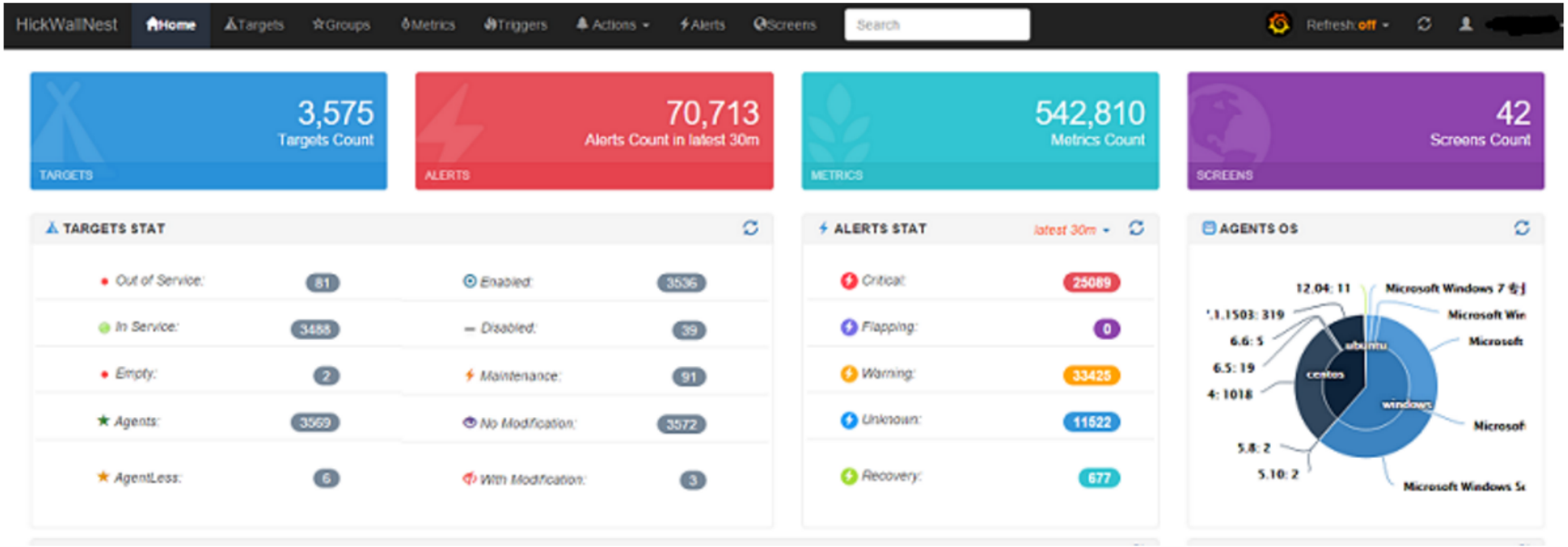
ES用于时间序列存储

- HICKWALL 监控报警系统简介

唐锐华 2016-MAY

rhtang@ctrip.com



hickwall 是一个企业级的监控告警系统。 虽然目前整体结构已经基本定型，但是仍然处于大规模的开发和重构的过程中。

这个项目的主要目标是实现多个主要用于生产的系统的监控数据的采集，存储，展现，并在此基础上配置告警，通知等。而且我们会提供一个集中的配置和展现的用户界面。通过这个界面，用户基本可以实现自助式的服务。[to be continue]

CONTENTS

- 为什么会有这个项目
- 现有开源项目的调研
- 项目整体设计与特点
- ES在使用过程中碰到的问题

为什么会有这个项目

现在的东西不能很好满足需求

为什么会有这个项目

我们的诉求:

- 稳定可靠是基础
- 高吞吐量且水平可扩展
- 高度定制化的采集
- 高度定制化的数据展示方式
- 灵活多变的数据提供方式
- 简单且兼顾复杂的告警配置
- 高度可集成
- 可以自服务
- 尽量减少排障登记时间

