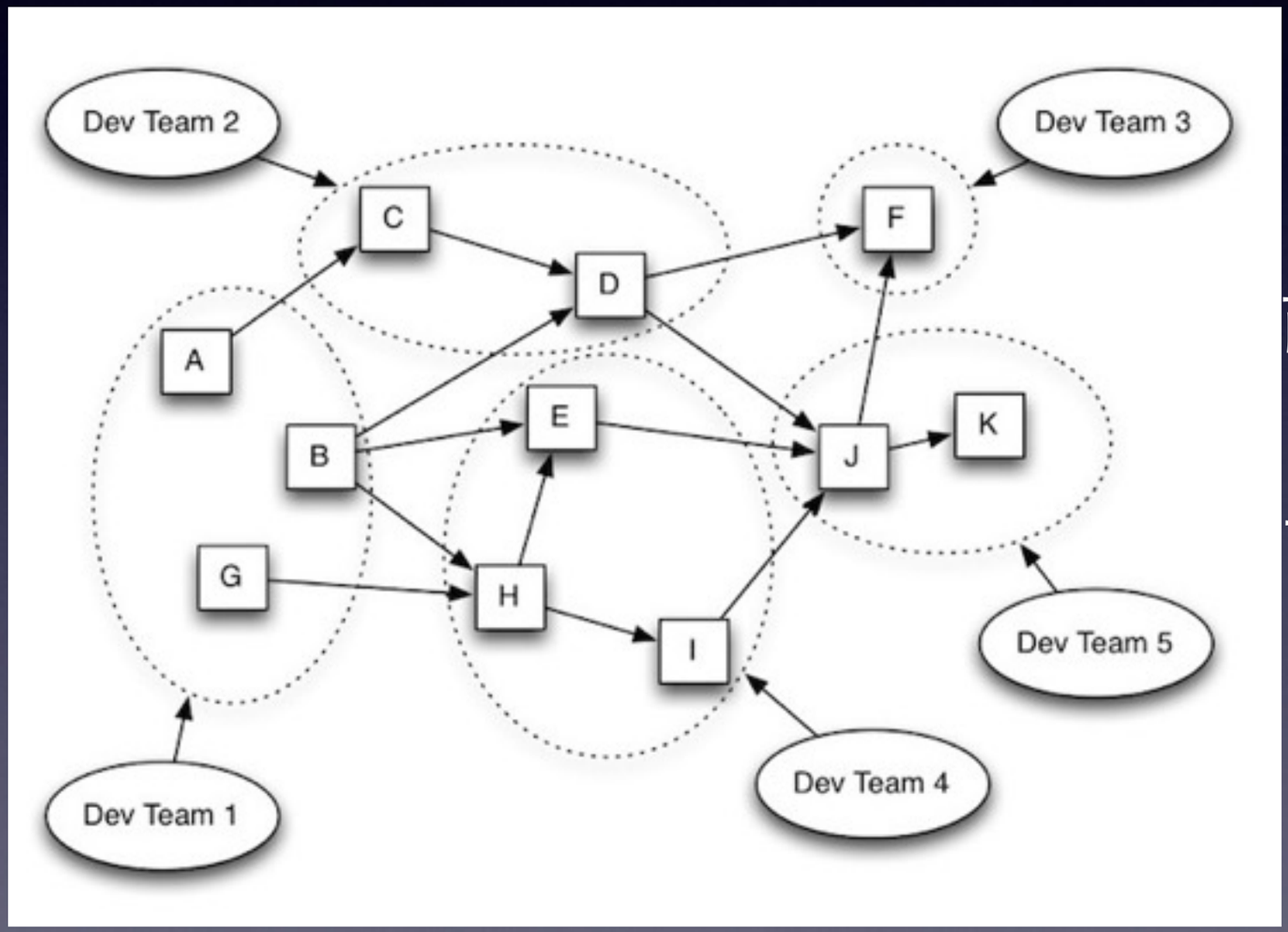
构建微服务体系下的全链路监控体系

姚捷@唯品会


一下微服务吧

微服务监控玩的怎样？

微服务长什么样?



而自治

立实体

同工作

微服务（对比单体应用）的优势

- 技术异构
- 弹性
- 可扩展性
- 简化部署
- 可复用性
- 可替代性

微服务的技术特性

基础组件众多

RPC框架（Client，Server）

治理组件 Local&Remote Proxy

服务注册中心，配置中心

安全组件，服务网关

异步消息系统

部署模式多样

单一服务，单一服务器

单一服务，多个服务器

多个服务，多个服务器

跨机房部署，容器化部署

治理功能强大

服务路由（上线／下线，临近机房）

负载均衡（Round Robin，权重）

自我保护（过载保护，服务降级）

当全链路监控系统遇到微服务，怎么玩?

我们要监控什么?

服务画像

我们要监控什么?

