 持续报错 set session to redis failed dial tcp 10.99.1.4:6380: connect: connection refused
  - 错误时间分布密集 (5分钟内连续5次), 表明Redis连接持续不可用
2. Trace层:
  - 所有失败请求的根Span /api/user/login 均返回500状态码
  - Redis SET: 子Span明确记录连接拒绝错误, Span状态码为2 (错误)
3. 网络特征:
  - 目标地址为内网IP 10.99.1.4, 排除公网网络问题
  - 端口 6380 为非常用Redis端口 (默认6379), 需检查配置合理性

### 思考过程

好的, 我将在更分析这个服务接口成功率下降到45.83%的问题, 根据提供的日志和Trace信息, 首先看日志是否有来自account\_svr的错误10.99.1.4:6380, 错误是connection refused, 这说明Redis连接出现了问题。

接下来看Trace数据, 多个/api/user/login的请求返回了HTTP 500错误, 每个Trace中的Redis SET操作都失败了, 同样是因为无法连接Span状态码都是2, 也就是错误状态, 进一步确认了操作失败。

首先考虑网络问题, 可能是Redis服务没启动, 或者端口被防火墙拦截, 或者配置错误。比如, 检查Redis服务是否运行, 确认6380端口是否开启是否正确, IP和端口有没有写错, 还有可能Redis服务器资源耗尽, 比如CPU、内存或者连接数过多导致无法响应。

影响方面, 用户登录时的会话信息无法写入Redis, 导致登录状态无法保持, 用户可能无法正常使用需要认证的页面。虽然user\_logic\_svr没失败会导致登录流程中断, 返回500错误, 从而影响成功率。

建议的操作首先是检查Redis服务的状态, 看看是否被清除或者没有启动, 然后测试网络连通性, 用telnet或nc命令测试端口是否可达。如果网络没有异常, 同时检查服务器的资源使用情况, 确认是否有资源不足的问题。另外, 核对account\_svr的配置是否正确, 确保连接的IP和端口和其他节点是否一致。最后, 考虑增加监控和告警, 以便及时发现类似问题。

天小雷 发现异常: 成功事件(7742)

特征分析 云新

日志检索 发现异常: 检测到异常日志

Tracing 发现异常: 检测到异常Trace

仪表盘 发现异常: 检测到异常仪表盘

事件分析 发现异常: 登录事件 8

### 根因分析

以下重要事件可能与异常相关:

1. **TransService\_svr\_huM 和 local-redis\_huM 的多次故障报警** (状态为 failure), 这些失败可能由代码问题、依赖库版本或环境配置变化引起, 影响服务稳定性。
2. **事件触发时间相近** (如 1749115889 和 1749115884), 建立的失败时间可能表明共享依赖库存在共同问题。
3. **均由定时任务触发 (Started by timer)**, 定时任务的配置错误可能导致重复失败。

### 根因

时间	事件	组件
2025-09-05 10:53:35	Redis 事件	local-redis_huM
2025-09-05 10:59:37	Redis 事件	local-redis_huM
2025-09-05 10:59:41	Redis 事件	TransService_svr_huM
2025-09-05 10:59:42	Redis 事件	TransService_svr_huM
2025-09-05 17:31:23	Redis 事件	local-redis_huM
2025-09-05 17:31:24	Redis 事件	local-redis_huM

### 异常分析

#### 1. 异常现象

检测到多条来自 account\_svr 服务的 error 级别日志, 错误内容均为 set session to redis failed dial tcp 10.99.1.4:6380: connect: connection refused.

#### 2. 原因推测

- 网络/服务器不可达: Redis 服务 10.99.1.4:6380 无法连接, 可能是由于以下原因:
  - Redis 服务未启动或重启
  - 网络防火墙安全策略阻挡了 6380 端口
  - IP 地址或端口配置错误
- 资源服务异常: Redis 集群节点资源耗尽 (如 CPU、内存、连接数)

#### 3. 影响范围

- 会话 (Session) 信息无法写入 Redis, 可能导致用户登录状态失效、登录失败等问题。
- 虽然 user\_logic\_svr 目前处于创建成功, 但会话存储失败可能导致后续业务流程中断。

#### 4. 建议排查方向

- 检查 Redis 服务 10.99.1.4:6380 的存活状态及日志
- 使用 telnet 10.99.1.4 6380 或 nc -vz 10.99.1.4 6380 验证端口连通性
- 检查 account\_svr 的 Redis 连接配置 (IP、端口、密码等)
- 监控 Redis 服务的资源使用情况 (CPU、内存、连接数)

