调研方向

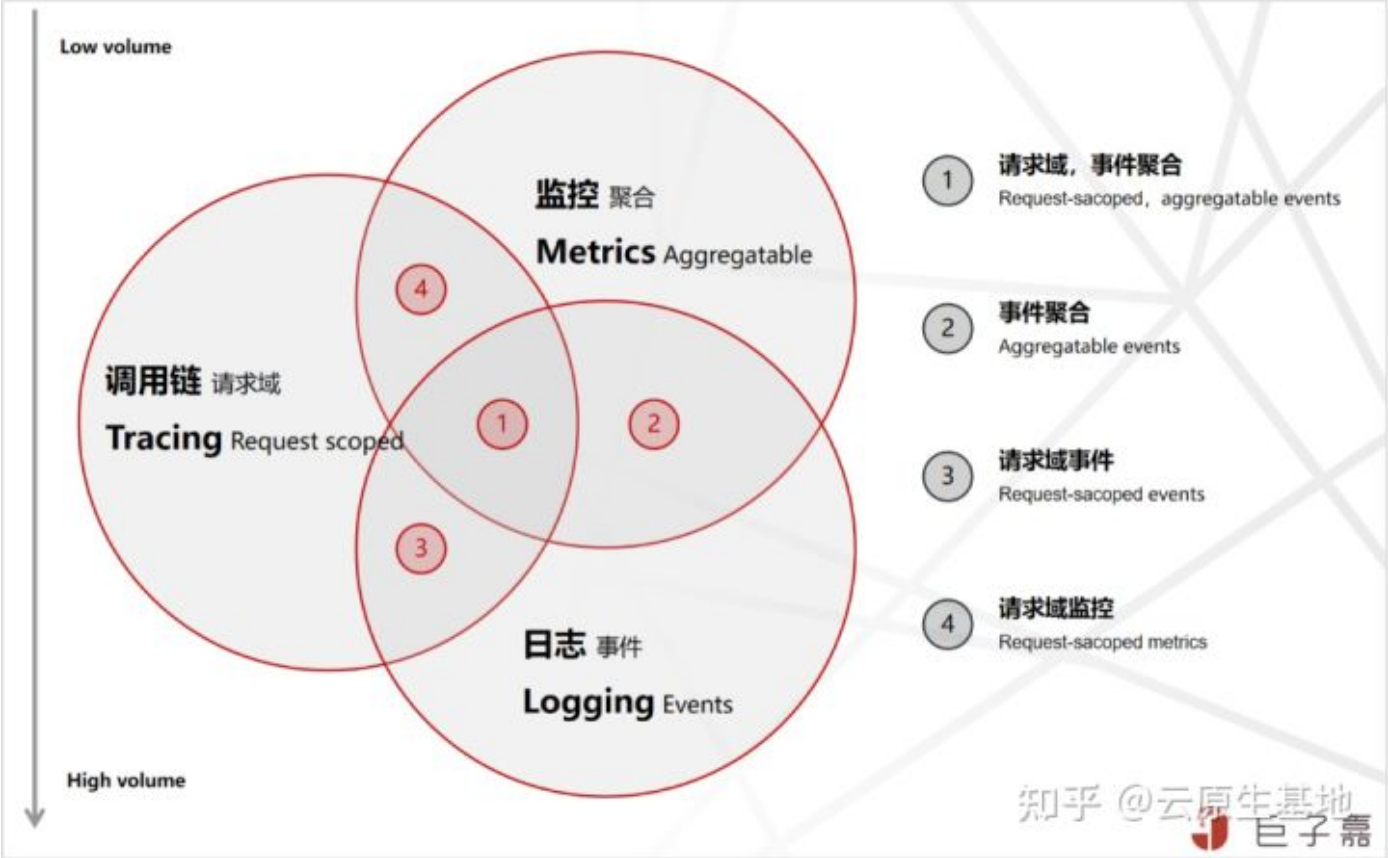
1. 大厂/创业解决方案是如何与云原生领域结合得更紧密的？
2. 微服务链路等等有更多的可观测性？
3. 创新性？

云原生时代的可观测性

时代背景

进入云原生时代，**应用的构建部署**与**运行时基础设施**都发生翻天覆地的变化，技术架构微服务化、运行时环境容器化、业务系统依赖关系复杂化，运行实例生命周期短，规模

大； **服务自动注册发现** ，监控也随着实时动态调整，传统预先配置再监控的方式已经无法满足云原生的场景。可观测性与监控是有本质区别的，监控更多的偏向 **自动化工具** ，可以替代人自动监控系统异常；而可观测性不仅包含传统监控的能力，更多的是 **面向业务**，强调将 **业务全过程透明化的理念**。



观测平台的四个层次

- 采集器：日志采集器、调用链跟踪器、指标采集器，如node-exporter；
- 观测工具：可以集成单个或多个采集器，进行数据可视化的工具，如Prometheus；
- 解决方案：对观测工具的拓展，下至系统信息采集，上至数据可视化、数据分析、安全监测、AIOPS；
- 观测平台：综合性功能的集大成者，云厂商自研平台；