服务画像

① 服务概览信息

② 服务性能指标

③ 服务拓扑关系

④ 服务调用链

⑤ 版本信息

⑥ 服务治理状态

⑦ 组件内部状态

服务概览信息

服务名称

服务部署在哪些机房

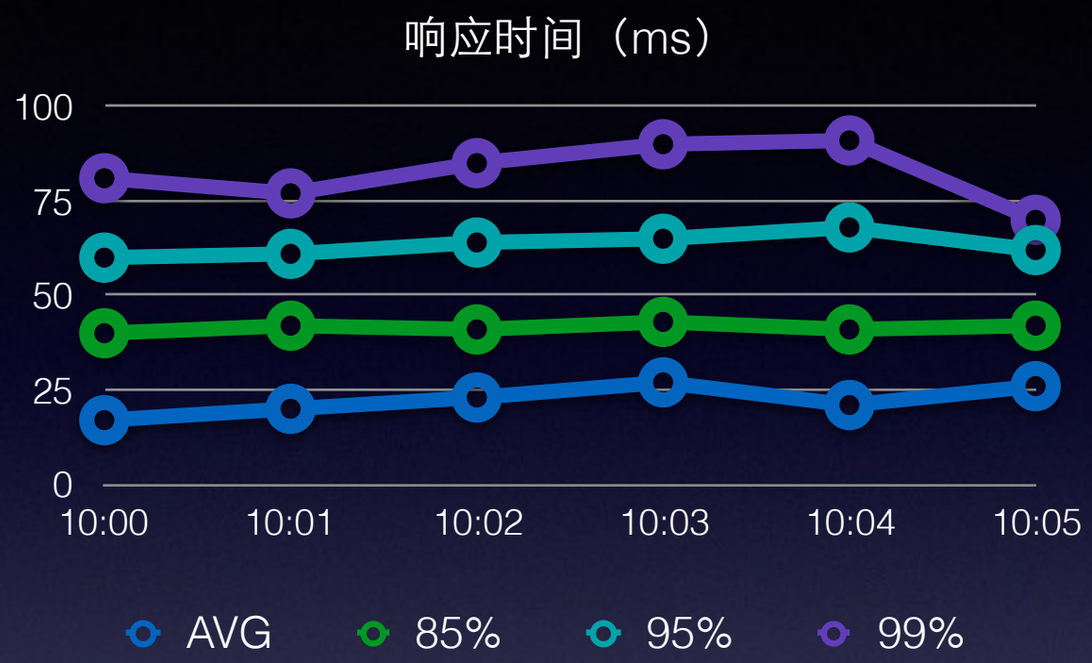
服务部署在哪些主机

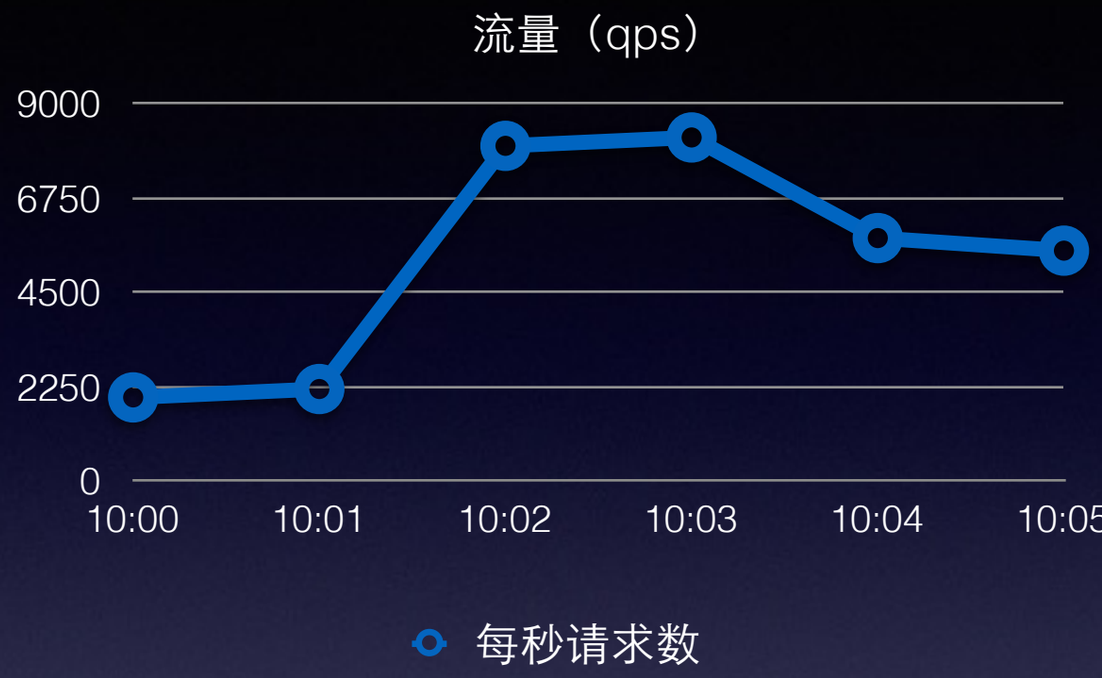
服务包含哪些API

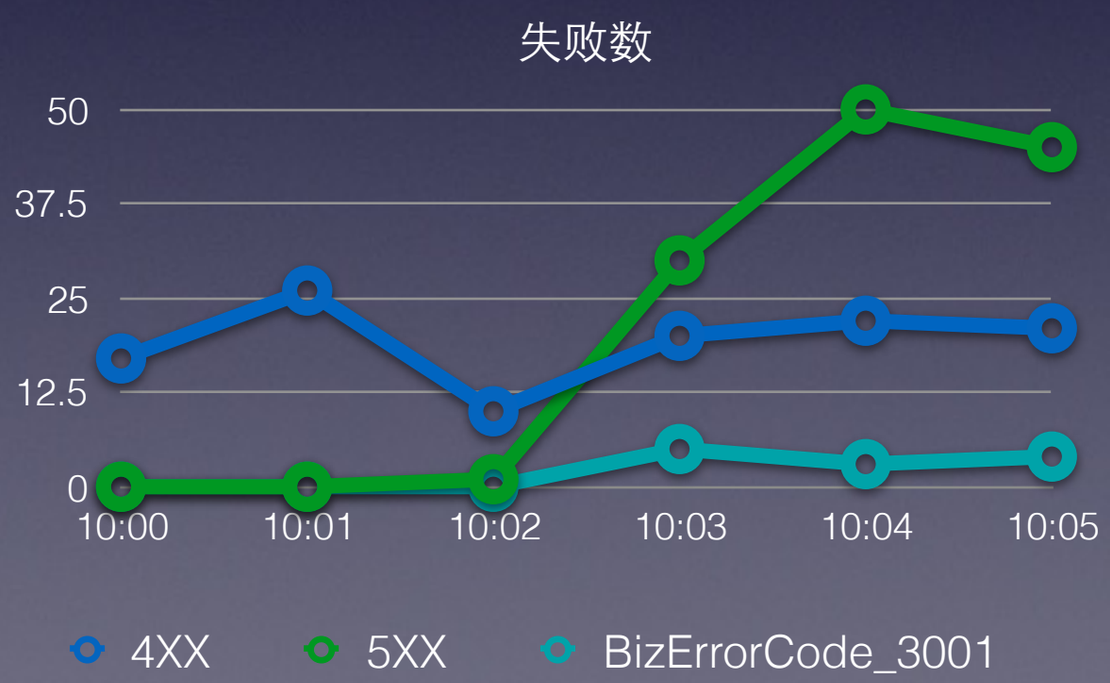
服务的相关配置信息

服务的负责人，开发人员，运维人员是谁

服务性能指标


响应时间（ms）
100
75
50
25
0
10:00
10:01
10:02
10:03
10:04
10:05
AVG
85%
95%
99%


流量（qps）
9000
6750
4500
2250
0
10:00
10:01
10:02
10:03
10:04
10:05
每秒请求数


失败数
50
37.5
25
12.5
0
10:00
10:01
10:02
10:03
10:04
10:05
4XX
5XX
BizErrorCode_3001


