

豆瓣百万级别指标监控实践

朱兆龙 @ Douban

QCon

2016.10.20~22
上海·宝华万豪酒店

全球软件开发大会 2016

[上海站]



购票热线: 010-64738142
会务咨询: qcon@cn.infoq.com
赞助咨询: sponsor@cn.infoq.com
议题提交: speakers@cn.infoq.com
在线咨询 (QQ): 1173834688

团 · 购 · 享 · 受 · 更 · 多 · 优 · 惠

7折 优惠 (截至06月21日)
现在报名, 立省2040元/张

Outline

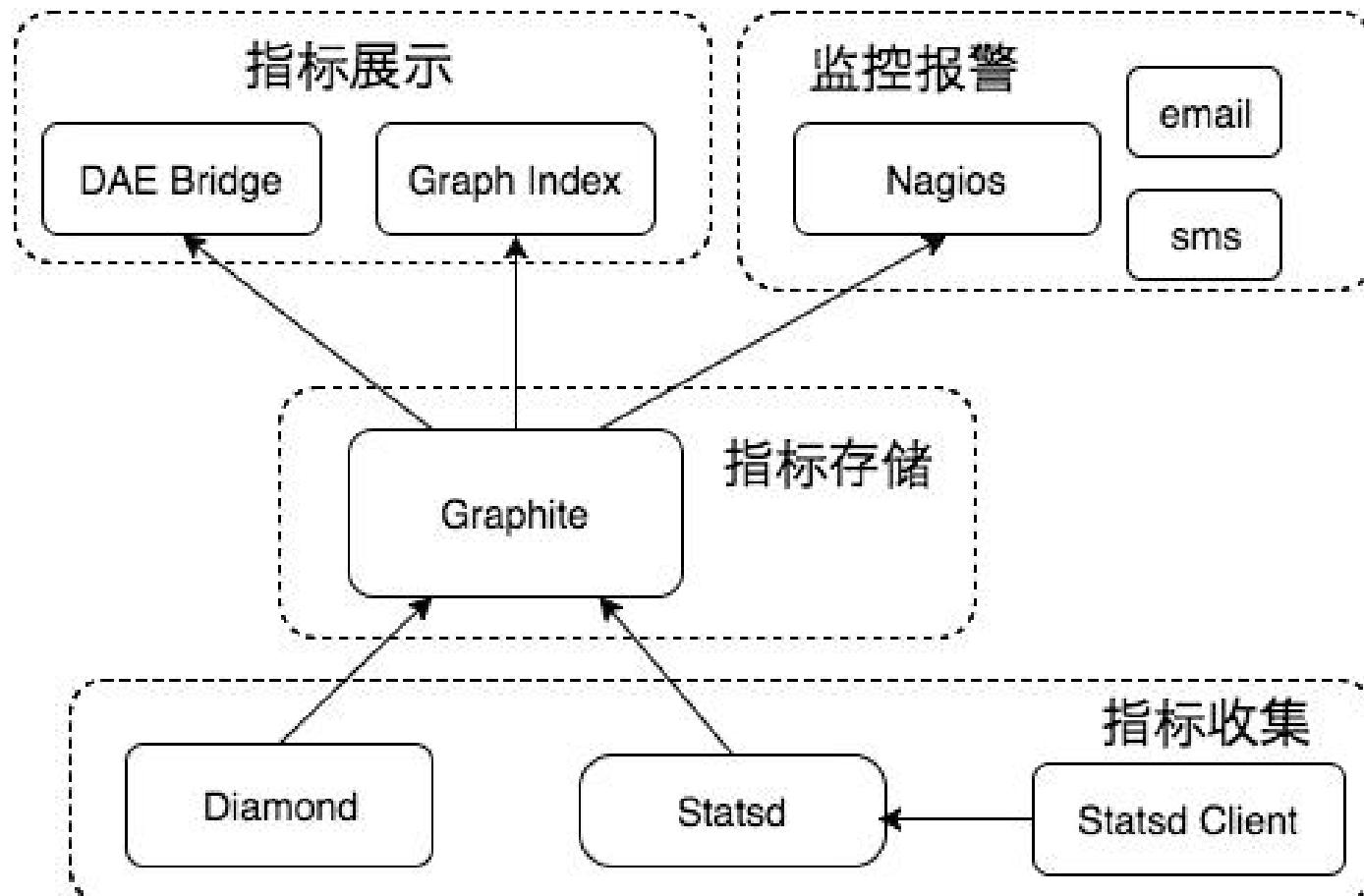


背景介绍

监控系统的重要性

- 实时报警
- 事后分析
- 容量规划

原来的总体架构

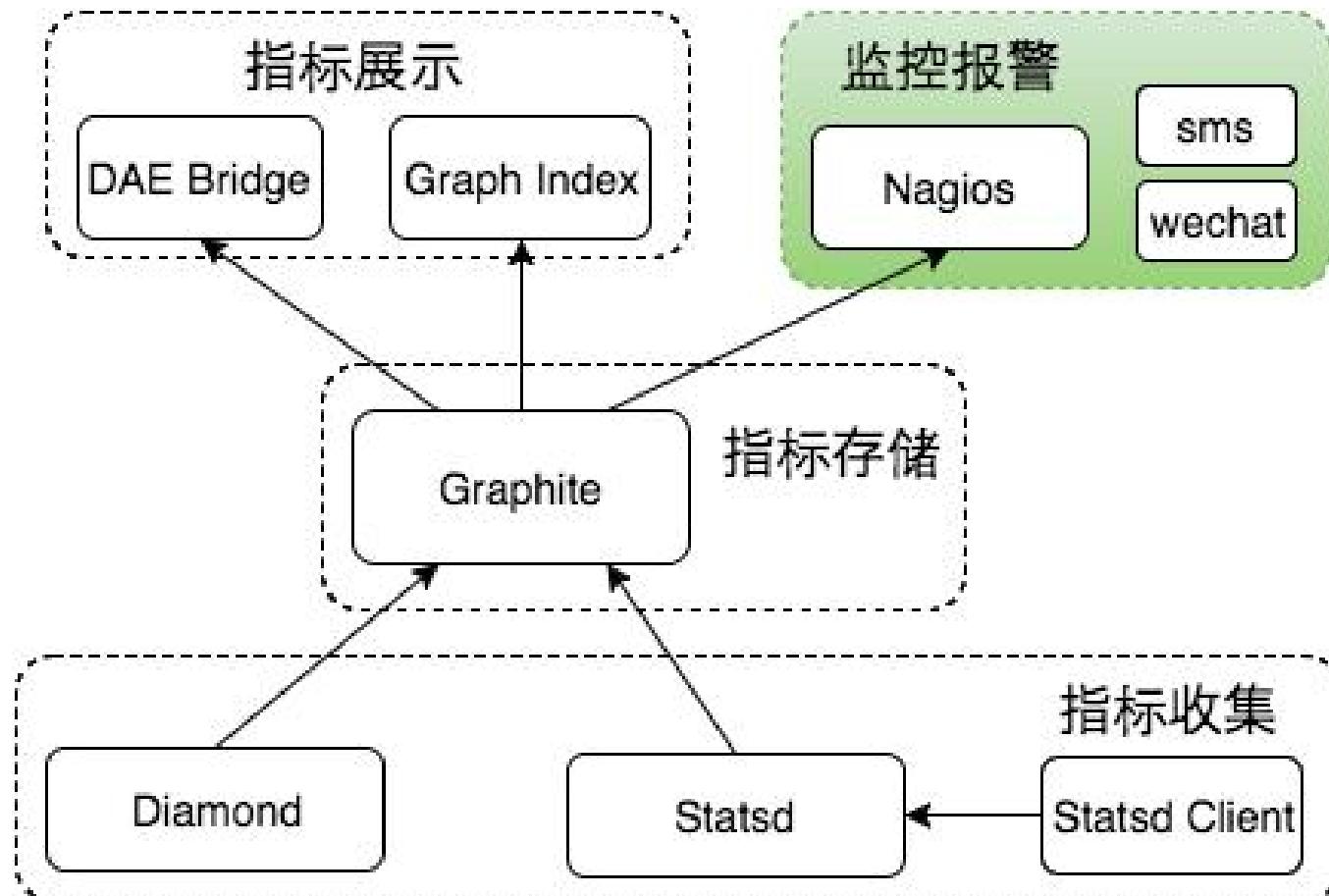


需求

- 实时、可靠的报警系统