可观测性的三个数据源

- Logging日志：**结构化与非结构化的混合数据** ，应用运行过程会持续输出日志数据，这些日志数据是业务系统运行状态的各种事件及业务处理逻辑时输出的，如果业务日志比较完善，基本上可以还原业务流程处理的全过程。日志服务主要就是收集并长时间保存日志，如果出现事故或者其他审计需求，就可以借助日志服务随时间查阅日志信息。日志服务由来已久，主要难点是解决大规模日志数据**采集，解析，存储，实时检索**等问题，还有就是寻找低侵入的方式（Agent，Sidecar）来标准化输出日志，提高日志数据的质量，降低业务系统日志优化成本；目前业内常见

的**ELK Stack**是日志服务的大成者。ELK Stack在云原生场景主要是解决日志采集适配，监听容器运行变化，灵活稳定有效地采集容器化应用的运行时日志即可。

- Tracing调用链：**结构化与非结构化的混合数据**，尽最大可能串联单个事务内全过程的日志数据，通过对请求打标、透传、串联，最终可以还原出 **一次完整的请求**，帮助工程师分析出请求中的各种异常点；调用链基本上是基于谷歌Dapper理论来实现的，但是对业务系统要求比较高，对业务性能是有一定的影响，在单个业务系统内比较容易实现，假如在异构应用间，尤其是串联共有云的服务组件与业务系统，业务系统前中后台的全流程，实现非常困难。
- Metrics监控：**结构化数据**， 描述的是在给定时间点对系统的度量，基本上分为四大类型，分别是 **Sum**、**Count**、**Last value**、**Histograms**，监控主要是底层基础资源运行状态的数据，通过多维度聚合、分析和可视化展示，帮助快速理解系统资源的运行状态。

可观测性的四个维度

- 分布式链路追踪技术：可观测的基石
- APM：Application Performance Monitoring，应用性能监控
- NPM：Network Performance Monitoring，网络性能监控
- RUM：Real User Monitoring，真实用户监控

业界产品系列

可观测性有三个维度，Logging、Tracing、Metrics，每个维度都有成熟的独立产品。

Logging：Fluentd（CNCF）、**ELK**、**EFK**等

Tracing：**Jaeger**（CNCF）等

Metrics：**Prometheus**（CNCF）等

业界解决方案与**OpenTelemetry**

基于上述三个维度进行排列组合，搭建云原生监控解决方案级别的系统。

困难：不同产品间的协议标准不同，排列组合存在兼容性问题。

CNCF基于此出发点，推出OpenTelemetry，旨在提供可观测性领域的标准，解决可观测性的数据模型、采集、处理、导出等的标准化问题，提供与第三方厂商无关的服务。