为什么会有这个项目

Zabbix 并不理想

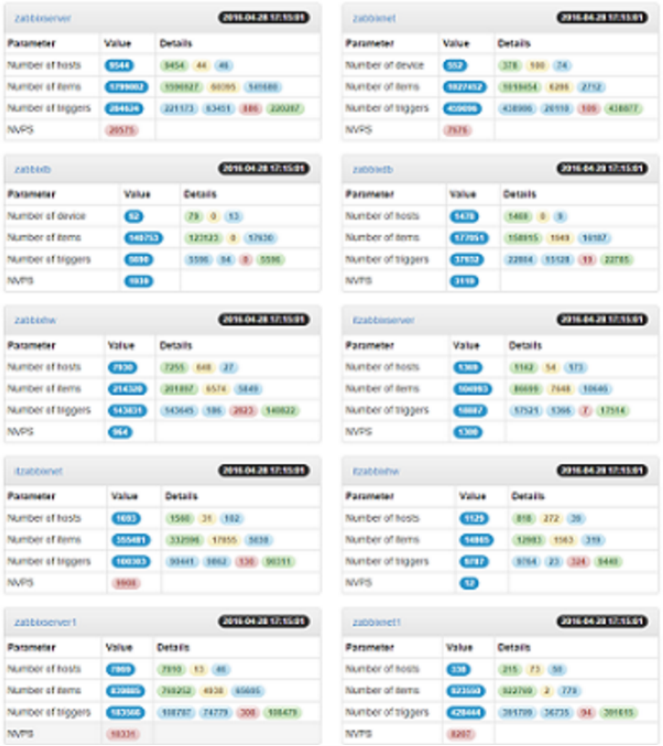
- **Zabbix 容易搭建，功能丰富稳定，但是数据吞吐量不高**

3w- / s　　40+ 台 物理机，另外 60+台虚拟机

为什么会有这个项目

Zabbix 并不理想

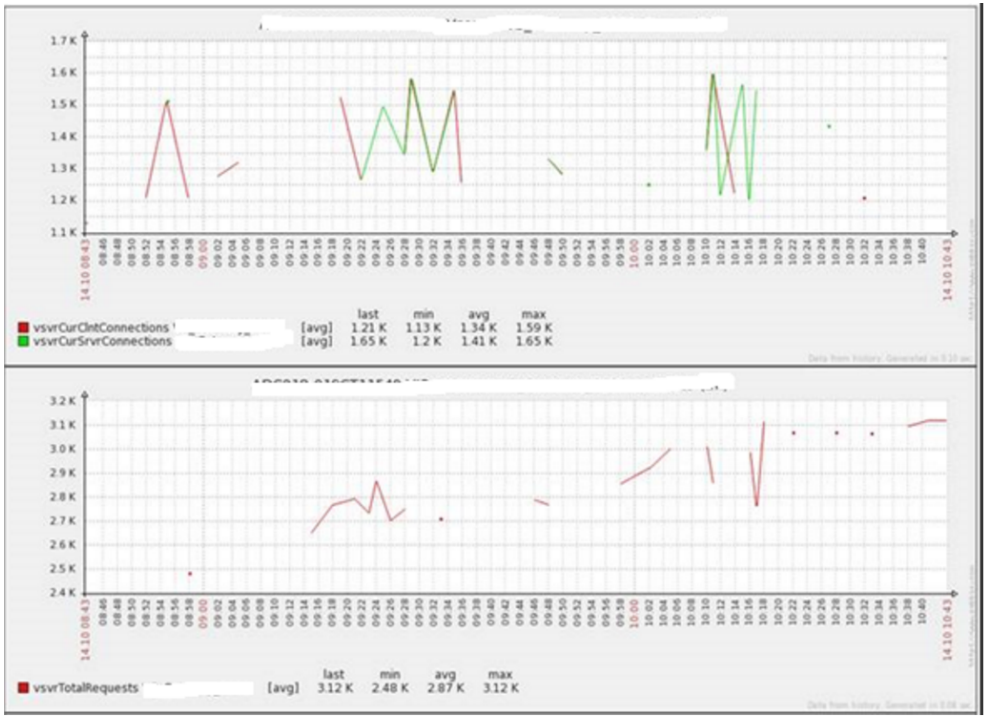
- **Zabbix 容易搭建，功能丰富稳定，但是数据吞吐量不高**
- **从zabbix里拿数据是一件让人伤心的事儿**



为什么会有这个项目

Zabbix 并不理想

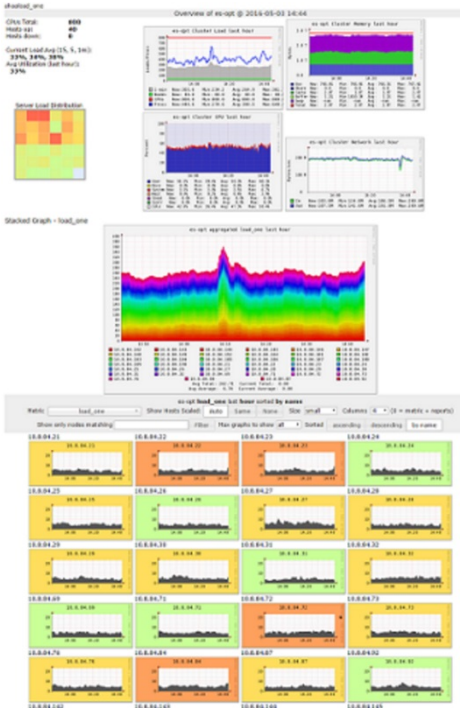
- **Zabbix 容易搭建，功能丰富稳定，但是数据吞吐量不高**
- **从zabbix里拿数据是一件让人伤心的事儿**
- **Bug 修复要等上游**



为什么会有这个项目

Zabbix 并不理想

- **Zabbix 容易搭建，功能丰富稳定，但是数据吞吐量不高**
- **从zabbix里拿数据是一件让人伤心的事儿**
- **Bug 修复要等上游**
- **定制能力比较差**