- 大量、高精度指标存储系统
- 灵活、方便的查询方式

监控报警

监控报警

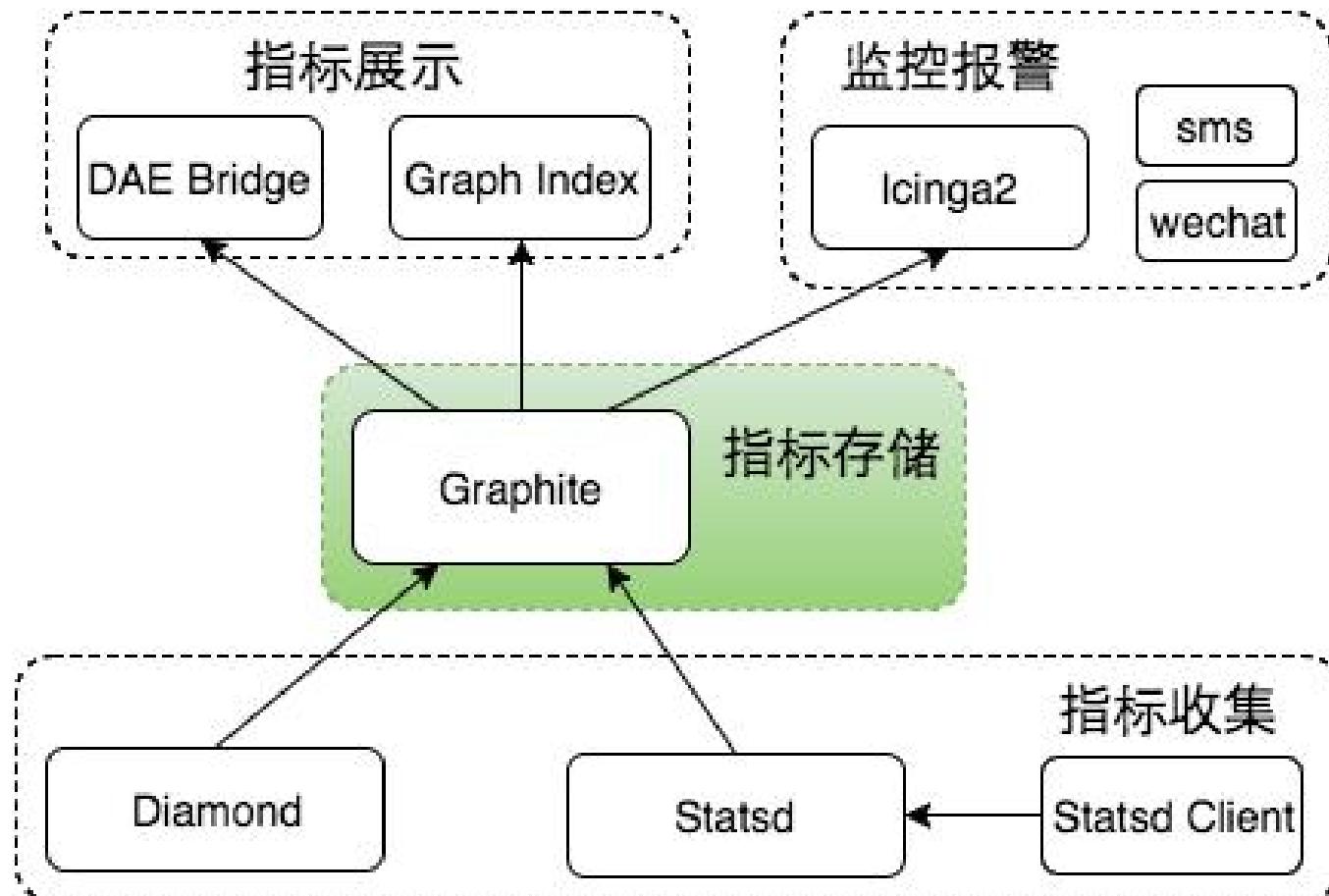


Icinga2

- Icinga2
 - 性能、扩展性、可用性
 - 语法简洁、Icinga2 API、Icingaweb2
- 规模
 - 2 个 master
 - 2 个 checker
 - 1 个 Icingaweb2
 - 1.5 万个监控项
 - 周期 1、5、10 min

指标存储

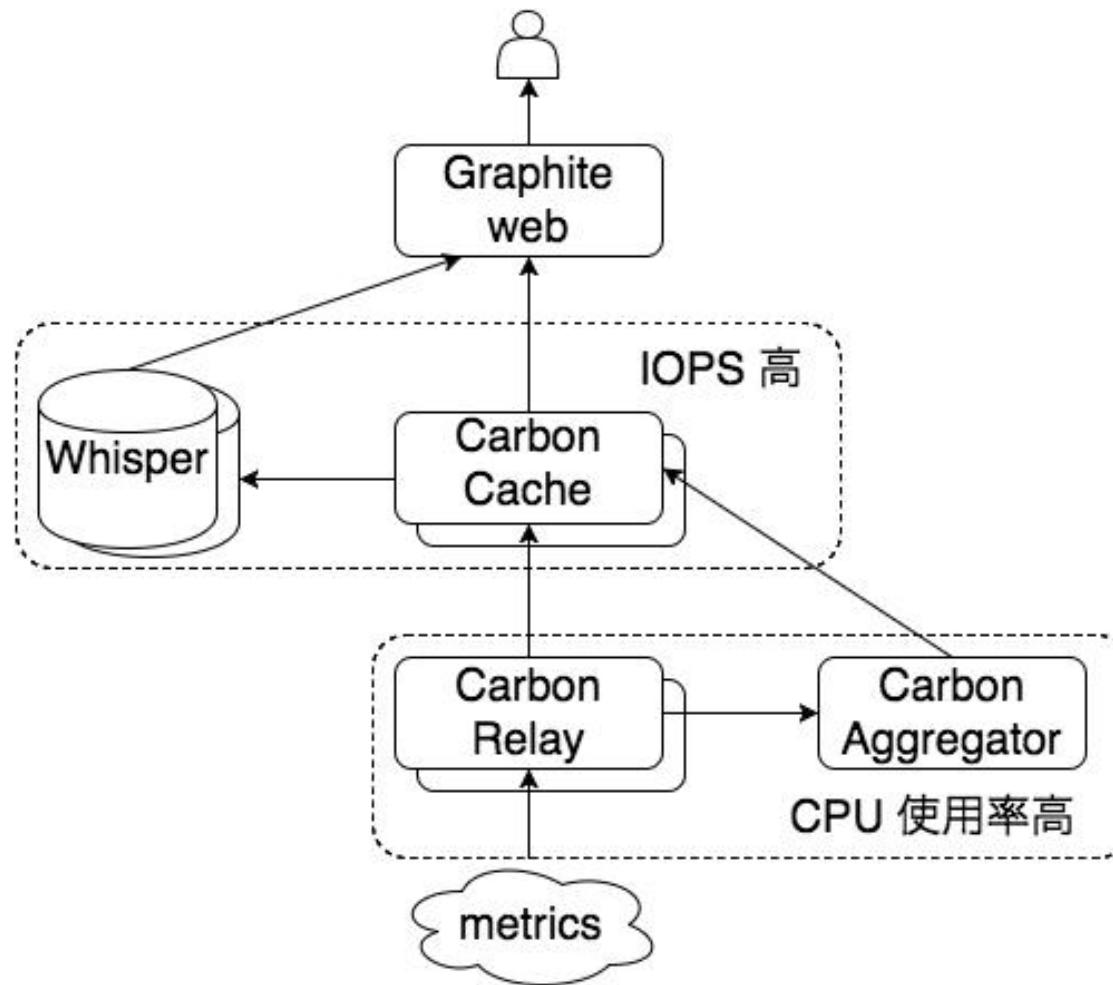
指标存储



存储需求

- 指标数量 2~3 million
- 每个指标的更新周期 1s ~ 10s
- 内存使用 < 20 G
- 写入 IOPS < 1000

Graphite 性能问题



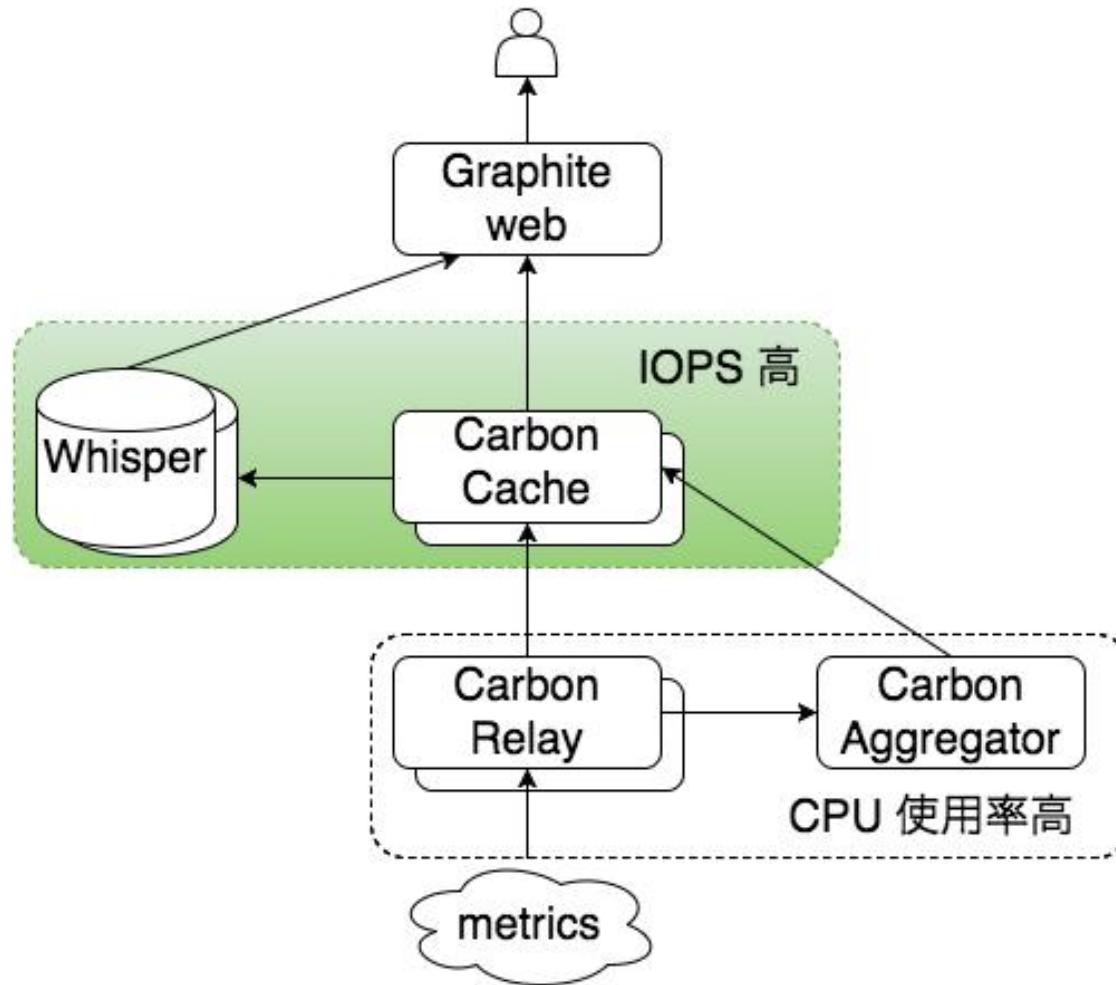
替换 Graphite ?