总结

Eunomia目前应当属于是采集器级别，目标是与其他开源组件完成适配工作，提供一套完整的云原生监控。

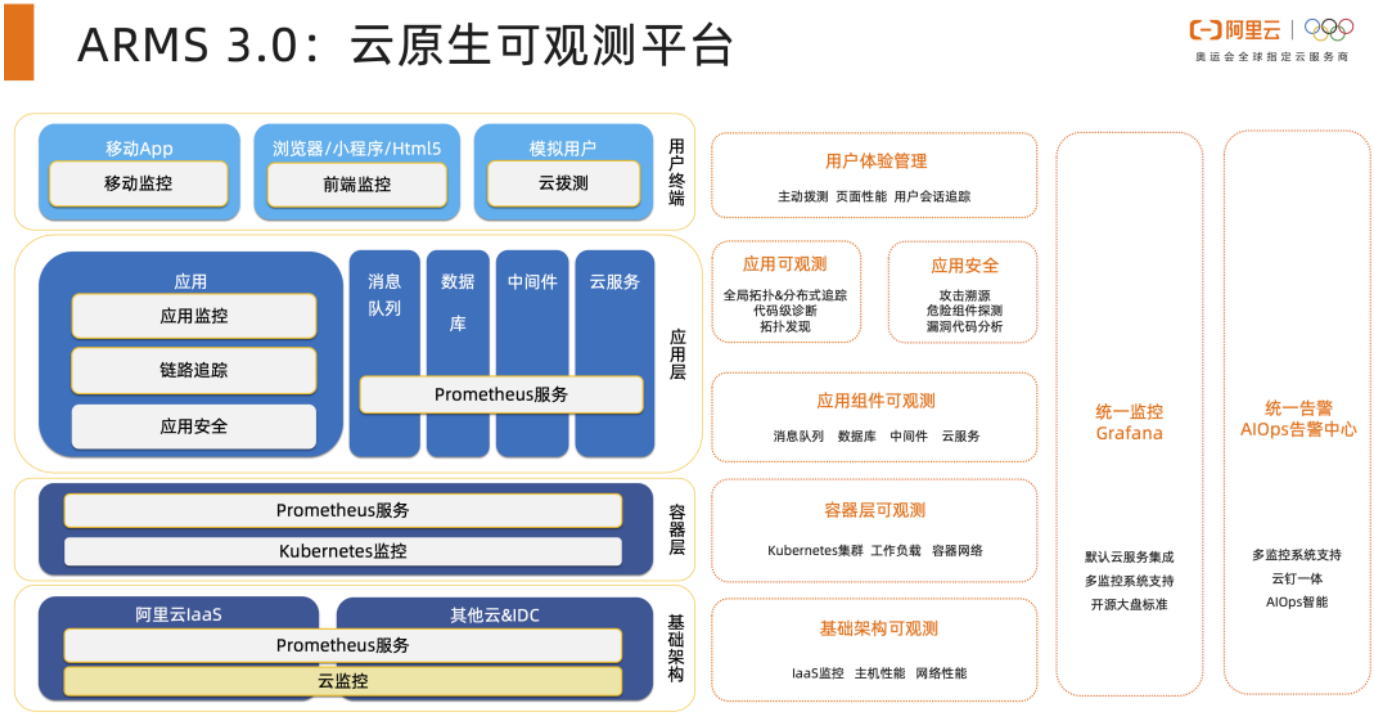
从数据源来说，缺少日志能力、调用链跟踪能力；

从功能来说，安全引擎不完善；

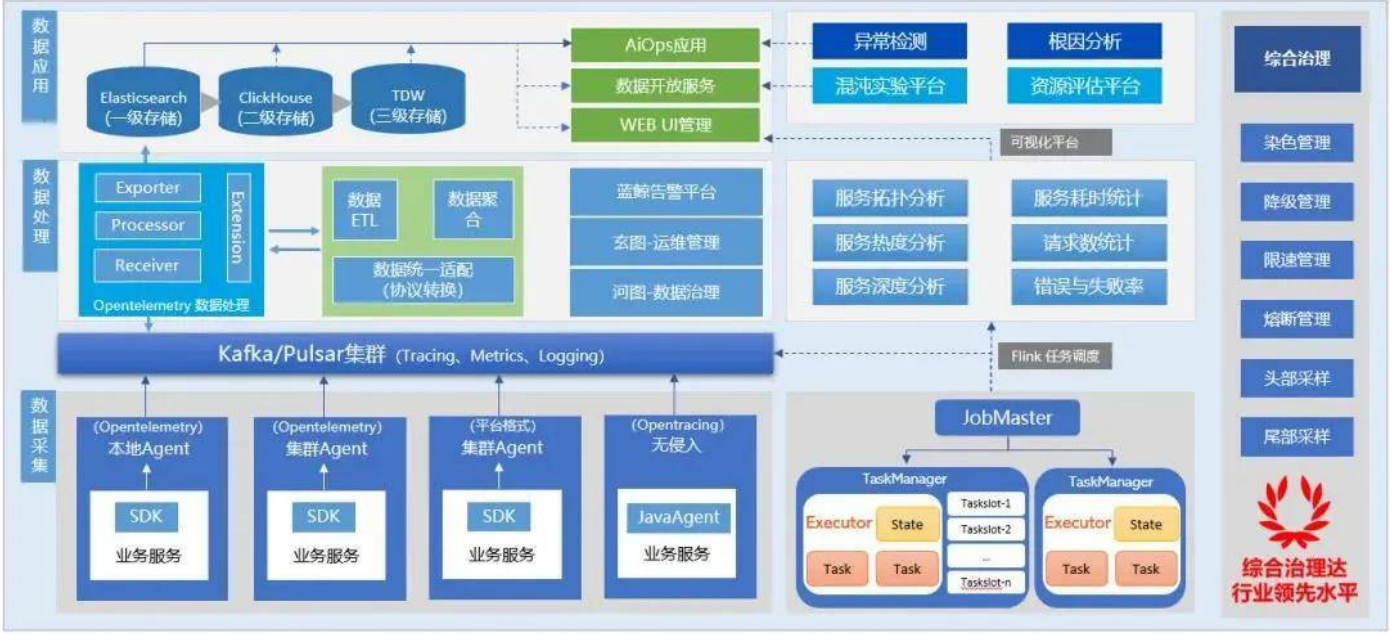
从整体架构来说，数据存储方案不完备：本地持久化？远端存储？；

业界案例