### 检查项

数据源: ai-dev-test-ops -> new-http-log

输出异常日志:

```
["level":"error","message":"/set session to redis failed dial tcp 10.99.1.4:6380: connect: connection..."]
```

### Tracing

发现异常: 检测到异常Trace

### 检查结论

### 分析结论

以下Trace存在异常, 原因如下:

1. **HTTP 500错误**: 根Span /api/user/login 的 http.status\_code 为500, 表示服务器内部错误。
2. **Redis连接失败**: 子Span Redis SET 中均出现错误事件 Redis SET failed: dial tcp 10.99.1.4:6380: connect: connection refused, 表明Redis服务 (IP: 10.99.1.4:6380) 无法连接。
3. **Span状态码异常**: 相关Span的 status\_code 为2 (错误状态), 进一步确认操作失败。

### 结论

所有异常Trace均因Redis服务不可用导致, 需检查Redis实例 (10.99.1.4:6380) 的网络连通性或服务器状态。

### 检查项

ai-dev-test-jaeger Service: user\_logic\_svr Operation: /api/user/login

疑似异常Trace: 共 5 个Trace 收起

Trace ID	Span ID	Component	Duration
6e22151b02d5577c	user_logic_svr: /api/user/login	123.88ms	
	user_logic_svr 1		
	account_svr 2		

Today 9:55:53am 14 分钟

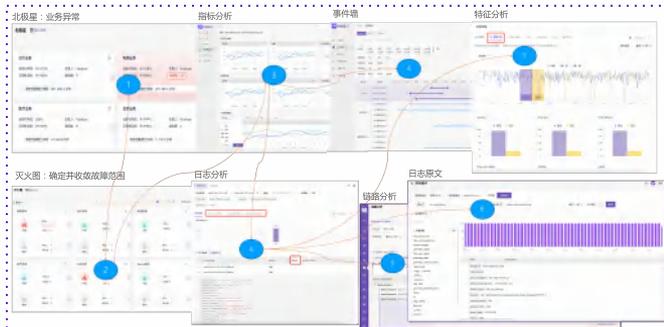
Trace ID	Span ID	Component	Duration
1c3e7ea5a0941a54	user_logic_svr: /api/user/login	79.58ms	