	优点	缺点
OpenTSDB	<ol style="list-style-type: none">1. Metric + Tags2. 集群方案成熟(HBase)3. 写高效(LSM-Tree)	<ol style="list-style-type: none">1. 查询函数有限2. 依赖 HBase
InfluxDB	<ol style="list-style-type: none">1. Metric + Tags2. 部署简单、无依赖3. 设计目标很好	<ol style="list-style-type: none">1. 集群方案不成熟2. 内存泄露3. 存储引擎在变化

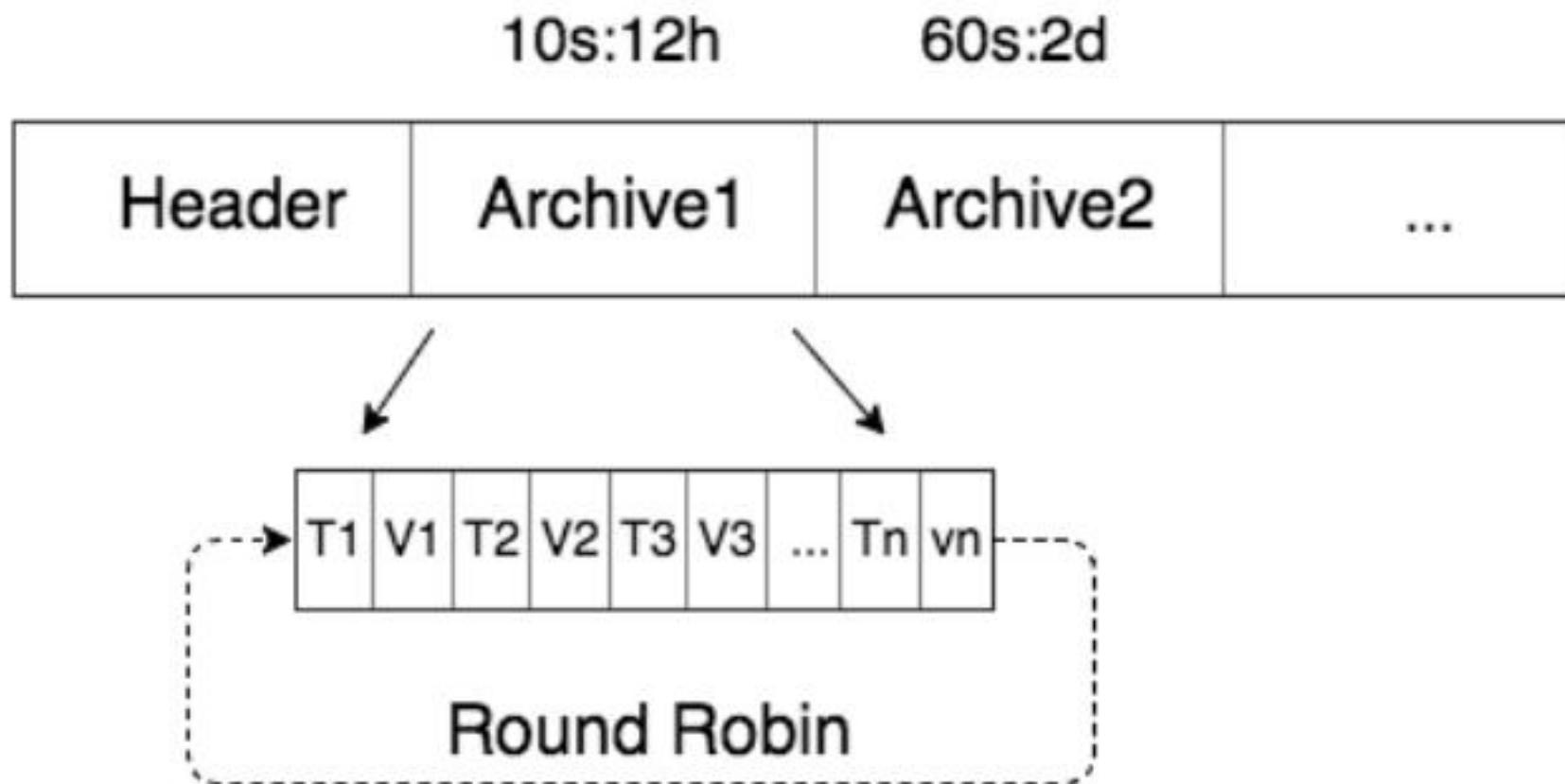
优化 Graphite

- 优点
 - 查询高效
 - 支持自动 Downsample
 - 支持保留策略
 - 实现简单
 - 复用 Graphite Web
- 需要解决的性能问题
 - Whisper 存储引擎 IOPS 高
 - Carbon 组件 CPU 使用率高

Whisper



Whisper 简介



Whisper Read

Read: (from_time, until_time)

① $12h < (\text{now} - \text{from_time}) \leq 2d$

②

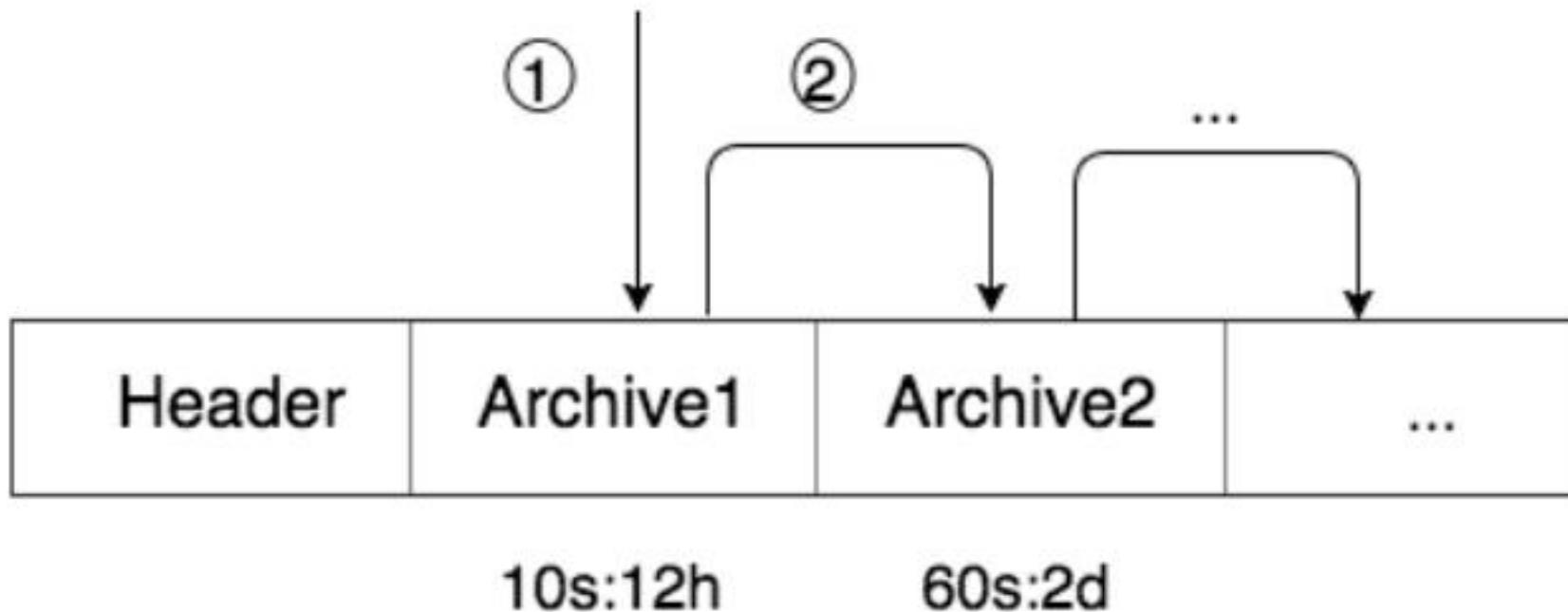
Header	Archive1	Archive2	...
--------	----------	----------	-----

10s:12h

60s:2d

Whisper Write

Write: [datapoint, ...]