开发很多各种大大小小的监控工具

为什么会有这个项目

光有固定监控项并不能够满足排障的需要

- 很多排障确实需要的信息不一定能有固定的监控项
- 譬如：单虚拟机多应用场景的 IO Wait

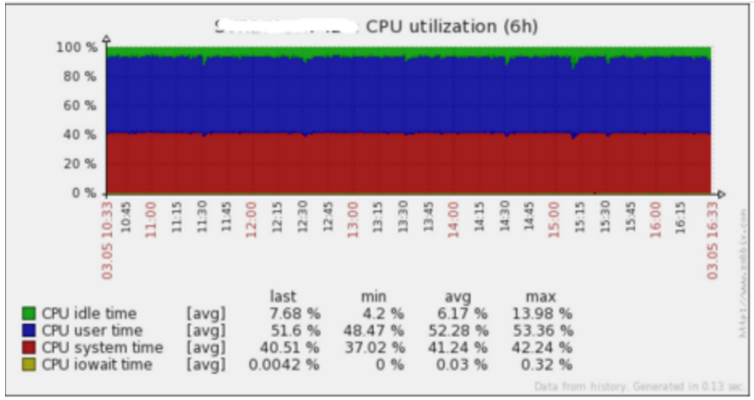
部分调研过的方案 – 以存储为例

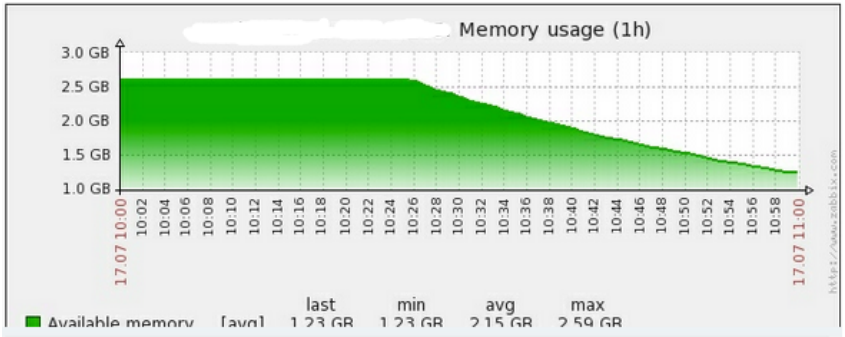
Graphite

- **De facto standard，接口简单好用**
- **Booking.com 用280台物理机存grahpite数据**

问题:

- **性能差，python开发**
- **Seek压力大**
- **伪集群**





部分调研过的方案 – 以存储为例

后起之秀 Influxdb

- 目标是解决 时间序列存储没有好用存储方案的问题
- 活跃度非常高，当前github上8k+星
- 融了资，有商业化支持

问题（我们测试到0.10 以前）：

- 存储方案频繁变更，而且我们的测试中没有能长期稳定的
- 集群方案不可用
- 配置和文档不匹配，需要看源码才行
- 各种bug，crash之后重启要等待很长时间的replay

部分调研过的方案 – 以存储为例

open-falcon

- **我们项目进行到一半open-falcon开源了**

和我们的不同:

- **RRD 存储 + hashing 伪集群**
- **不可配置**
- **Linux 为主，windows只能通过python + 计划任务**

部分调研过的方案 – 以存储为例

ES 有比较成熟的使用案例

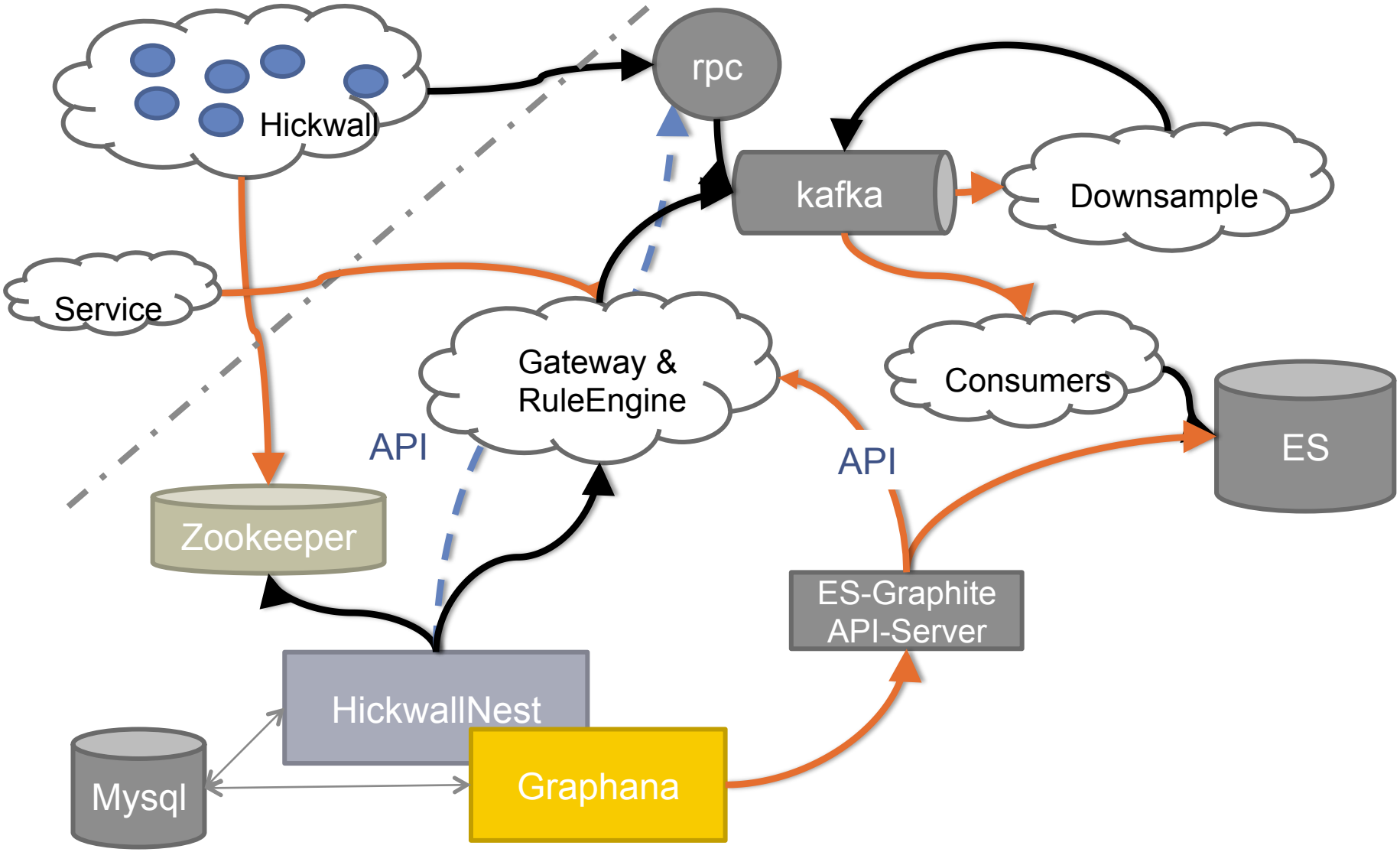
- **性能优异**
- **集群稳定性高**
- **易于维护**
- **ES用于日志分析等有很多成功应用的案例**
- **团队内部积累了丰富的使用经验**
- **生产日志的吞吐量远超监控数据的初步设计容量**