异常日志数
100
75
50
25
0
10:00
10:01
10:02
10:03
10:04
10:05
日志数量

服务性能指标

精度

秒级 or 分钟级

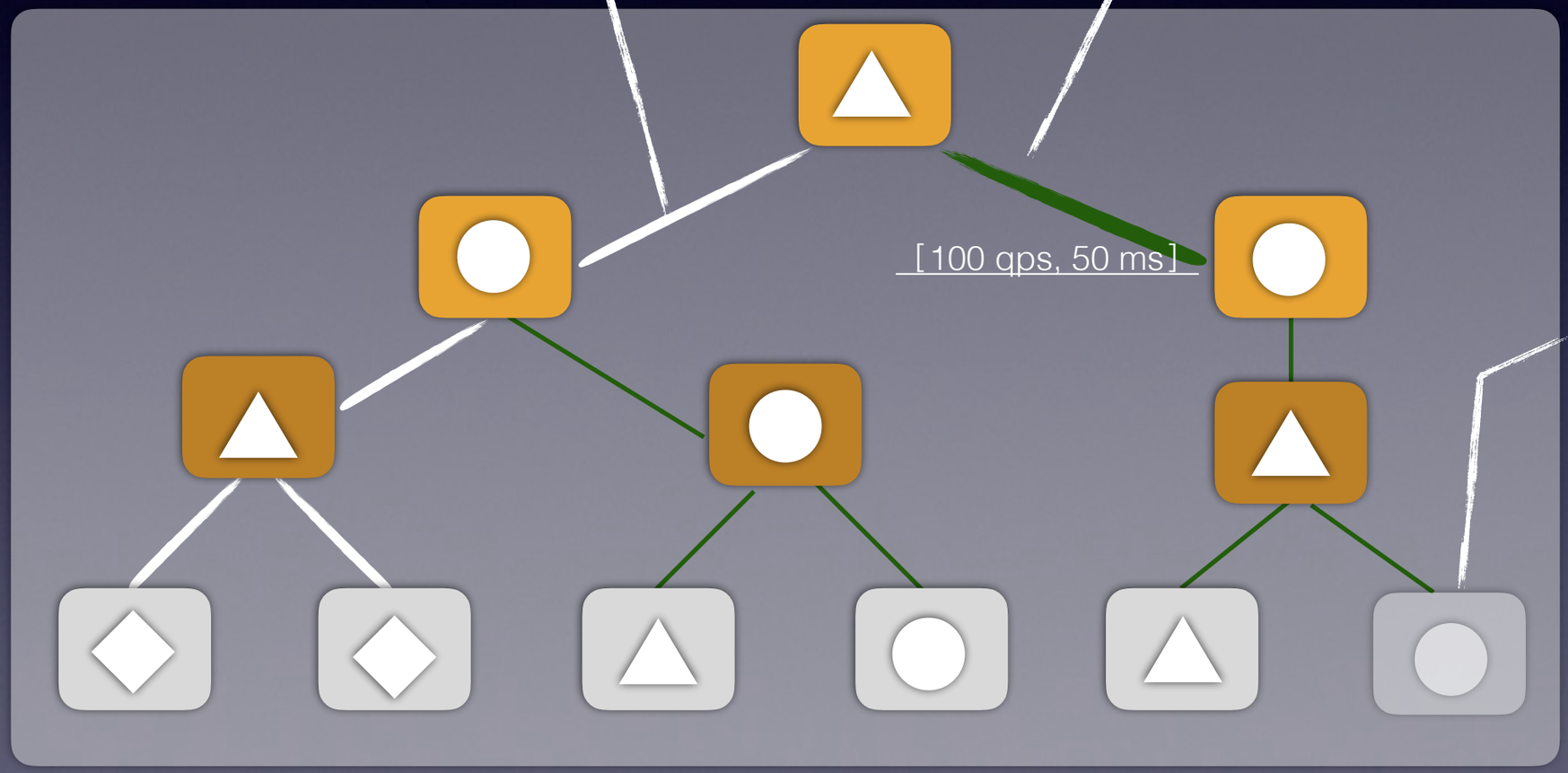
维度

- ✓ App
- ✓ IDC
- ✓ Host
- ✓ API

服务拓扑关系

☑服务之间的调用关系

☑A call B服务调用关系上的性能指标



☑体现服务强弱依赖

手工维护 → 自动化

服务调用链

查询订单号是XXX的调用链

查询耗时超过N毫秒的慢调用

调用结果

附加业务字段

返回码

调用耗时

异常日志

查询返回码是500的调用链

出入参数

版本信息

“RPC客户端v2.5.1版本发现bug，
需要立即发布v2.5.2修复并升级”

哪些服务正在使用RPC客户端v2.5.1？

组件版本

“好不容易把RPC客户端升级到
v2.5.1并发布到生产环境了”