# 面向不同人群建立针对性的视图



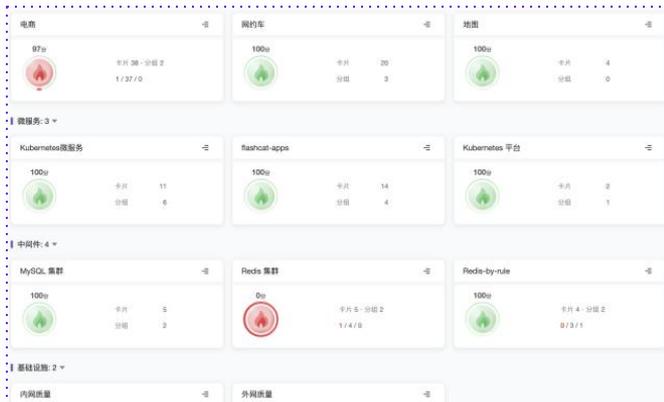
## 1、业务/管理层视图

业务状态、全局视图



## 2、研发视图

串联数据 排查问题 细节分析



## 3、运维视图

全局IT状态 故障/问题/告警视图

# 用户案例



醫院管理局  
HOSPITAL  
AUTHORITY



莉莉丝游戏  
LILYS GAMES



niu



中国铁路12306  
12306 CHINA RAILWAY



鹰角网络  
HYPERGRYPH



地平线科技



zenlayer



悠星网络



小马智行



## 医疗及连锁

香港医管局、高济健康、益丰大药房

## 餐饮连锁

星巴克、海底捞、金拱门、吉野家、零食很忙

## 游戏

莉莉丝游戏、悠星网络、鹰角网络、途游游戏

## 交通出行

小马智行、六分科技、真点科技

路特斯科技、保时捷、吉利汽车

嘀嗒出行、阳光出行、UU跑腿、八维通

## 科技

地平线科技、小牛电动、安克创新、影石Insta360

鹿客科技、Zenlayer、新钛云服、融云科技

## 金融

国泰君安期货、华盛证券、东莞证券

灵均投资、萨摩耶、顽岩资本

## 其他

海大集团、旭辉集团、中国电信、盛大、中免日上

知乎、Klook客路、作业帮、快看漫画、金山办公

# 附录

# 系统架构

# 水平扩展的部署案例



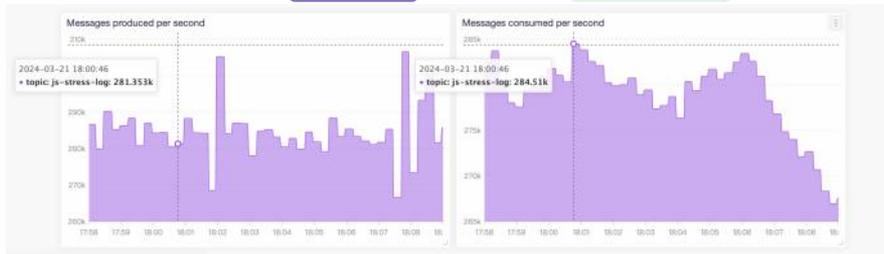
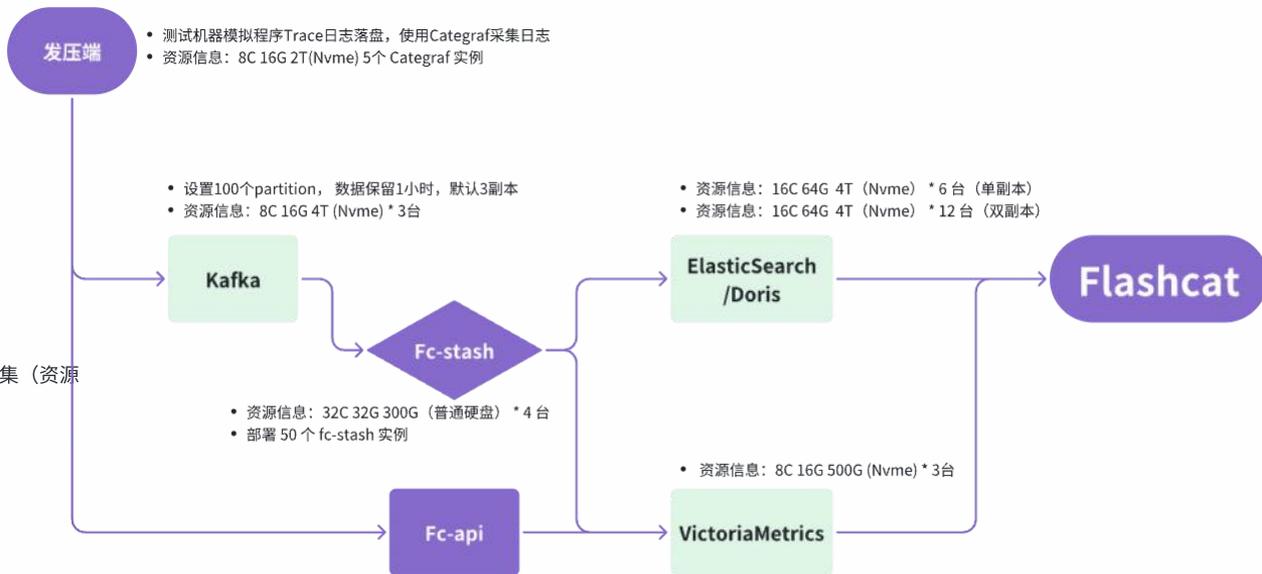
## 测试目标:

1. 支持峰值20万tps的数据写入
2. 支持每天15TB数据的处理和分析
3. 各组件均可水平扩展

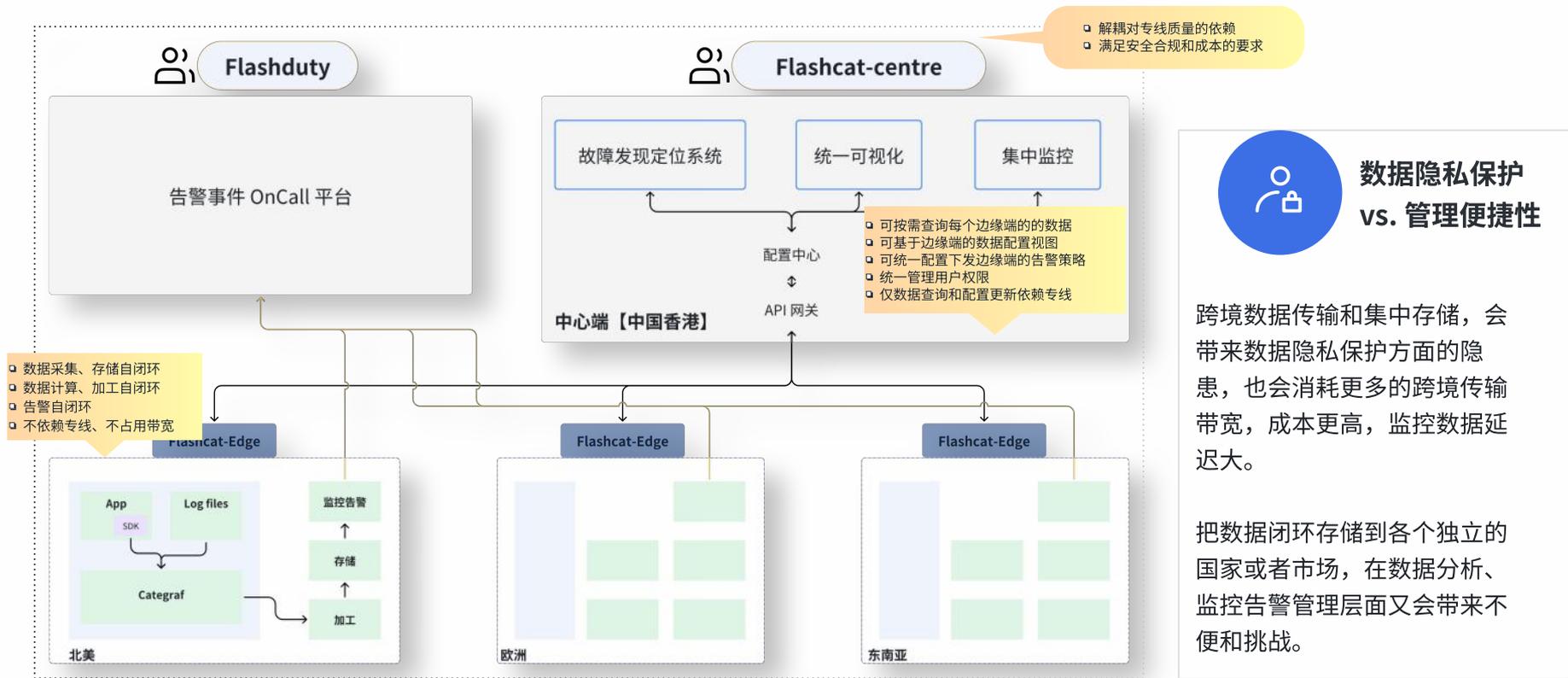
## 测试结论:

1. 各组件均可通过水平扩展增加来提升处理能力
2. 其中数据采集器Categraf单机达到了每秒5万条日志的采集（资源消耗3Core/300MB内存）
3. 最终系统处理数据的峰值达到了28万tps，并稳定运行

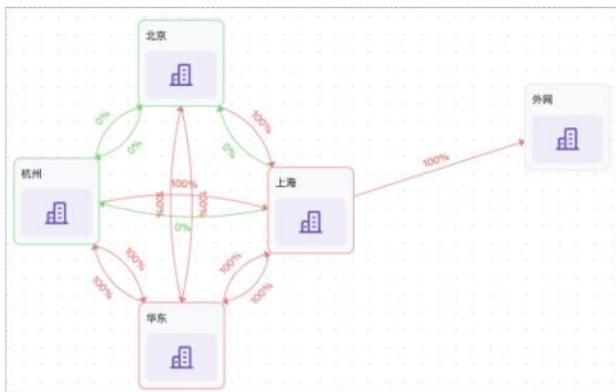
- ☆ 1. 在同等资源消耗的前提下，Categraf对日志的采集速度比 filebeat 提升了25%
2. 压测过程中，在继续增加资源的情况下，系统处理数据的峰值达到了58万tps



# 边缘部署模式



# 采集器增强 - Pingmesh



ICMP 延迟 丢包

源	目的	中国_香港	中国_北京	华北1_青岛	华北3_张家口	华北1_深圳
中国_香港	0.19ms	网络不通	网络不通	0.14ms	网络不通	网络不通
华北1_青岛	网络不通	0.16ms	0.23ms	网络不通	0.2ms	网络不通
华北1_北京	网络不通	0.16ms	0.16ms	网络不通	0.17ms	网络不通
华北3_张家口	0.18ms	网络不通	网络不通	0.27ms	网络不通	网络不通
华北1_深圳	网络不通	0.16ms	0.17ms	网络不通	0.14ms	网络不通