项目的整体设计与特点

功能	组件
数据采集(agent/agentless)	hickwall, gateway
配置管理	hickwallNest
数据存储	ES , redis
数据展现	grafana(改) + api-server
数据传输	rpc_proxy + kafka
告警	Gateway rule engine

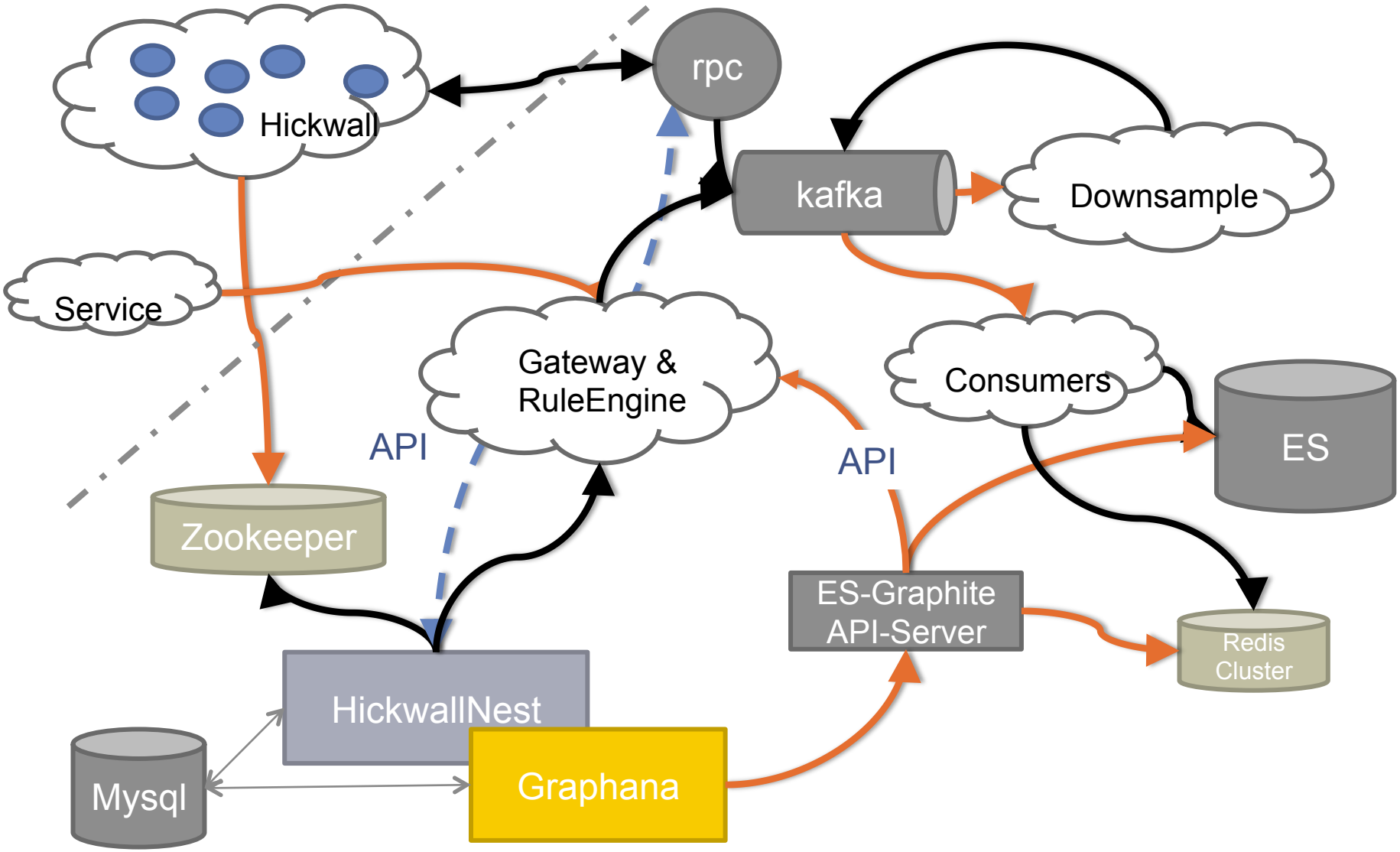
ARCHITECTURE V1.5

now



ARCHITECTURE V1.6

future



主要组件的介绍 – hickwall (agent)