怎么知道生产环境RPC客户端已经升级到v2.5.1？

组件版本

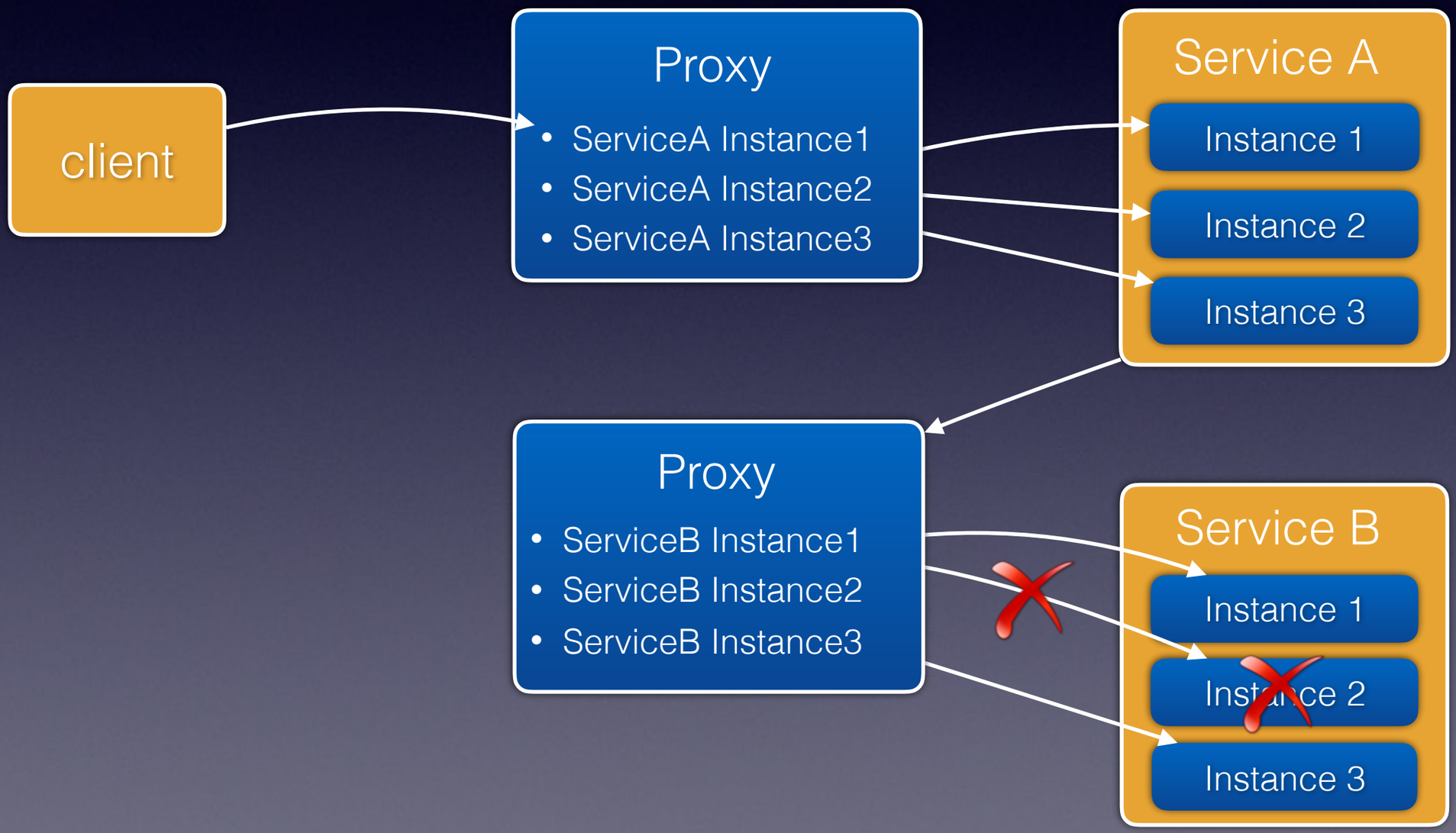
“服务v1.0.1版本灰度发布到了
GD6-OSP-CATE-API这台主机”