阿里巴巴：**ARMS** （应用实时监控服务）



腾讯云

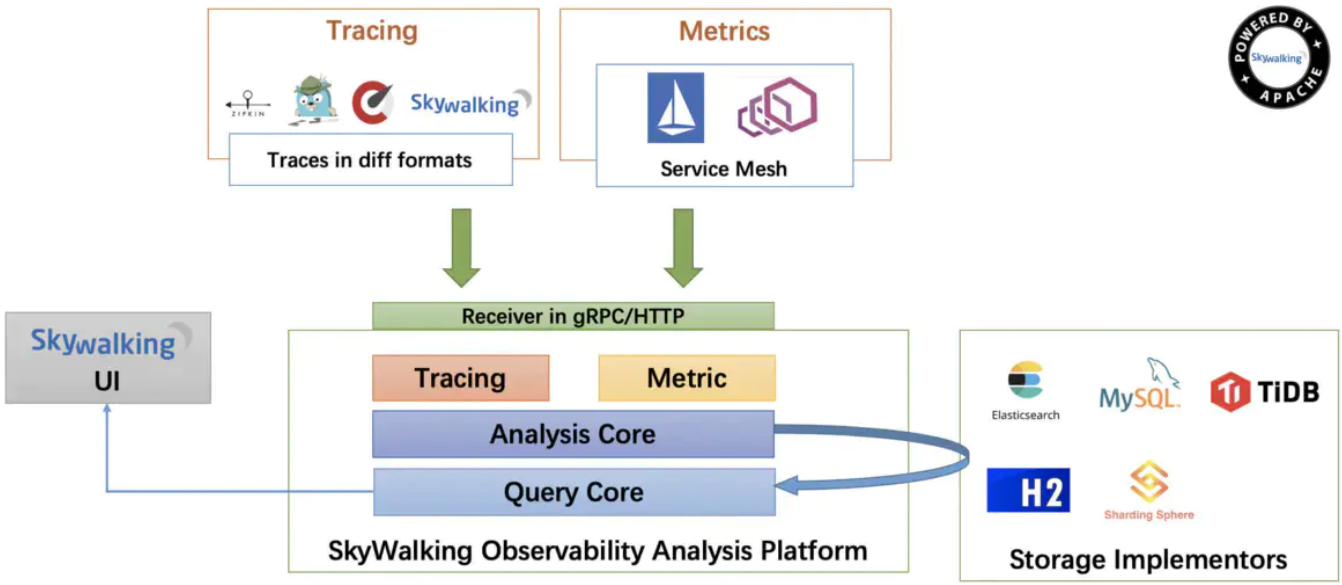


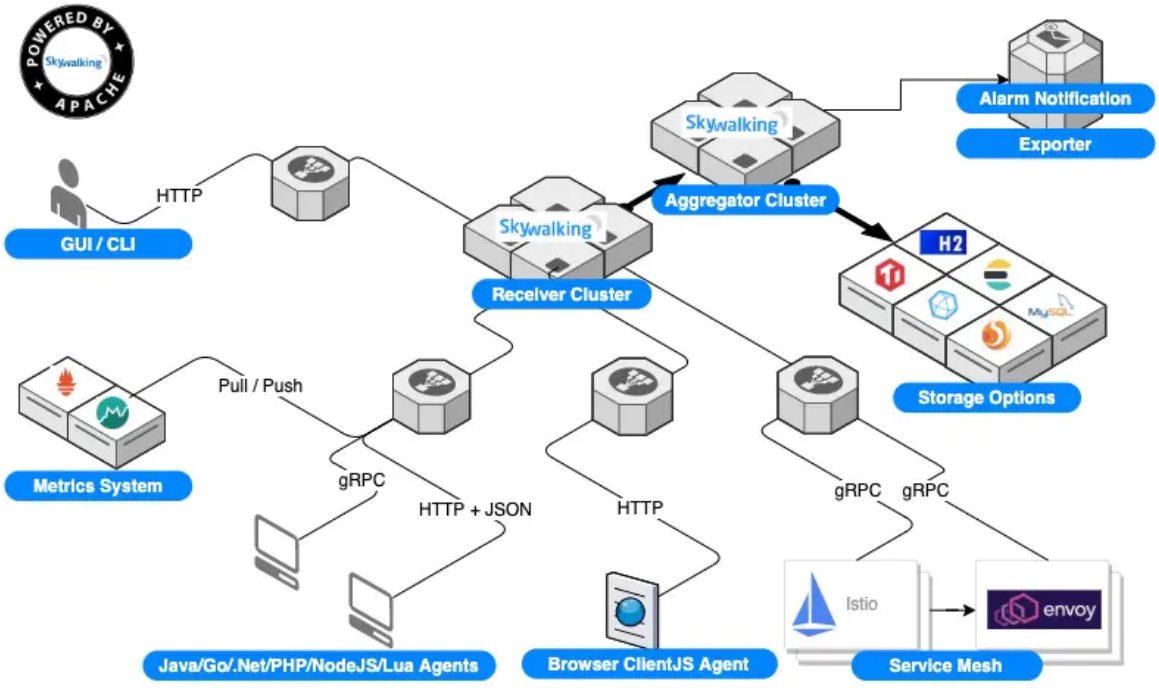
百度云

未知

Skywalking

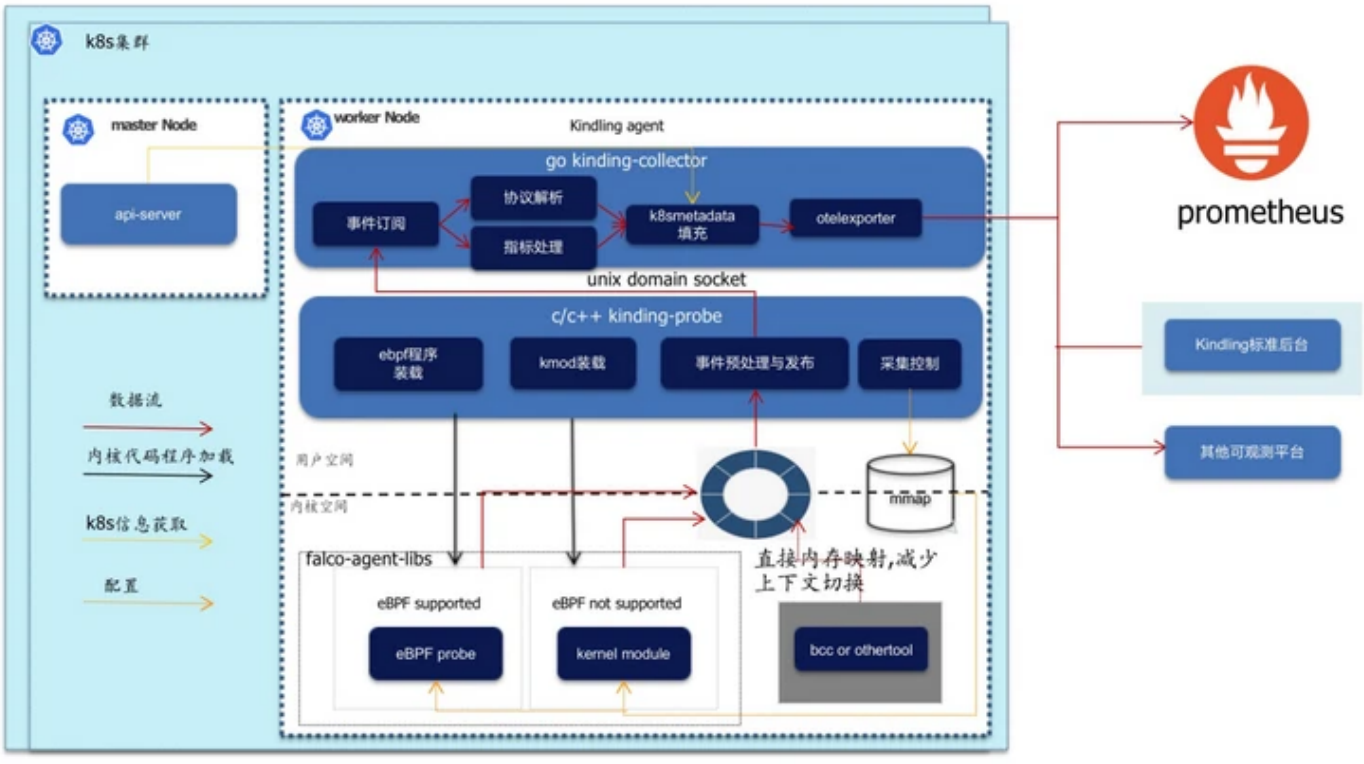
SkyWalking 逻辑上分为四部分: 探针, 平台后端, 存储和用户界面。