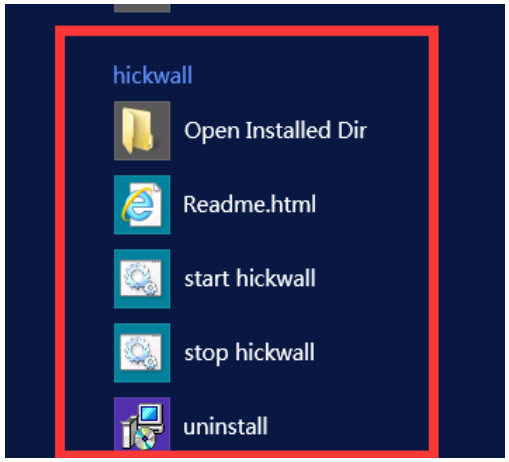
负责agent base 的采集和告警

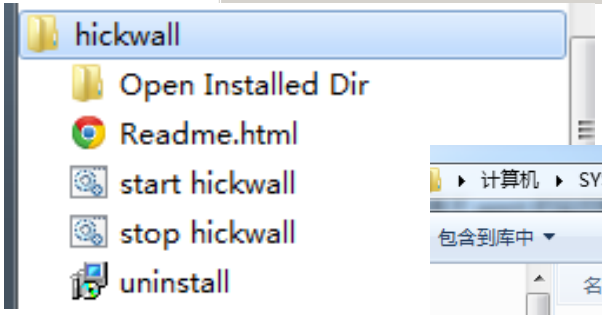
Go 1.4.2开发(支持go 1.5+)

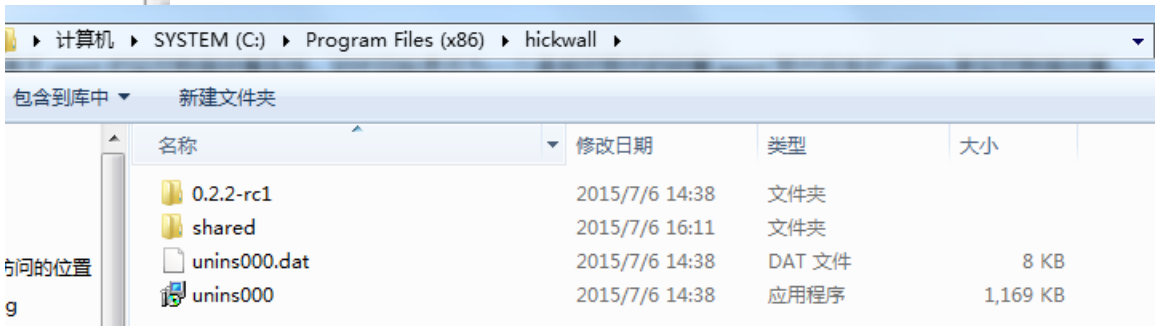
主要特点：

- 方便安装: msi, rpm, deb
- Daemonize: hickwall service install/start/stop/uninstall
- 可配置: 文件配置/etcd/zookeeper/集中式管理的配置
- 支持windows，linux，之后会考虑增加mac支持
- 多种backend支持。主要精力集中在kafka/rpc_proxy
- 提供少量API，未来会增加很多实时数据的api
- 进程级监控，目前正在优化linux下的实现方式
- 脚本， 告警DSL
- TopN
- Mmap 本地数据共享
- Docker 监控在计划中

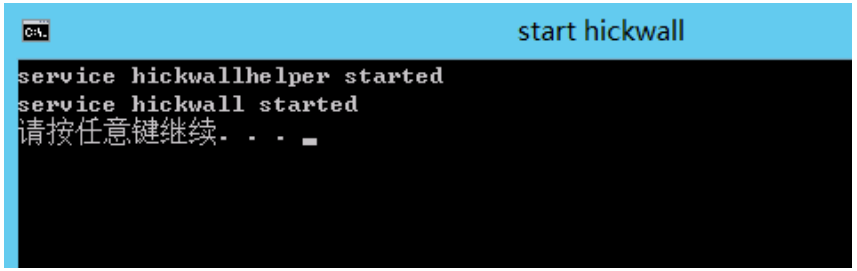
主要组件的介绍 – hickwall (agent)





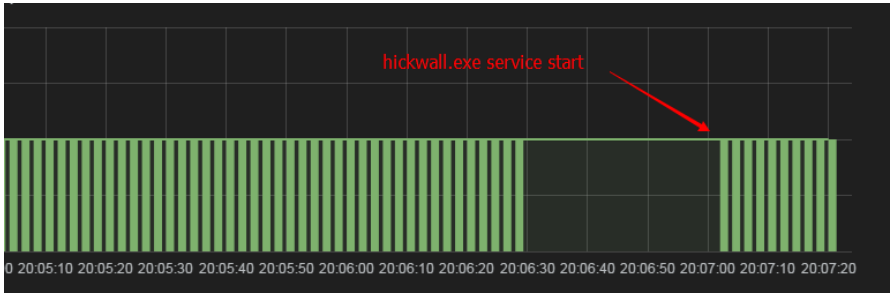


主要组件的介绍 – hickwall (agent)