怎么知道这台主机的服务已升级到v1.0.1？

服务版本

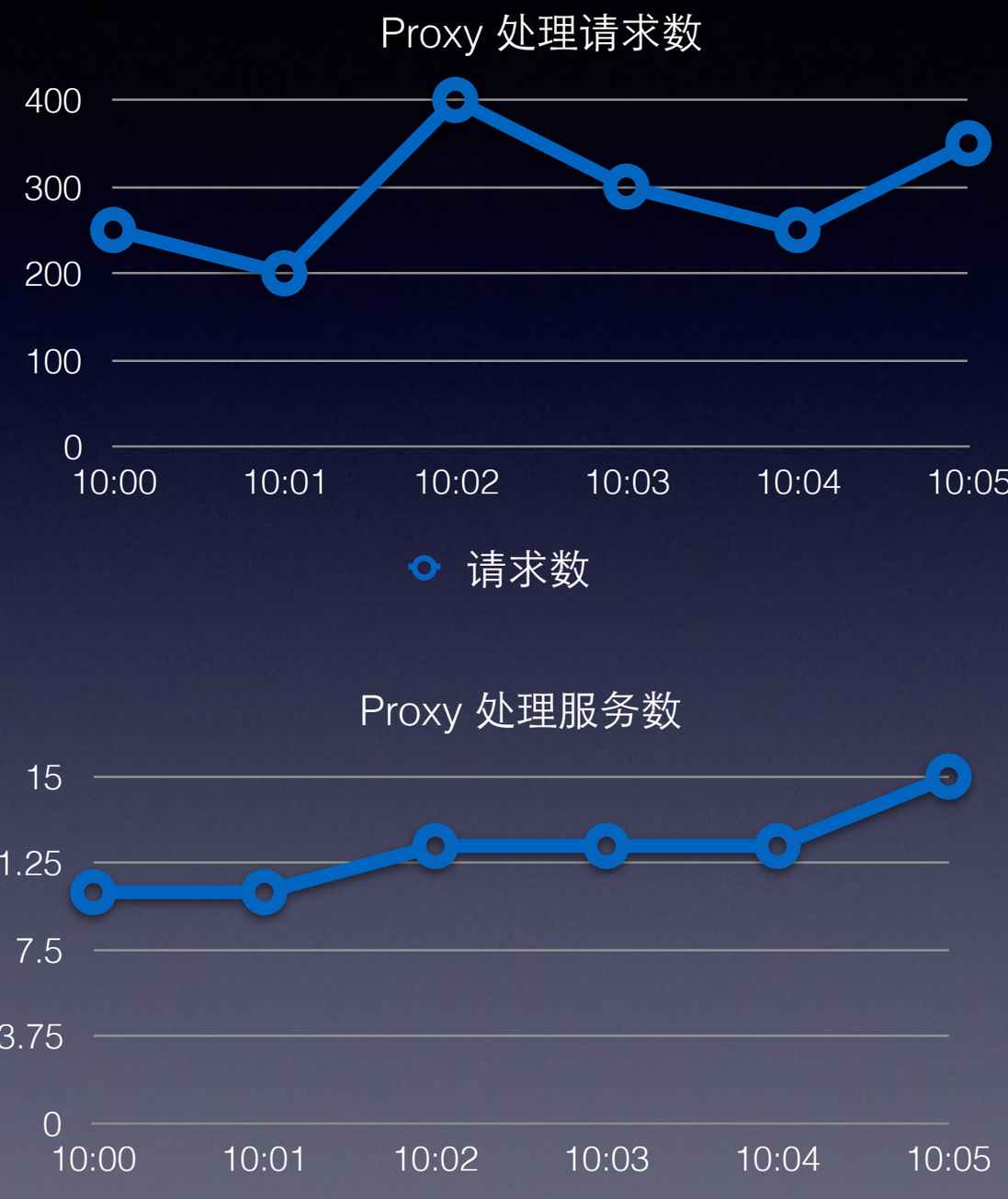
服务治理状态

熔断


client
Proxy
• ServiceA Instance1
• ServiceA Instance2
• ServiceA Instance3
Service A
Instance 1
Instance 2
Instance 3
Proxy
• ServiceB Instance1
• ServiceB Instance2
• ServiceB Instance3
Service B
Instance 1
Instance 2
Instance 3

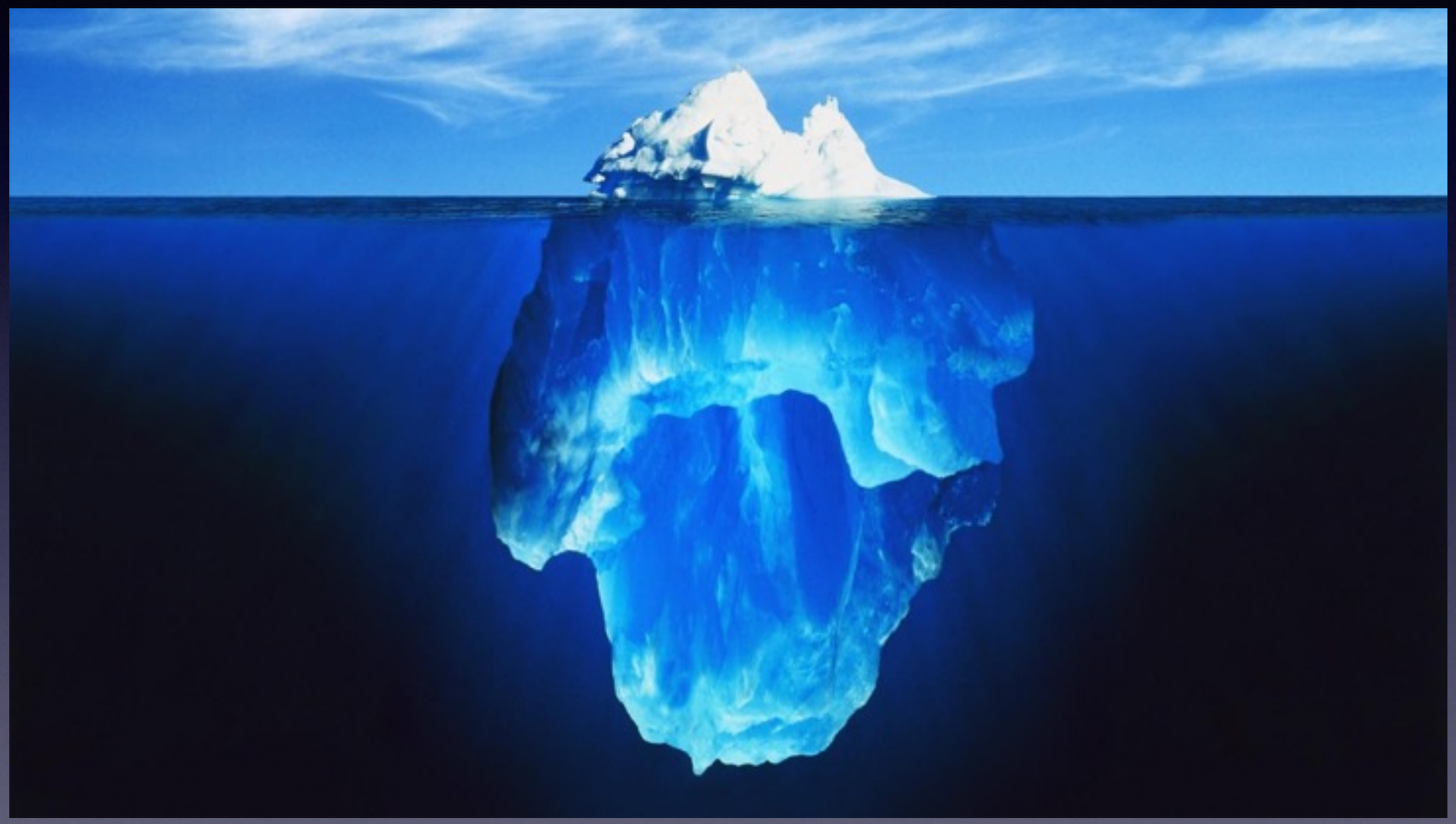
组件内部状态


Proxy 活动线程数
50
37.5
25
12.5
0
10:00
10:01
10:02
10:03
10:04
10:05
bossgroup
workgroup