Whisper 为什么 IOPS 高？

- 一个指标对应一个文件
- 自动 Downsample 引起的级联更新

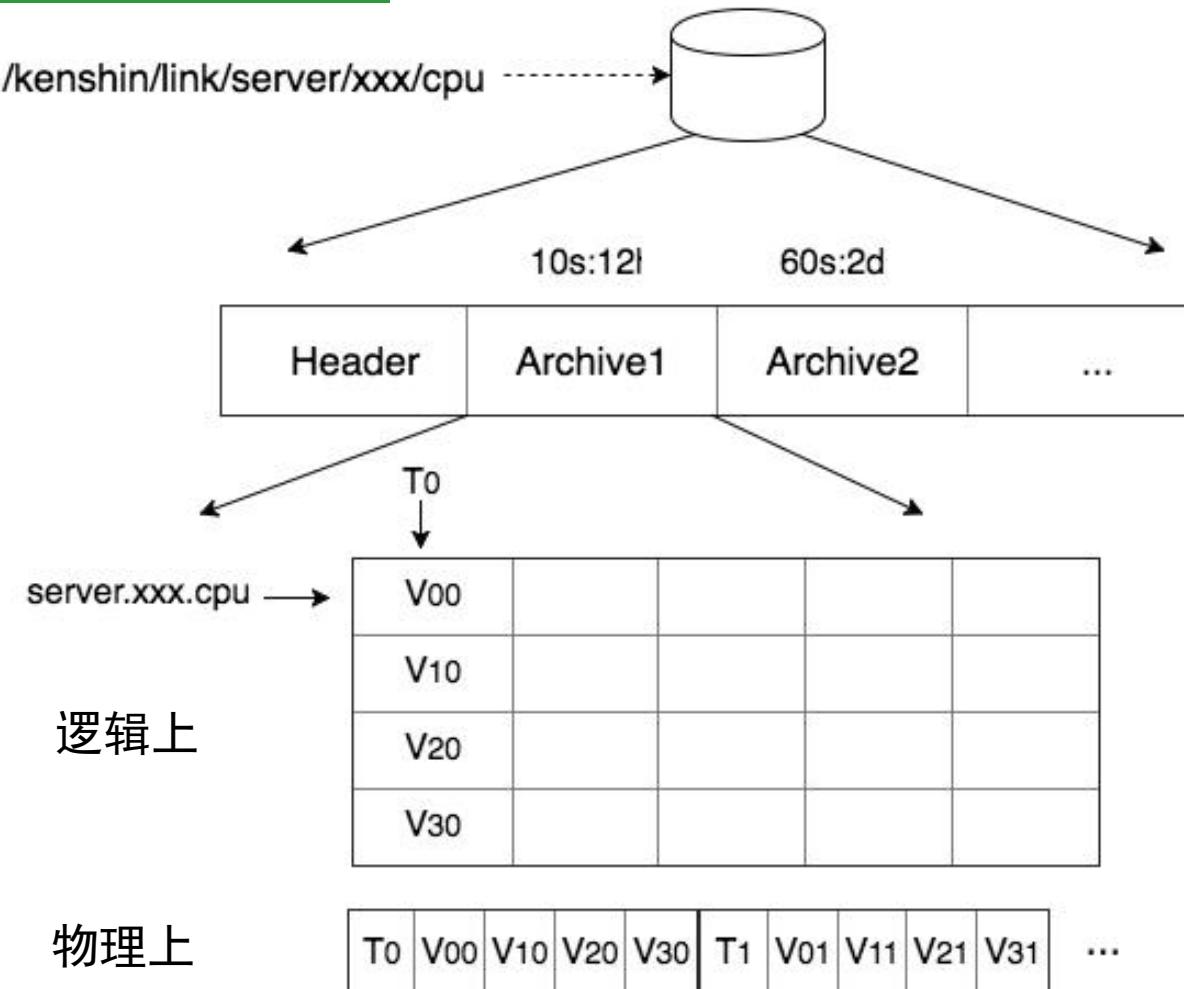
Kenshin 设计

- Kenshin 名称：来源于电影《浪客剑心》
- 基本思路
 - 合并指标文件
 - 减少级联更新
- 目标
 - 减小 IOPS
 - 保持查询速度
 - 兼容 Graphite Web

问题

- 怎么样合并指标文件？

Kenshin 文件结构



问题

- 怎么样合并指标文件？
- 合并后会有什么副作用？

合并后会有什么副作用？

- 副作用：影响查询速度
- 解决办法：cache

问题

- 怎么样合并指标文件？
- 合并后会有什么副作用？
- 什么样的指标合并在一起？

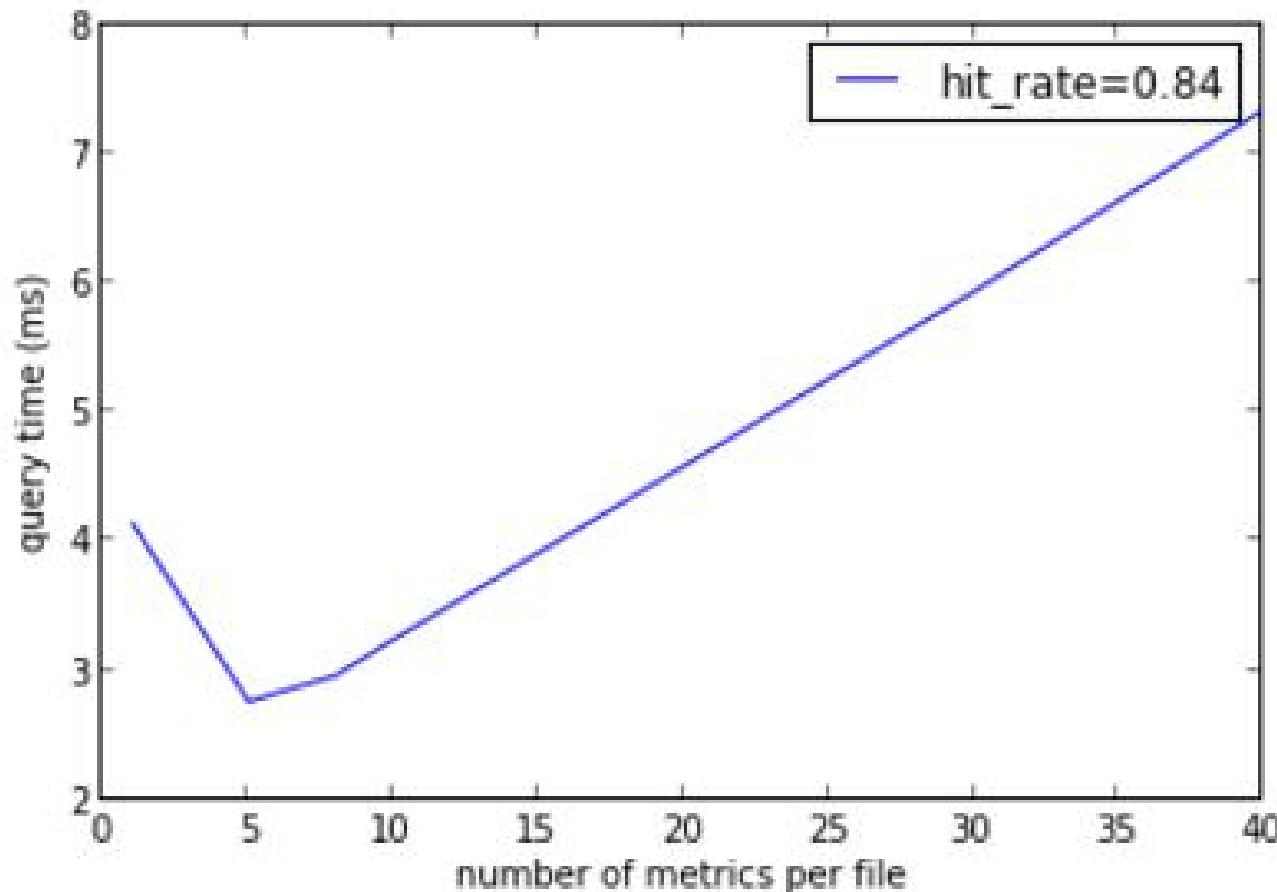
什么样的指标合并在一起？

- 理想：一起被查询的指标应该合并在一起
- 假设：创建时间相邻的指标可能会被一起查询

问题

- 怎么样合并指标文件？
- 合并后会有什么副作用？
- 什么样的指标合并在一起？
- 多少个指标应该合并在一起？

多少个指标合并在一起？



问题

- 怎么样合并指标文件？
- 合并后会有什么副作用？
- 什么样的指标合并在一起？
- 多少个指标应该合并在一起？
- 怎样减少级联更新？

怎样减少级联更新？

- 延迟 Downsample 操作

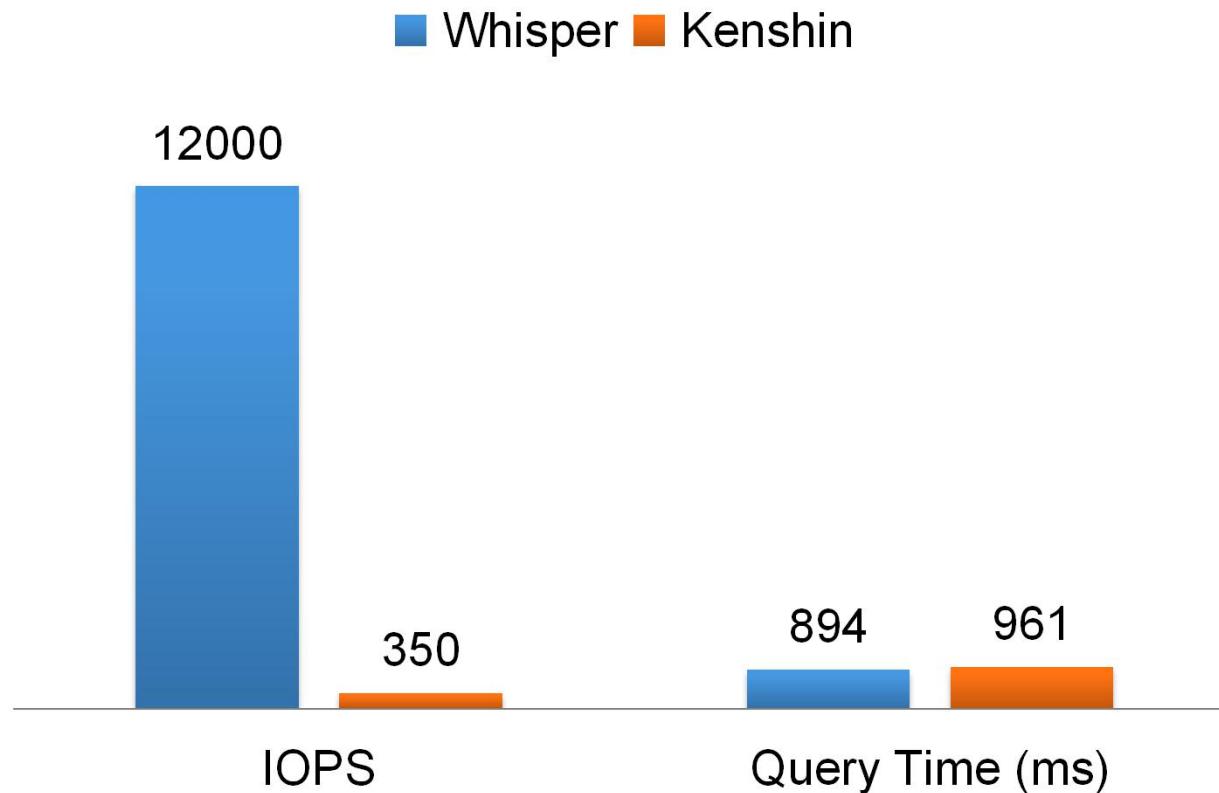
问题

- 怎么样合并指标文件？
- 合并后会有什么副作用？
- 什么样的指标合并在一起？
- 多少个指标应该合并在一起？
- 怎样减少级联更新？
- 减少级联更新会有什么副作用？