ICMP 延迟 丢包 请选择标签, 多个标签间的关系是"与"

拓朴 表格 配置 2024-04-10 19:56:08 off

源	目的	中国_香港			华北1_青岛			华北3_张家口	
		172.22.1.11/32	172.22.1.12/32	172.22.1.13/32	172.22.1.5/32	172.22.1.6/32	172.22.1.7/32	172.22.1.8/32	172.22.1.9/32
中国_香港	172.22.1.11/32	0.26ms	0.28ms	0.36ms					
	172.22.1.12/32	0.21ms	0.28ms	0.31ms					
	172.22.1.13/32	0.24ms	0.33ms	0.34ms					
华北1_青岛	172.22.1.5/32				0.24ms	0.34ms	0.35ms		
	172.22.1.6/32				0.17ms	0.25ms	0.18ms		
	172.22.1.7/32				0.16ms	0.2ms	0.15ms		
华北3_张家口	172.22.1.8/32							0.24ms	0.17ms
	172.22.1.9/32							0.23ms	0.4ms
	172.22.1.10/32							0.28ms	0.4ms
	172.22.1.2/32		1.19ms		0.17ms				

无数据 正常(0, 0) 缓慢(0.2, 0.9) 较差(0.9, 1.2) 很差(1.2, max)

1 中国\_香港 网络探测 / 中国\_香港

ICMP 延迟 丢包 ping

2025-01-02 15:01:48 off

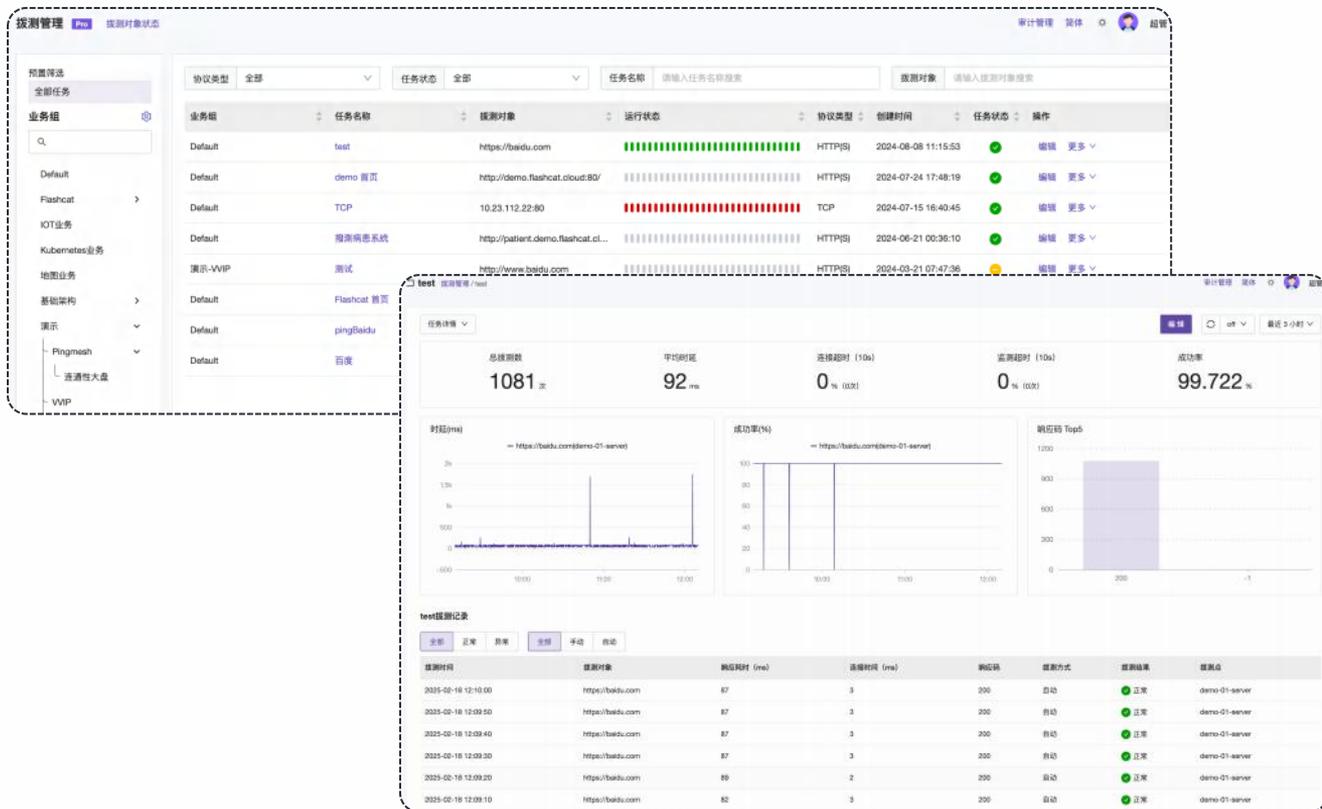
源	目的	rack1#1	rack1#2
rack1#1		0%	0%
rack1#2		0%	0%