Proxy 处理请求数
400
300
200
100
0
10:00
10:01
10:02
10:03
10:04
10:05
请求数

Proxy 错误重试次数
30
22.5
15
7.5
0
10:00
10:01
10:02
10:03
10:04
10:05
错误重试次数

Proxy 处理服务数
15
11.25
7.5
3.75
0
10:00
10:01
10:02
10:03
10:04
10:05
处理服务数



核心日志数据模型

Trace Model

Event Model

Log Model

Metric Model

Trace Model

属性名	描述	示例
Trace ID	调用链唯一标识，通常采用UUID的算法生成，并且需要端到端透传到下游	47484600805474672 37
Span ID	调用链Span唯一标识，通常采用UUID算法生成，并且需要透传下游	6283422850509101259
Parent Span ID	当前Span的父Span ID，用来维护Span的父子关系	9748460010547467231
Span Type	表示当前Span的属于那种服务类型，例如Dubbo Server，OSP Server等	OSP_Server
Service Name	调用的服务名称	com.vip.categoryService
API Name	调用的服务的API的名称	getCategories
APP	服务所在业务域名称	osp-category.vip.com
Host	主机名	GD6-OSP-CATE-API
Server Received Time	服务端接收请求的时间戳	1437835095689
Server Send Time	服务端响应请求的时间戳	1437835097405
Duration	服务端处理耗时	1716
Sample Rate	采样率，由顶层服务设定，透传到下游服务	0.1
Server Result	服务调用结果	success
Response Code	服务调用返回响应码	200
Custom Fields	附加业务字段，用来做调用链检索	orderId=0001&userId=2001

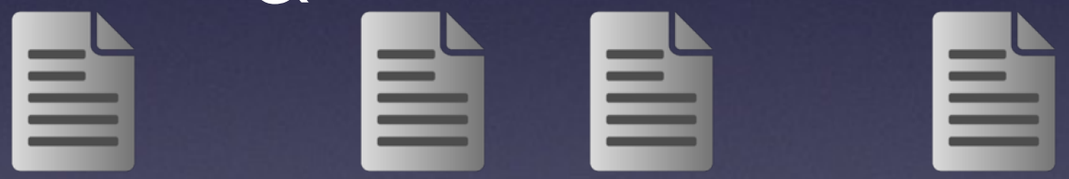
Event Model

属性名	描述	示例
Type	事件类型	CircuitBreaker.Open
Name	事件名称	CircuitBreaker.Open
Service Name	事件发生所在服务	com.vip.categoryService
App	事件发生所在业务域	osp-category.vip.com
Level	事件等级	critical
Timestamp	事件发生时间	1452735137965
Title	事件标题	com.vip.infrastructure.qa.te
Message	事件详细描述	com.vip.infrastructure.qa.te
Host	事件发生所在主机	GD6-OSP-CART-API16
Instance	事件发生所在应用实例名	osp-proxy16857
Module	事件发生所在的应用模块	osp-proxy
Tags	服务端处理耗时	蓝调blue, 大众喜爱，微博达人
ttl	采样率，由顶层服务设定，透传到下游服务	3600
Custom Fields	自定义字段列表,k／v键值对，k的名字可任意	orderId: '12345'