减少级联更新的副作用

- 副作用：低精度 Archive 的近期数据延迟到达
- 假设：用户查询长期趋势时容忍近期数据的缺失

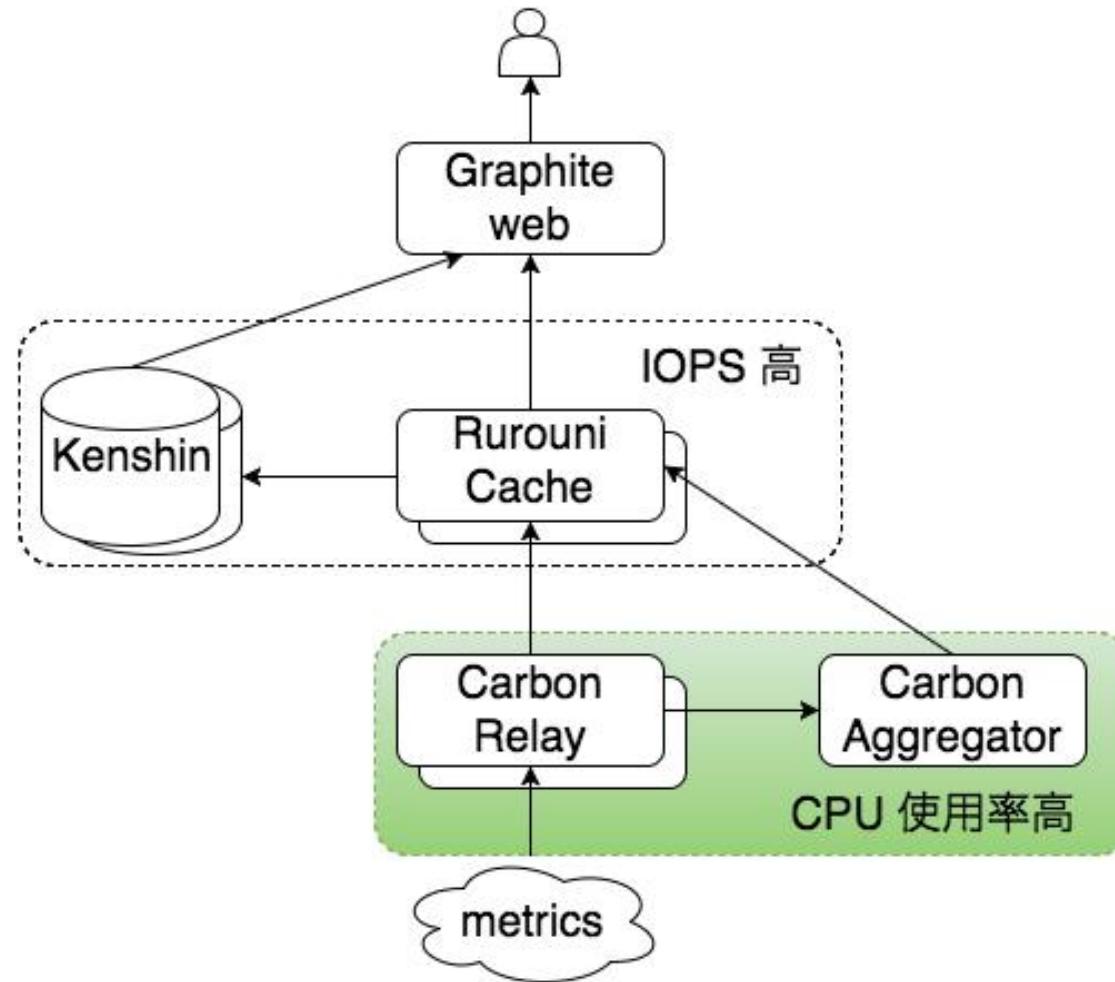
Kenshin 的性能



Kenshin 小结

- IOPS 减少了 40 倍
 - 合并指标: 8
 - 减少级联更新: 5
- 平衡的目标
 - 写性能
 - 读性能
 - 及时性

Carbon Relay



为什么 CPU 使用率高？

- 使用加密性 MD5 Hash 来计算路由信息
- Python 实现

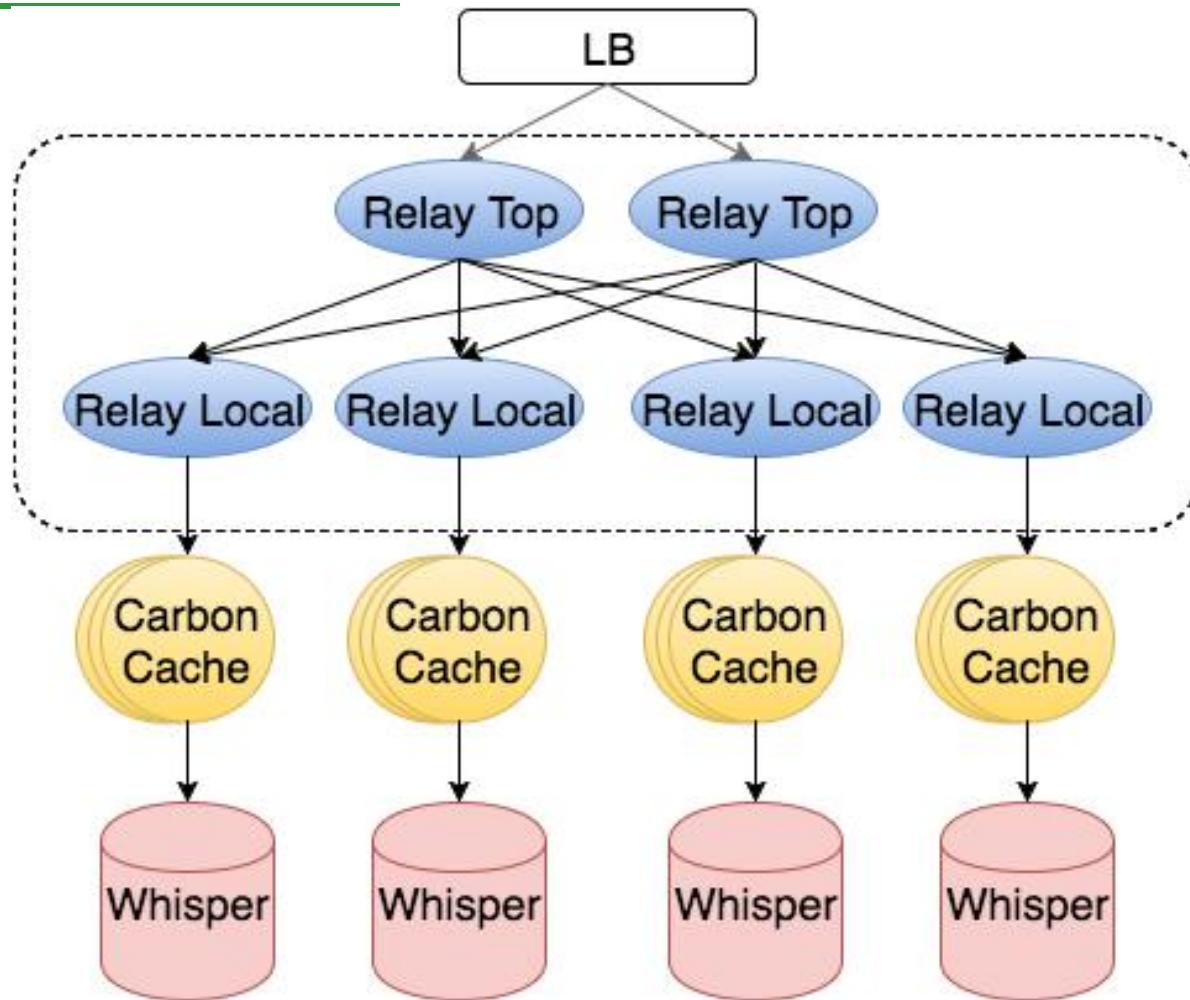