DataDog：USM

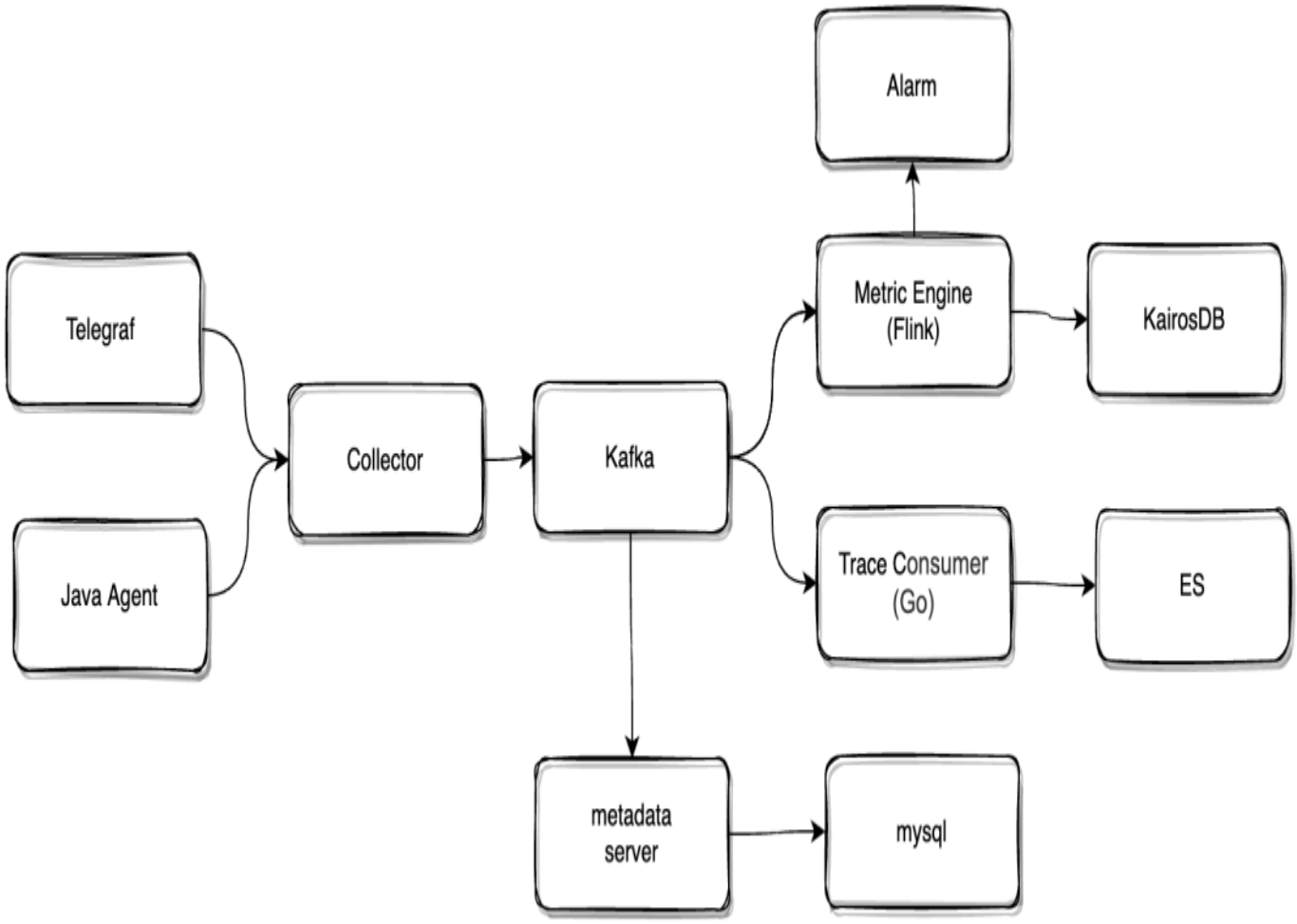
Kindling



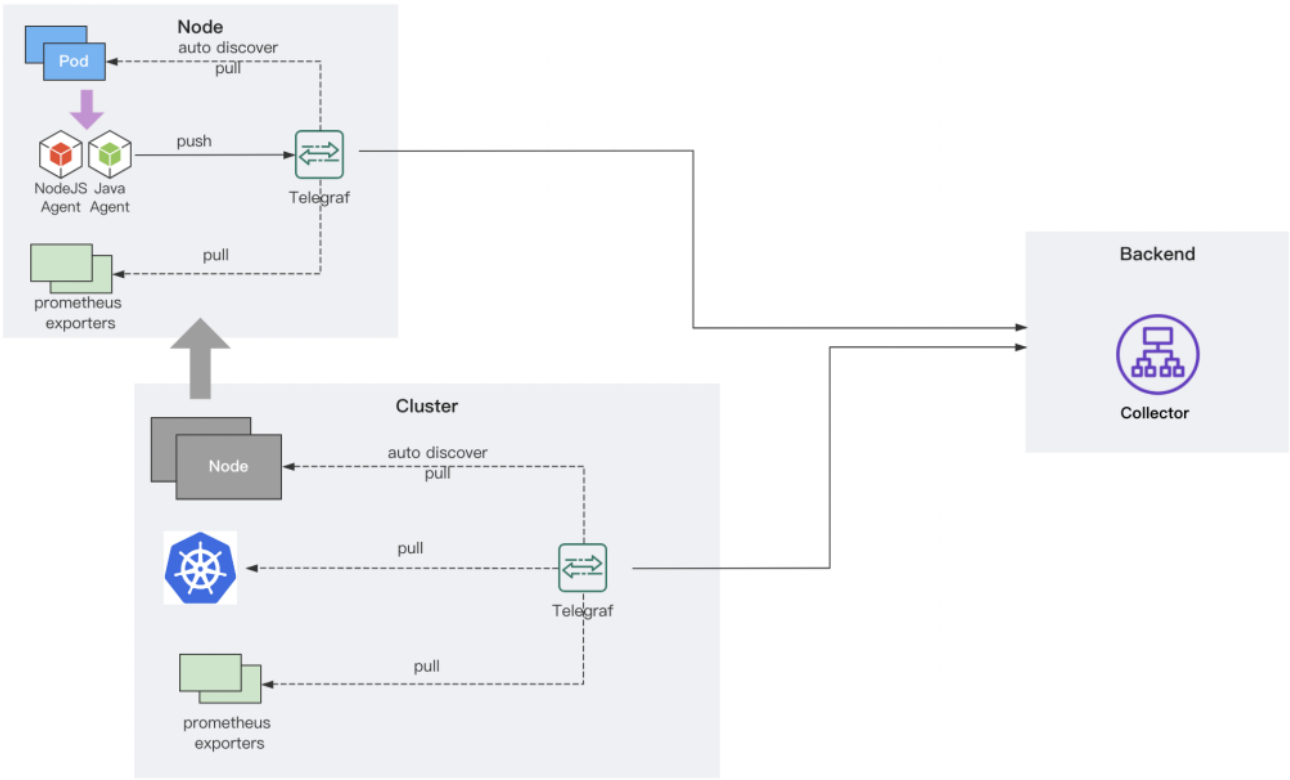
端点科技的**OpenTelemetry**实践

端点科技实践OpenTelemetry

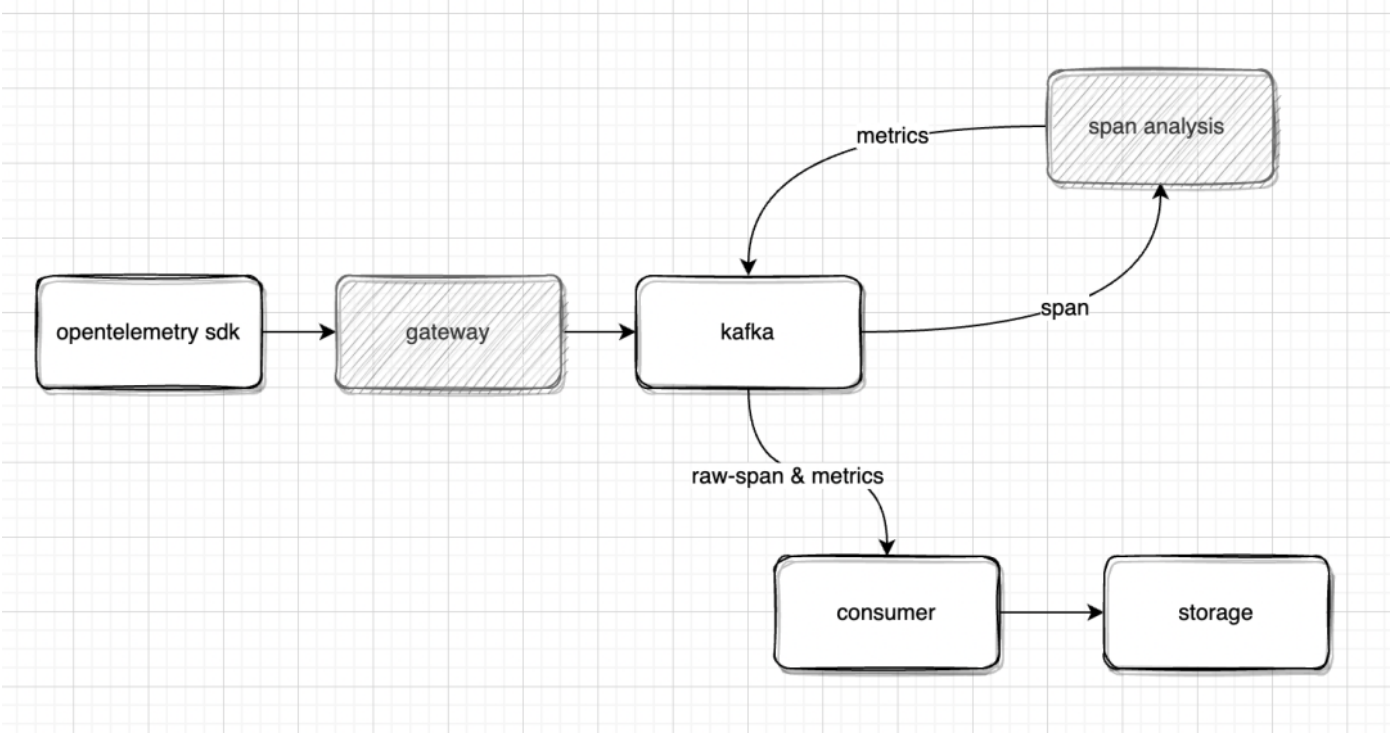
系统架构


Alarm
Telegraf
Metric Engine
(Flink)
KairosDB
Collector
Kafka
Java Agent
Trace Consumer
(Go)
ES
metadata
server
mysql

集群采集端


Node
auto discover
pull
Pod
push
NodeJS Agent
Java Agent
Telegraf
pull
prometheus exporters
Backend
Collector
Cluster
Node
auto discover
pull
pull
Telegraf
pull
prometheus exporters

接入OpenTelemetry



平台对比

厂商	观测工具	层次	架构设计	创新特性	国内	国际
阿里云	ARMS3.0	平台	ARMS3.0架构	1.服务启动热点（热点优化） 2.服务依赖的分析 3.服务应用调用链（微服务观测） 4.结合专家经验的智能运维（AIOPS）	主流	\