日志采集方式

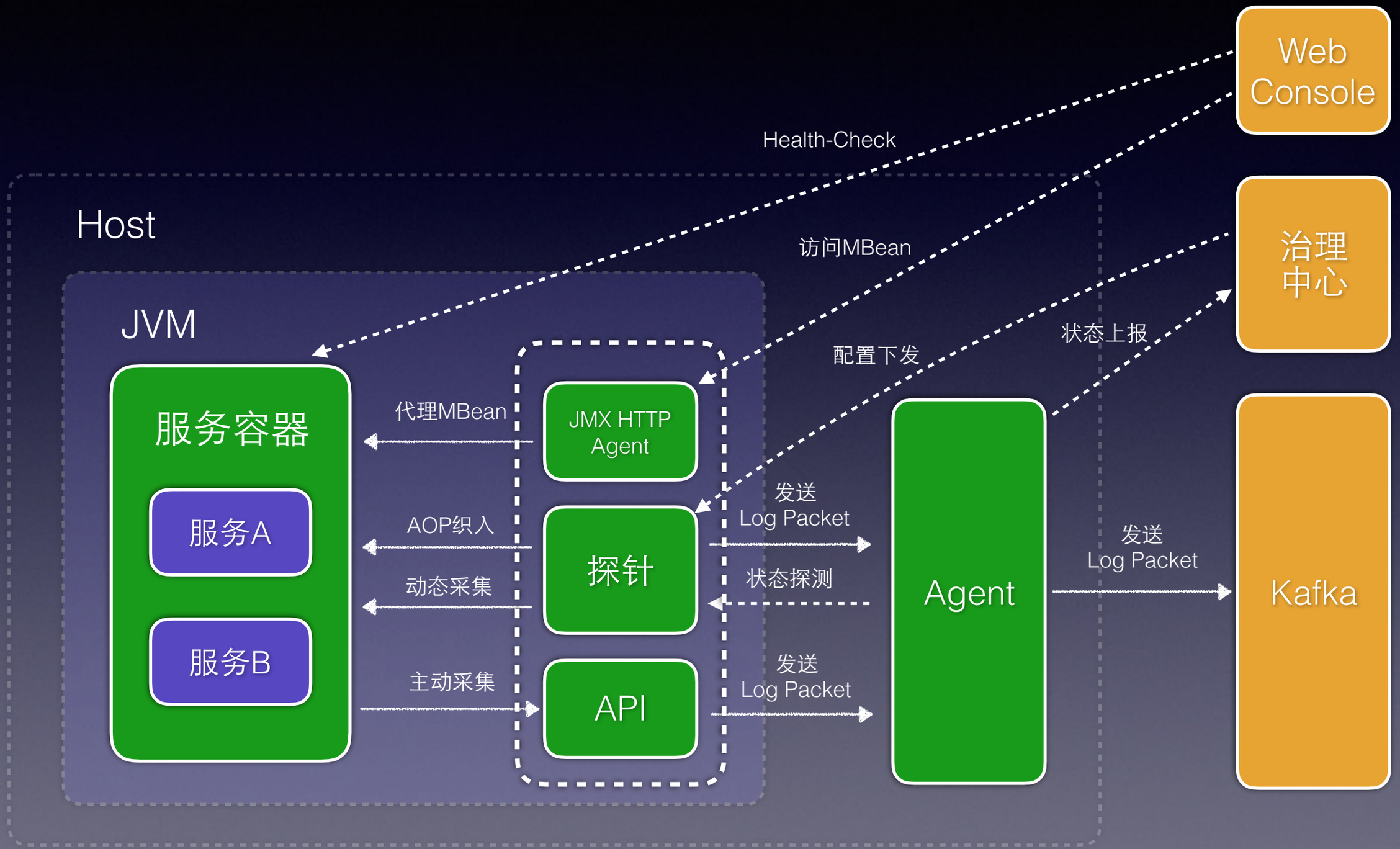
API主动埋点采集动态织入埋点采集

&



Event Log　Metric Log　Trace Log　Error Log

客户端（Java）监控数据采集架构



序列化协议

JSON

- ☑ 易读
- ☐ 冗长
- ☐ 解析性能差

VS

AVRO / PB

- ☐ 二进制消息
- ☑ 高压缩比
- ☑ 高性能

本地日志传输

Logging to file

- ☑ 磁盘高性能
- ☑ 语言无关
- ☐ disk io瓶颈
- ☐ 绑定log框架

VS

UDP

- ☑ Send – and – Forget
- ☑ 协议格式简单
- ☑ 无本地磁盘IO
- ☑ 脱离Log框架

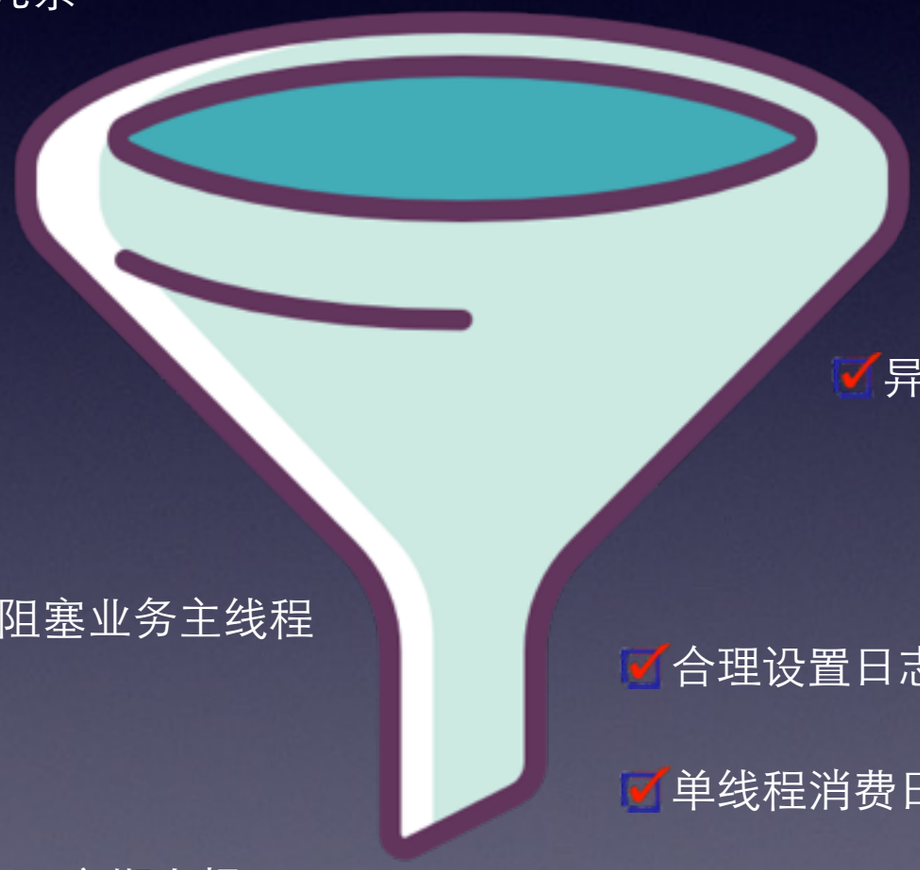
Bottleneck

☑合理设计日志格式，尽量降低日志冗余

☑日志采集粒度可降级

☑采用日志队列模式，避免阻塞业务主线程

☑监控日志丢弃率，定期上报



☑峰值流量下，设置合理的采样率

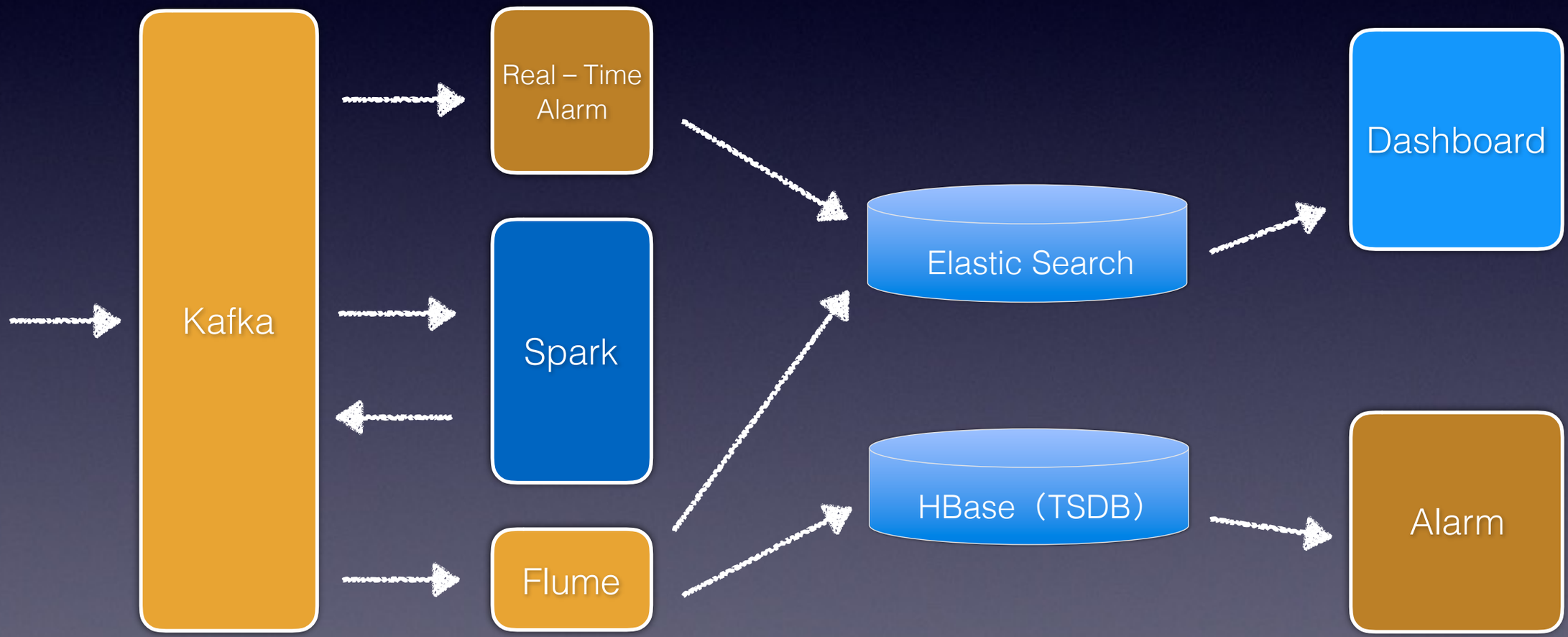
☑异常日志收敛

☑合理设置日志队列大小，允许日志丢弃

☑单线程消费日志队列，批量获取日志

☑网络传输采用二进制序列化格式