FNV-1a

- MD5 Hash 替换为 FNV-1a Hash
- 效果：CPU 降为原来的 50%

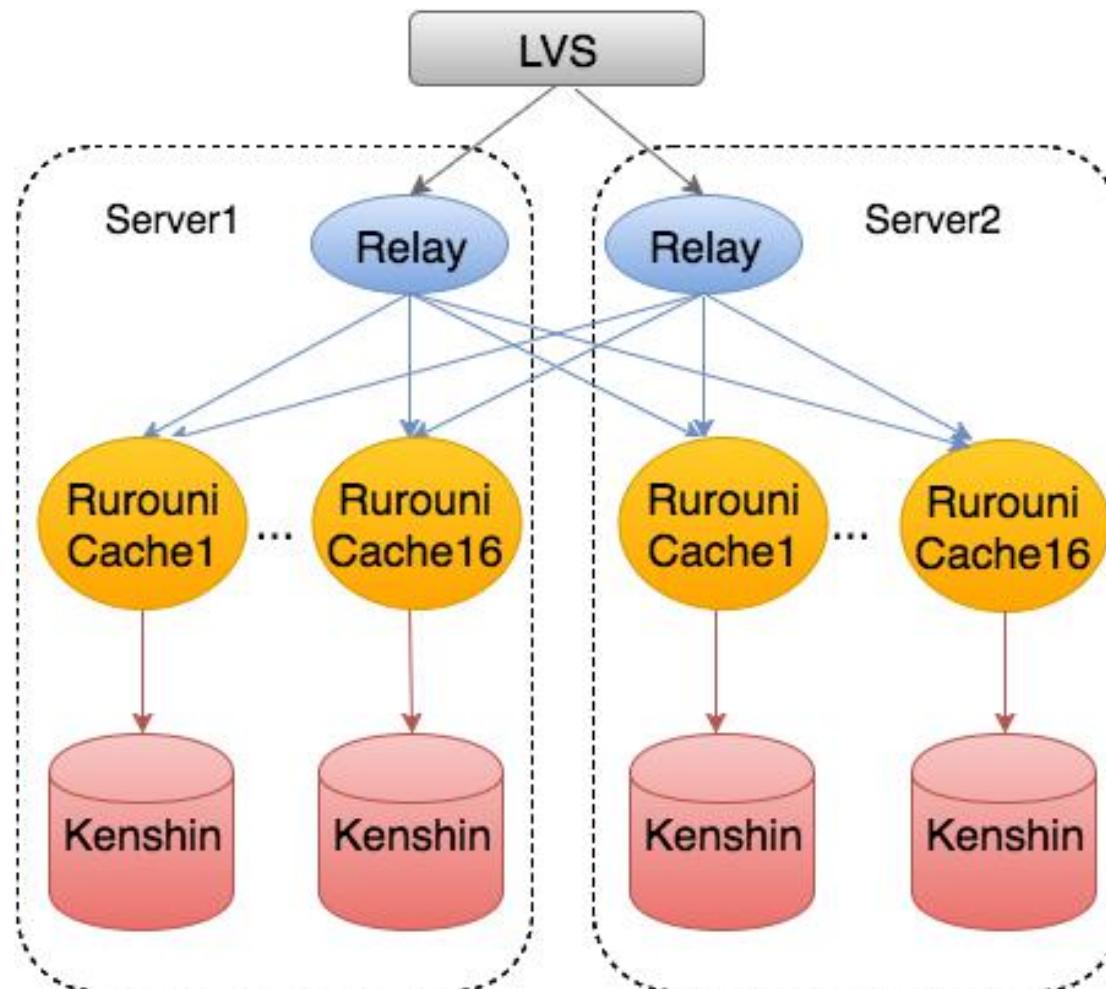
Carbon-C-Relay

- Carbon-Relay 替换为 Carbon-C-Relay
- 效果：CPU 降为原来的 10%

典型的 Graphite 架构



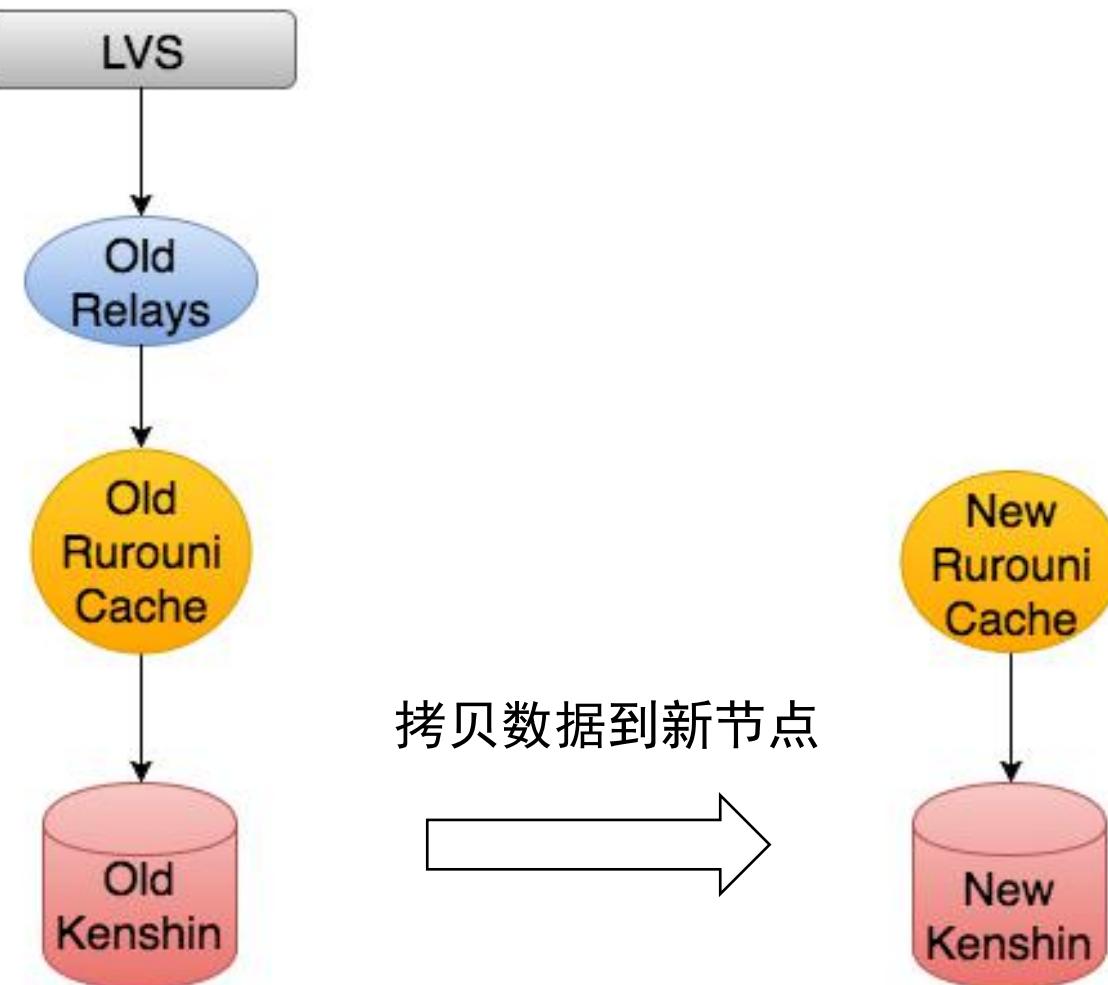
简化后的架构



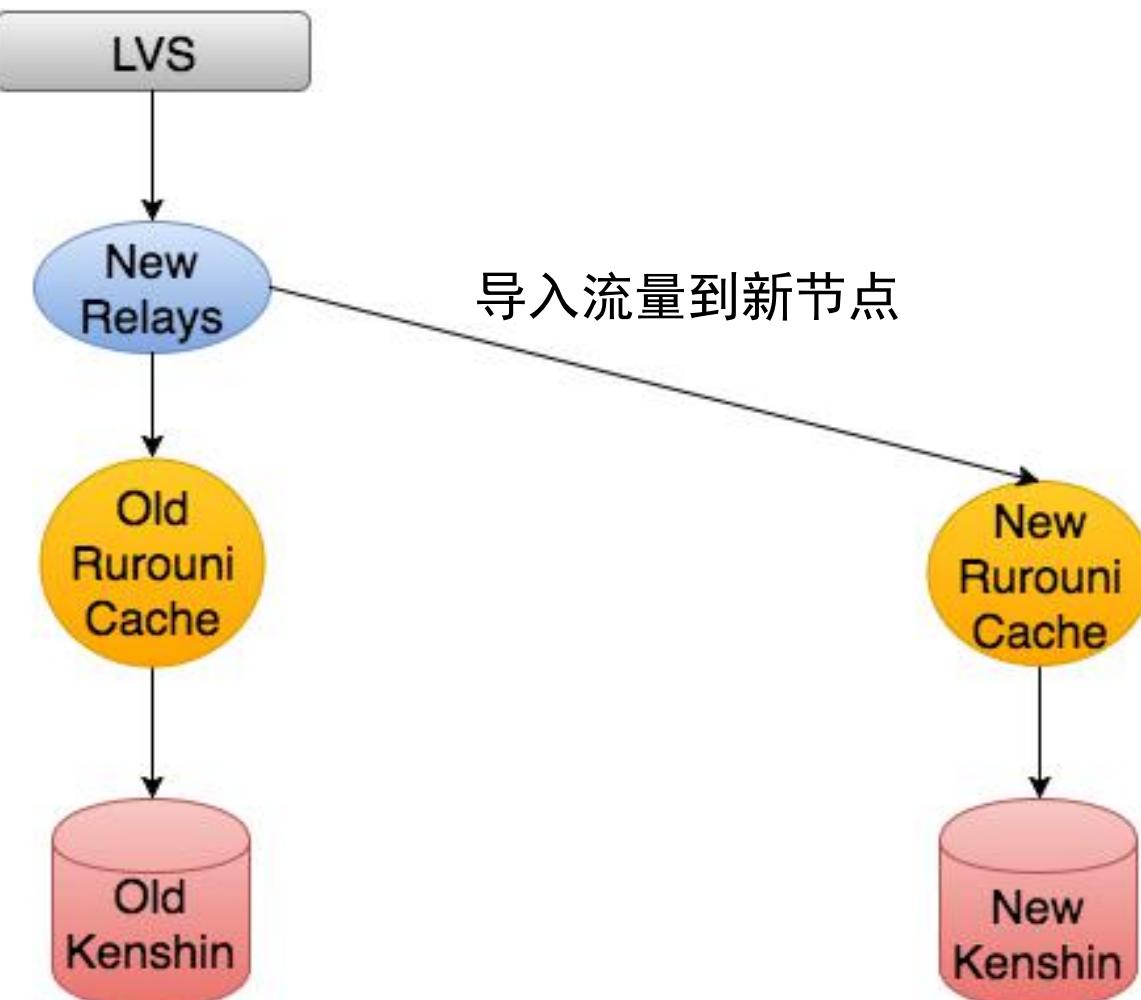
扩容