ICMP 延迟 丢包

2024-03-26 13:29:57 off

源	目的	172.22.1.2/32	172.22.1.3/32	172.22.1.4/32
172.22.1.2/32		0%	0%	0%
172.22.1.3/32		0%	0%	0%
172.22.1.4/32		0%	0%	0%

# 采集器增强 - 网络拨测



## 多协议

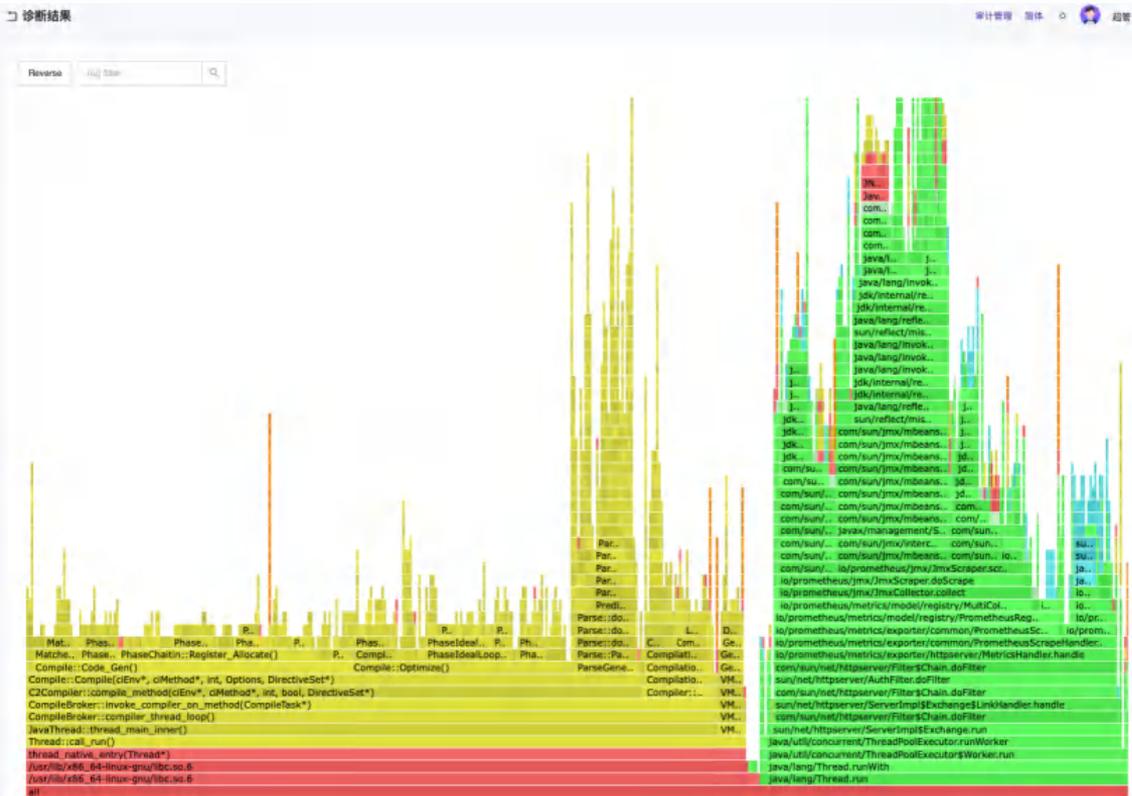
HTTP、TCP、UDP、  
ICMP、WSDL

针对 Reponse 做匹配

## 多拨测点

可选择安装了Categraf  
的一个或多个设备

# 采集器增强 - 持续性能剖析



针对Java等技术栈进行性能分析，生成火焰图

# 谢谢

<https://flashcat.cloud>