名称	PID	描述	状态	工作组
HomeGroupProvider		HomeGroup Provider	已停止	LocalServiceNetworkRes
HomeGroupListener		HomeGroup Listener	已停止	LocalSystemNetworkRest
hkmsvc		Health Key and Certificate Manage...	已停止	netsvcs
hidserv	780	Human Interface Device Access	正在运行	LocalSystemNetworkRest
hickwallhelper	47460	hickwall helper service	正在运行	暂缺
hickwall	46796	monitoring system	正在运行	暂缺
gupdatem		Google 更新服务 (gupdatem)	已停止	暂缺
gupdate	30920	Google 更新服务 (gupdate)	正在运行	暂缺
gpsvc	504	Group Policy Client	正在运行	netsvcs
ftpsvc		Microsoft FTP Service	已停止	ftpsvc
FontCache3.0.0.0		Windows Presentation Foundation F...	已停止	暂缺

主要组件的介绍 – hickwall (agent)




```
sys_collector_interval: 5s

groups:
-   prefix: "oletest"
    collector_win_pdh:
    -   interval: "30s"
        queries:
        -   query: "\\.NET CLR Exceptions(_Global_)\\# of Exceps Thrown / sec"
        -   query: "\\.NET CLR Exceptions(_Global_)\\# of Exceps Thrown"
        -   query: "\\.NET CLR LocksAndThreads(_Global_)\\# of current logical threads"
        -   query: "\\.NET CLR LocksAndThreads(_Global_)\\Contention Rate / sec"
        -   query: "\\.NET CLR LocksAndThreads(_Global_)\\Current Queue Length"
        -   query: "\\.NET CLR Memory(_Global_)\\# Bytes in all Heaps"
        -   query: "\\.NET CLR Memory(_Global_)\\# Gen 0 Collections"
        -   query: "\\.NET CLR Memory(_Global_)\\# Gen 1 Collections"
        -   query: "\\.NET CLR Memory(_Global_)\\# Gen 2 Collections"
        -   query: "\\.NET CLR Memory(_Global_)\\# of Pinned Objects"
        -   query: "\\.NET CLR Memory(_Global_)\\% Time in GC"
        -   query: "\\.NET CLR Memory(_Global_)\\Allocated Bytes/sec"
        # -   query: "\\.NET CLR Memory(_Global_)\\Large Object Heap size"
```

Name:	dst54869
hickwall.version:	0.2.5
hickwall.build:	c5c7be8
OS:	windows
PlatformVersion:	Microsoft Windows 7 专业版 Service Pack 1 - 6.1.7601
IP:	[redacted]
Agent Api Port:	[redacted]
Domain:	cn1.global.ctrip.com
NumberOfProcessors:	1
NumberOfLogicalProcessors:	4
TotalPhsycialMemoryKb:	8179664
Architecture:	64
NetworkInterfaces:	**IPv4:** [redacted] **Name:** {[redacted]1AAA44108} **HardwareAddr:** [redacted] **Description:** Intel(R) 82579LM Gigabit Network Connection

Macro:

主要组件的介绍 - CONSUMER

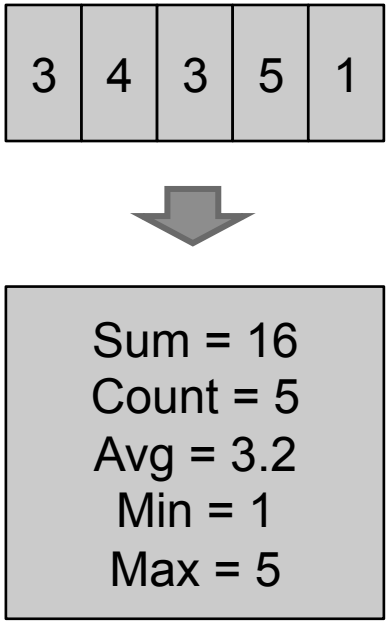
consumer – java

- 从kafka中消费各个metric topic，存储到ES， redis中
- 水平可扩展
- jmx 数据吐到报警系统的graphite中

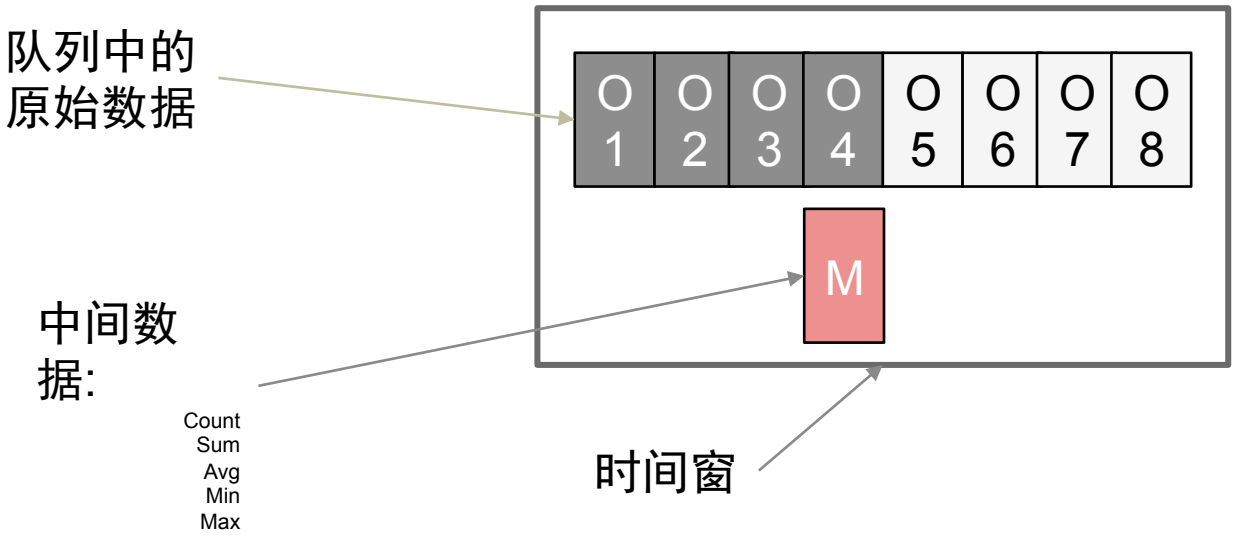
主要组件的介绍 - CONSUMER

主要组件的介绍 - DOWNSAMPLE

什么是Downsample



主要组件的介绍 - DOWNSAMPLE



Drawbacks：

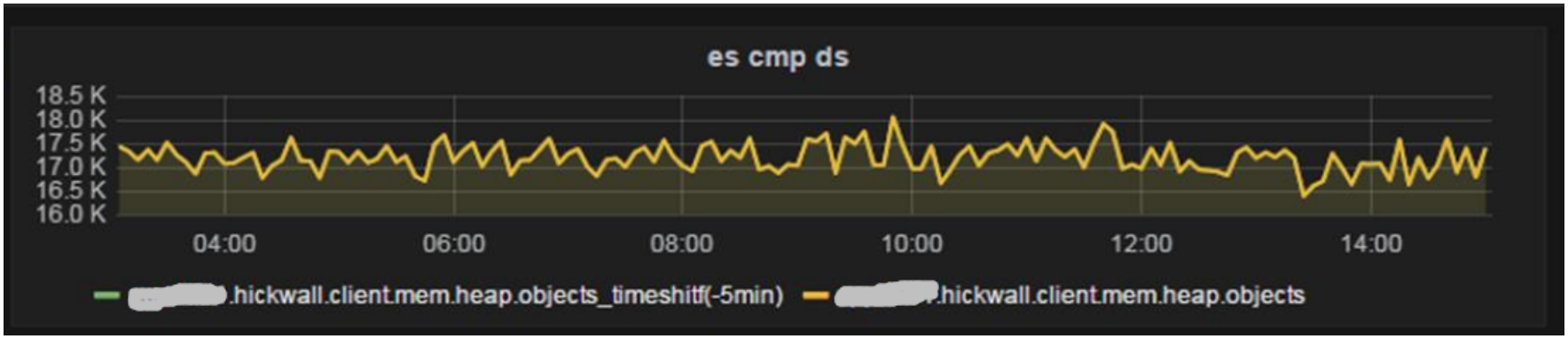
- Rawdata -> 中间结果 partition 一一对应
- 修改kafka库，实现rebalance通知
- 中间结果数据量也非常大

主要组件的介绍 - DOWNSAMPLE

downsample – java

- 从kafka中消费原始数据，根据配置的downsample schema来downsample. 支持avg, sum两种
- 根据采集到数据的meta来决定downsample的方式。 avg 或者 sum
- 会保留数据的tag，每一种tag会有一个单独的结果
- 结果和用ES原始数据做的聚合保持一致和RDD的结果保持一直
- 可以水平扩展，reblance不丢数据
- jmx 数据吐到报警系统的graphite中

主要组件的介绍 - DOWNSAMPLE



目前持续稳定，ES计算的数据和Downsample的数据保持一致

主要组件的介绍 - DOWNSAMPLE

Name	Method	Status	Type	Initiator	Size	Time	Timeline – Start Time 1.00 s 1.50 s 2.00 s
render	POST	200	xhr	angular.js?bus...	4.4 KB	71 ms	
render	POST	200	xhr	angular.js?bus...	20.3 KB	2.09 s	