- 迁移 Hash 桶到新节点
- Rehash 重新划分 Hash 桶

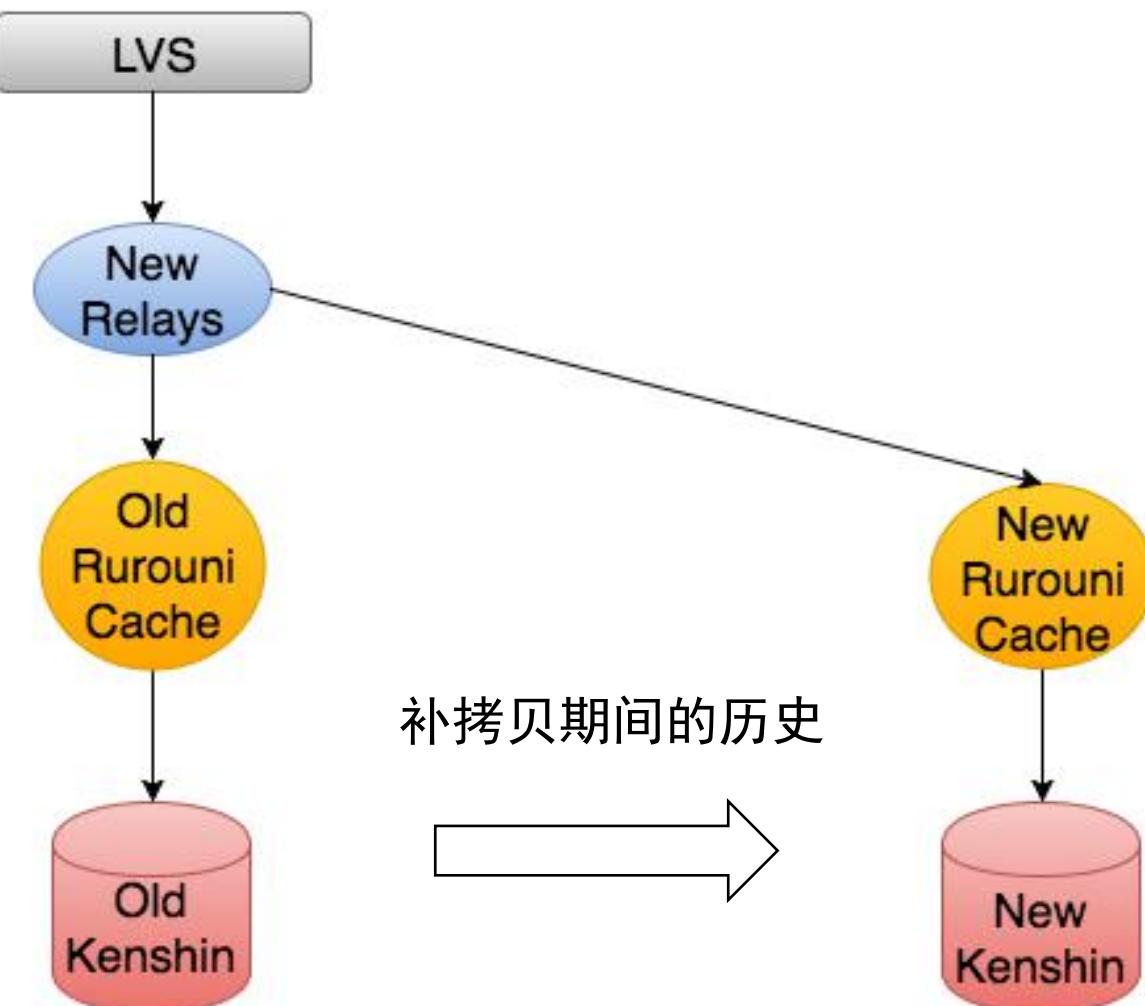
扩容 - 拷贝数据



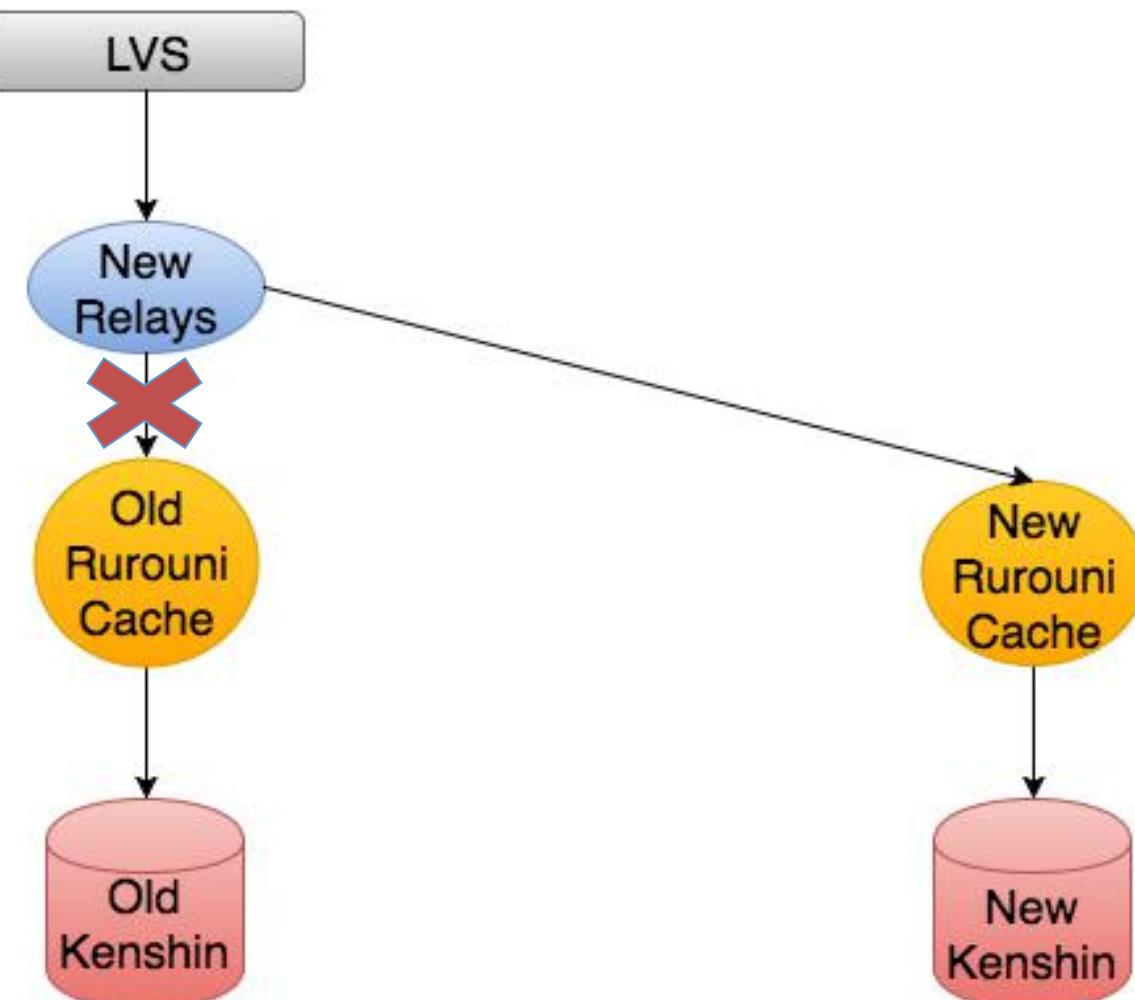
扩容 - 导入流量



扩容 - 拷贝数据



扩容 - 删除旧实例



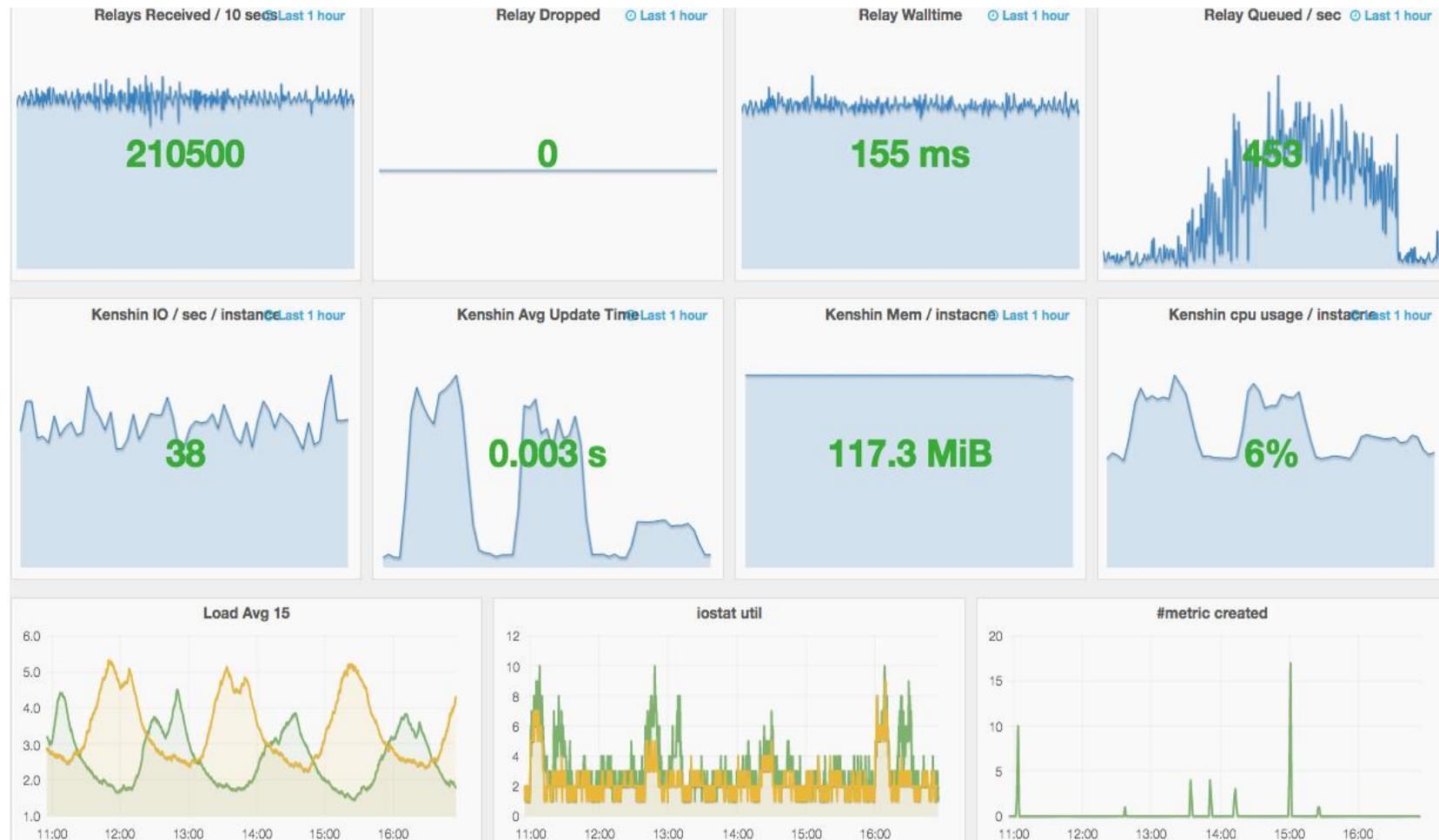
线上配置

- 2 台服务器
 - 1T SSD
 - 16G 内存
 - 8 CPU
- 指标发送间隔
 - Statsd 发送间隔 10 秒
 - Diamond 发送间隔 20 秒