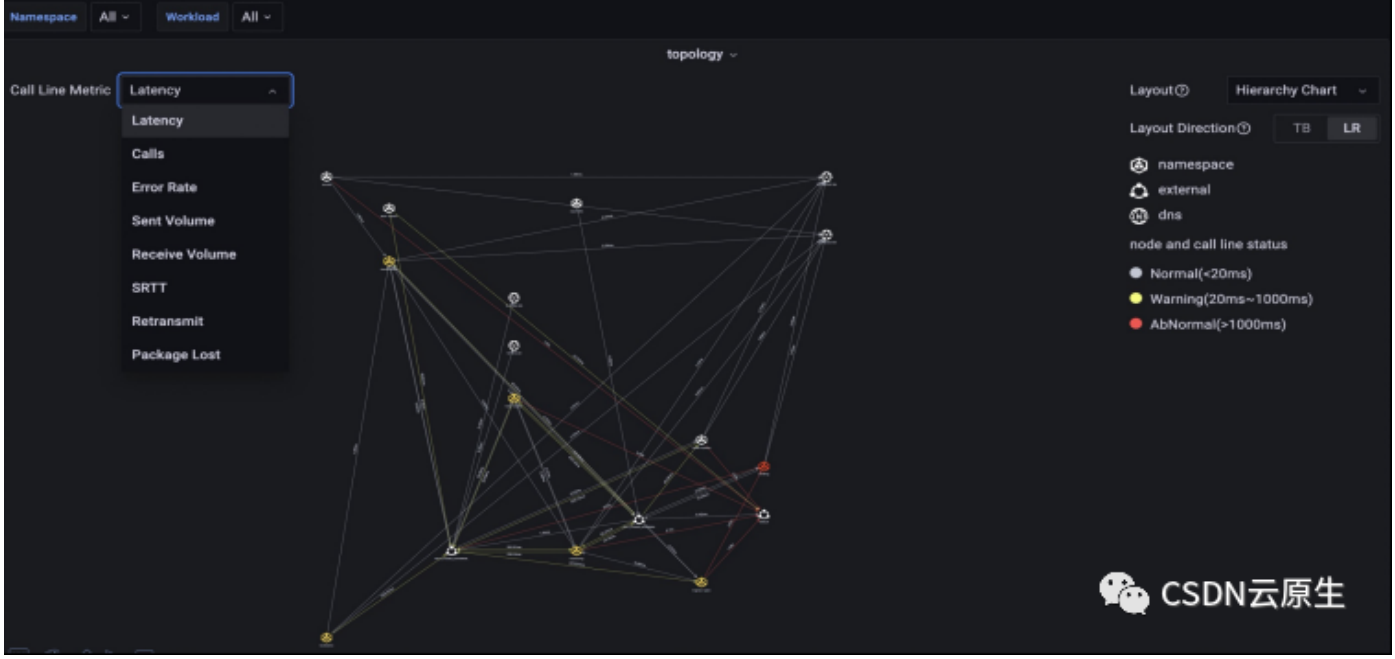
厂商	观测工具	层次	架构设计	创新特性	国内	国际
腾讯云	玄图	平台	腾讯玄图-可观测平台详解	1. OneSDK 统一上报：遵循 OpenTelemetry 协议规范，集成指标、追踪、日志能力-OneSDK，解决多节点上报时间误差至微妙级。 2. 灵活的数据治理能力； 3. 丰富的能力扩展支持； 4. 多语言 SDK 支持； 5.服务解耦&分级存储：引入 Kafka/Pulsar 消息中间件做上下游解耦，极大扩展前后台服务能力；	主流	\
DataDog	USM	平台	\	\	\	主流
百度云	Prometheus	解决方案	未知	1. 解决大规模线上业务的集群化方案 2. 解决实战中数据量过多、对集群性能影响较大等问题	自用	\
Skywalking	Apache Skywaliking	解决方案	Skywalking	1. 分布式追踪 2.性能指标分析 3.服务依赖分析	开源	Apache

厂商	观测工具	层次	架构设计	创新特性	国内	国际
Kindling	Kindling	解决方案	Kindling探针架构	1. 补充前面提到的云原生可观测性盲区（即关于云原生基础设施层，以及应用层面的调用问题） 2. 基础设施层增强指标力度，更细粒度的指标 3. 在应用层面，依赖自研动态**APM探针**的能力以实现分析应用具体问题能力	初创	\
端点科技	自研解决方案	解决方案	架构	实用性强、适合中小公司快速搭建	自用	\

可参考的创新点

（1）pod调用拓扑（kindling）

实现使用eBPF tracepoint实现pod调用拓扑以及如何获取pod http接口的黄金指标。在请求的调用的过程中，系统调用层面会connect一个具体的地址，所以我们可以从系统调用获取到进程和地址的调用关系。在我们的数据预处理端以进程以及IP:port为key关联上container以及K8s相关的metadata作为label。我们目前采用一种分布式达标的方式，在每一个worker节点上都会有一个探针，这个探针会去获取当前集群的metadata数据并把它们保存在内存中，用来实现原始eBPF数据的丰富。这种方式的内存占比也是可以接受，虽然要获取到整个集群K8s相关的数据，但根据测试发现，我们整体label信息的保存也就30~40兆。另外有一个问题如果客户端未装探针，外部请求来到集群内，那么如何得到调用源？这个可以从服务端来分析这个调用，在服务端会通过accept系统调用实现一个连接，通过这个连接，可以获取到具体的调用源IP。下图是Kubernetes集群中拓扑实现的效果。



（2）微服务的可观测性

微服务架构逐渐普及，导致可观测问题变得十分复杂，微服务架构普及后，问题变得更加严峻。一个服务被拆分成数个黑盒的、虚拟的微服务，故障排除彻底成为一种折磨。

（3）分布式的可观测性

- 分布式应用观测系统

文献引用

云原生可观察性-概述篇

2022，我们该如何理解可观测技术_问题_子系统_故障

阿里云云原生微服务可观测性实践

腾讯在云原生可观测领域的探索与实践

腾讯玄图-可观测平台详解

百度可观测系列丨基于 Prometheus 的大规模线上业务监控实践

OpenTelemetry 在云原生 PaaS 中的落地实践