Pipeline


Real – Time
Alarm
Kafka
Spark
Flume
Elastic Search
HBase（TSDB）
Dashboard
Alarm

海量数据

100,000,000

峰值日志量／分钟

5T+

日均日志存储量

1T+

日均索引量

500+

接入业务域

20,000+

主机

1,5000,000,000

日志量／每天

海纳百川

流式计算

- ✓ 按需设计Kafka message header
- ✓ 二阶段计算模型
- ✓ 记录数据回放Checkpoint，支持容错
- ✓ 数据限流
- ✓ 内核参数／JVM参数优化

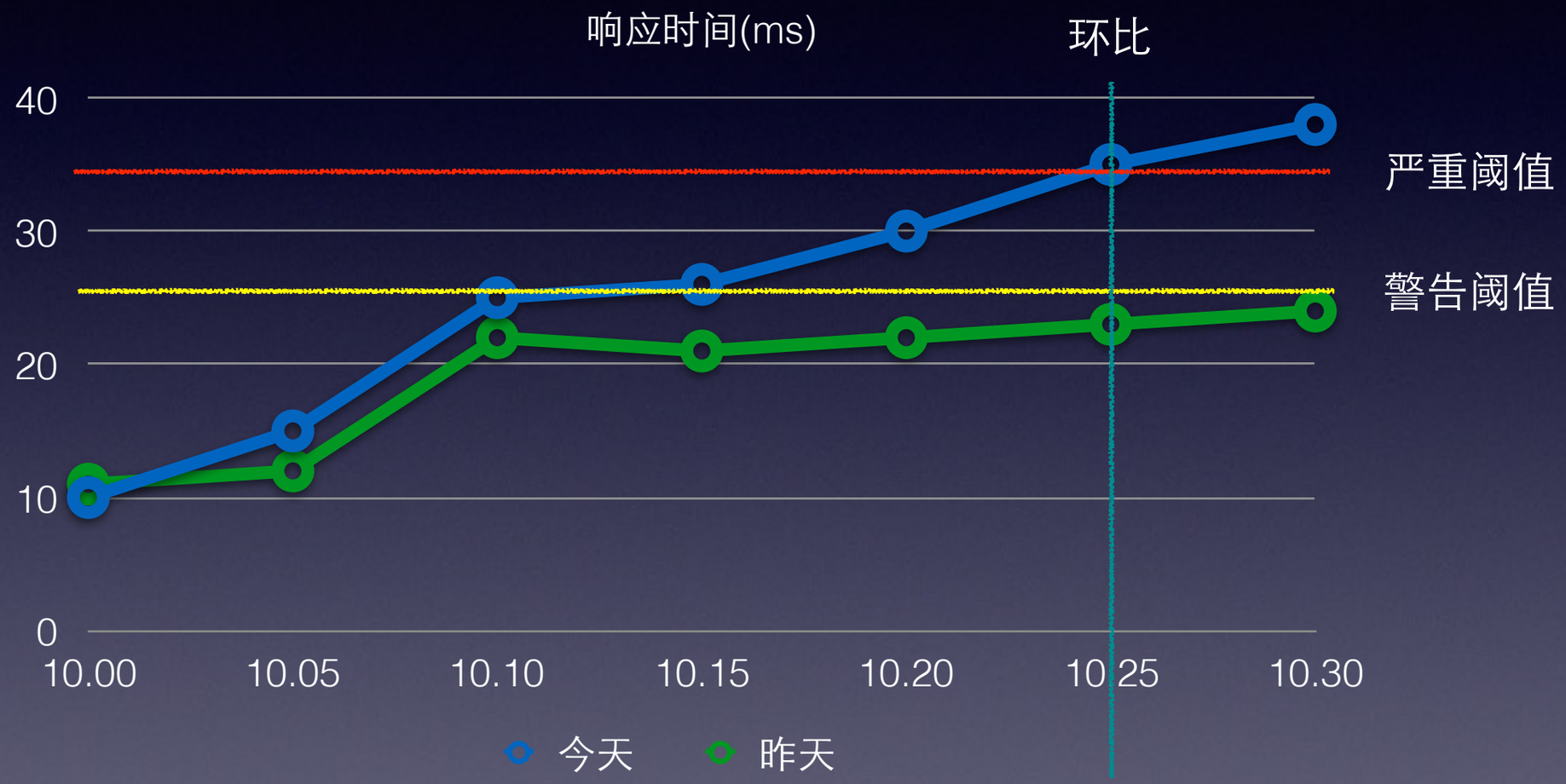
海量存储

- ✓ 基于OpenTSDB，保存全量指标数据
- ✓ 控制Cardinality大小
- ✓ Shift to metric
- ✓ 指标数据加盐存储
- ✓ ES维护二级索引，HBase存储裸日志

高速查询

- ✓ 设置合理的查询时间范围
- ✓ 先查ES，再在HBase中一键定位调用链

告警引擎



- 定义告警模型
- 多维度（服务／业务域／主机）
- 梯度告警

三分建设，七分运营

监控系统的SLA

三分建设，七分运营

监控系统的SLA

自监控，治理

客户端监控

系统监控

组件监控

Q&A