线上性能

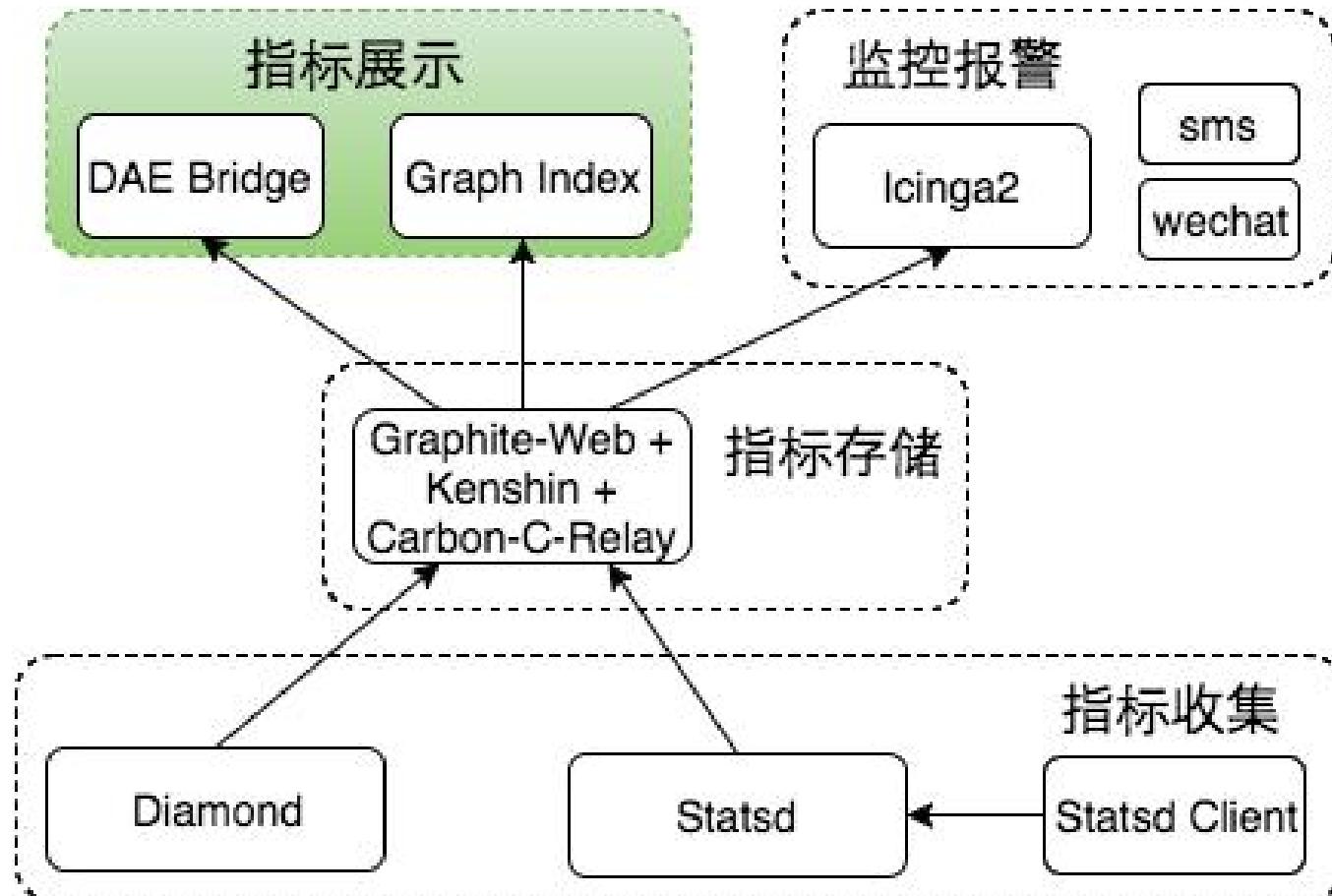
- 指标数量 (cardinality)
 - 92 万
- Carbon-C-Relay
 - 1.2 ~ 1.3 mln metrics / min
 - 120M 内存
 - 15% CPU 使用率
- Rurouni-Cache
 - 36 IOPS
 - 117.4M 内存
 - 6.0% CPU 使用率
- Graphite-Web
 - 518 metrics / second

线上性能



指标展示

指标展示



DAE Bridge



<http://> /render/?target=points%28stats.counters.dae.movie.waylife.dae.api.mc.set.rate%2C400%29&target=points%28stats.counters.dae.movie.waylife.dae.a

2016/03/29 15:40 ~ 2016/03/30 15:36



Graph-index

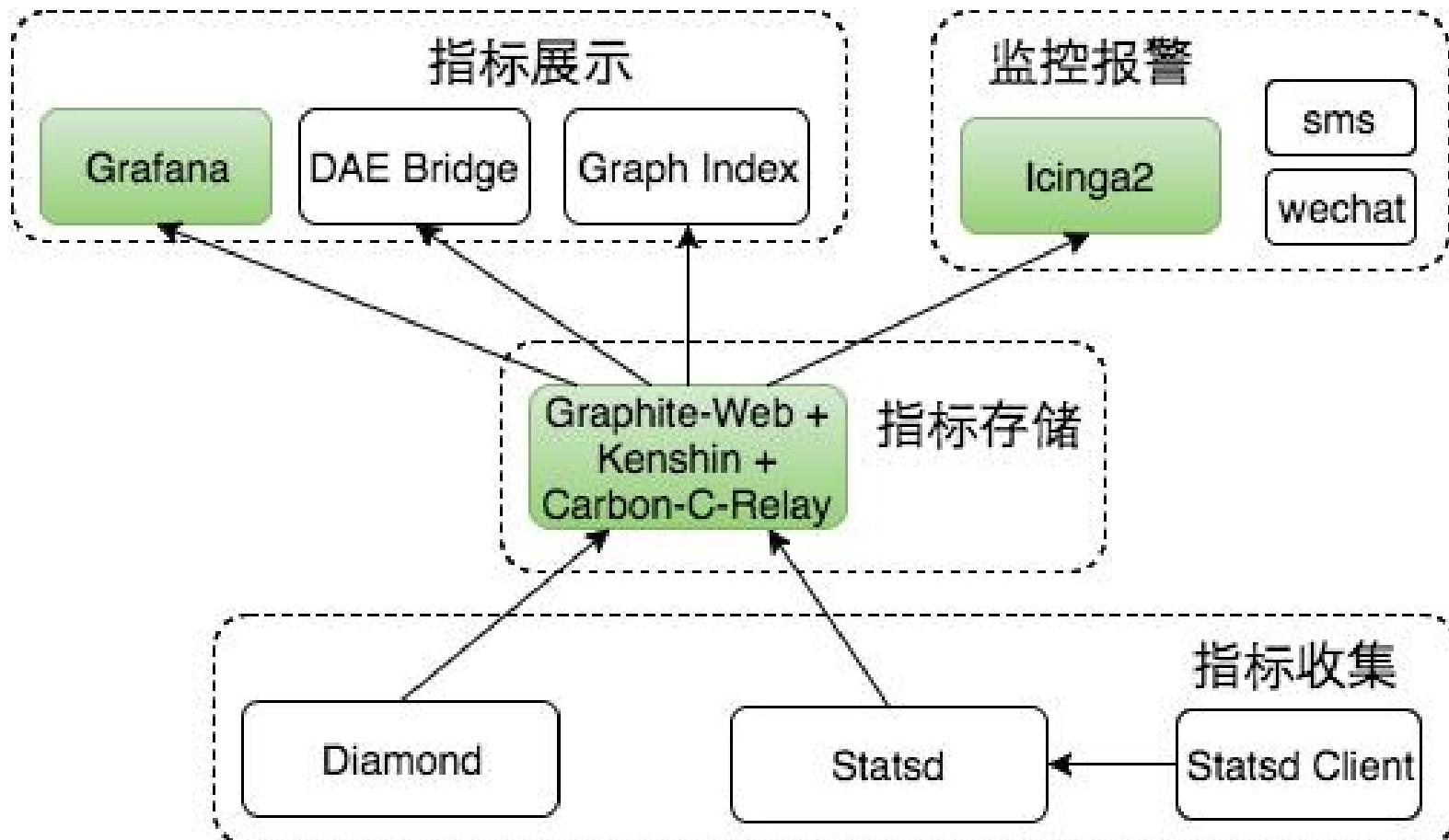
- <regex>
- <regex> group by <index>
- plugin:<plugin_name>:<prefix>
- merge:<regex>

Grafana

- 模板灵活
- 简化 graphite 函数的编写
 - $a(b(c(x)))) \rightarrow x | c | b | a$
- 自定义界面
- 配色漂亮

总结

整体架构



未来计划

- 智能的监控报警
- Kenshin 开源

THANKS!