原始数据 2.09s
Downsample数据 71ms

Downsample 仍然保留了tag

主要组件的介绍 – API SERVER

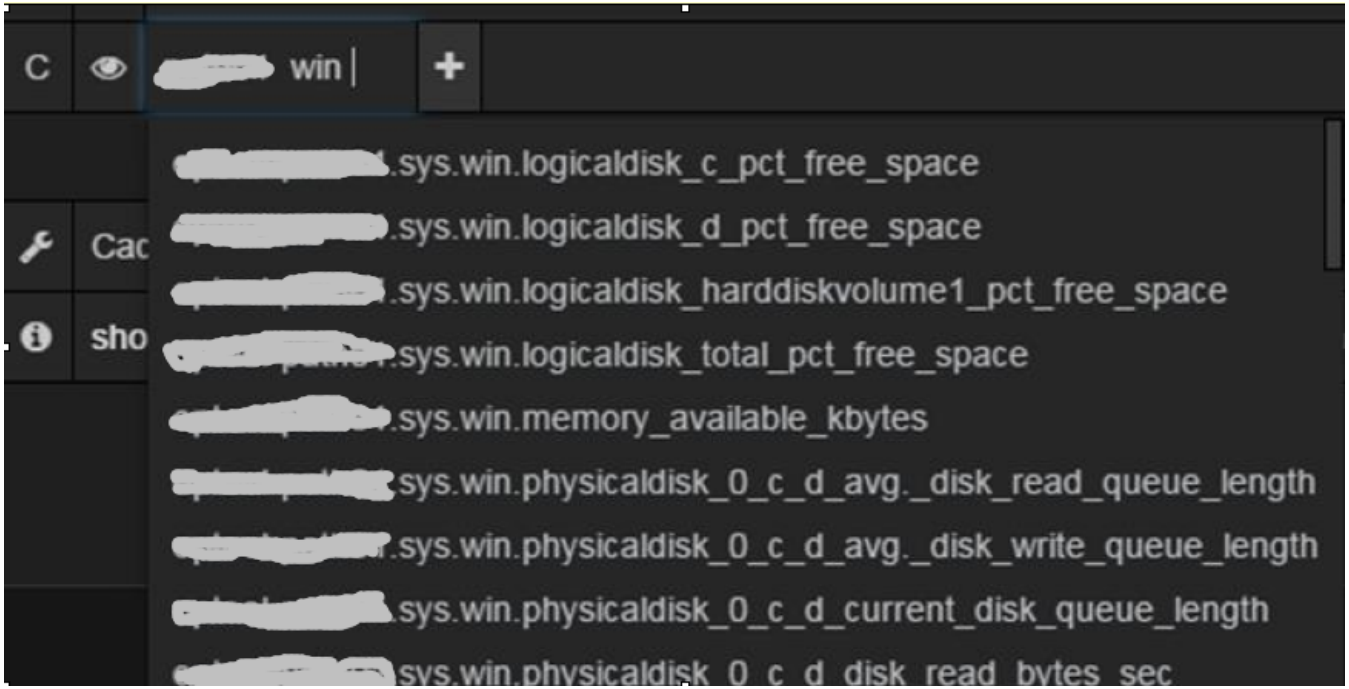
API SERVER – go & scala

- 负责向内部组件 和 第三方提供数据
- 兼容 graphite api，实现了大部分grapihte api 的function
- 根据用户的查询时间跨度自动决定返回数据的精度
- 变态查询过滤，优化查询，合理利用后端存储
- 隔离后端存储，用户不需要关心存储的最终方案
- 数据缓存
- go是遗留问题，以后会抽空全面转向scala

主要组件的介绍 – API SERVER

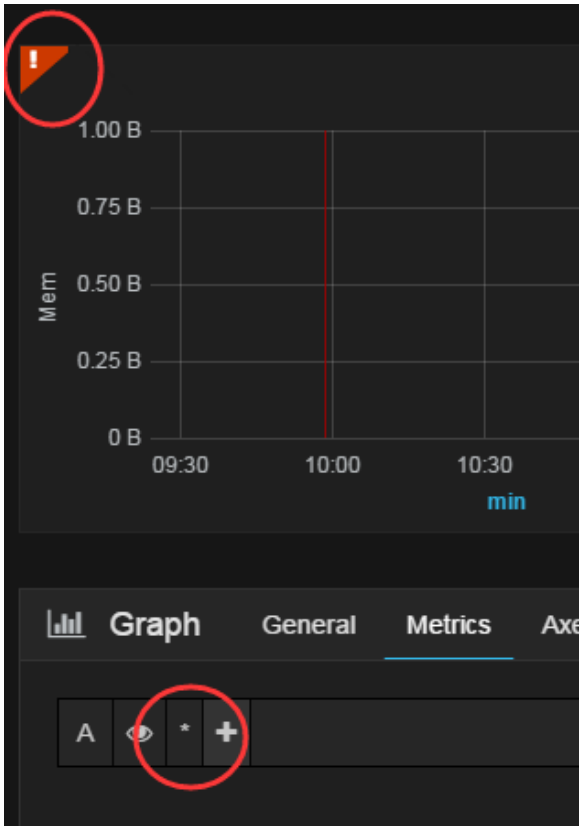
Grafana看的数据全部是通过api-server获取的

主要组件的介绍 – API SERVER

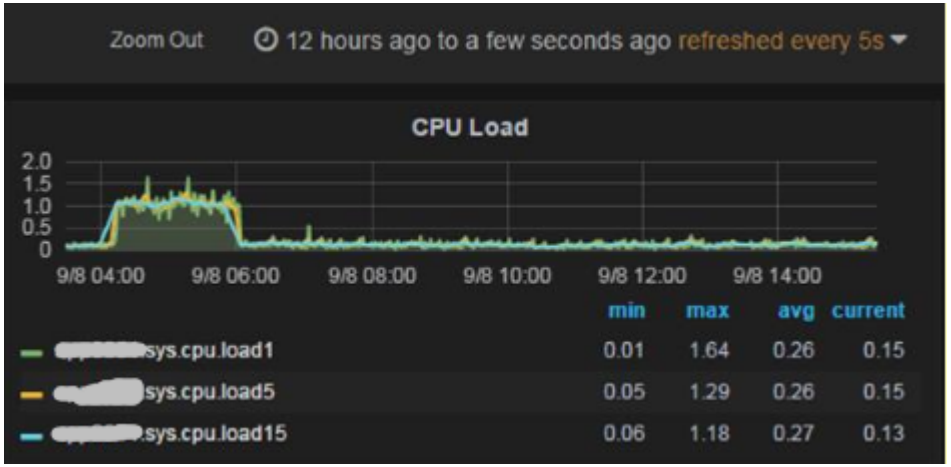


Metric name 自动提示（利用ES的搜索功能 ）

主要组件的介绍 – API SERVER



变态查询过滤



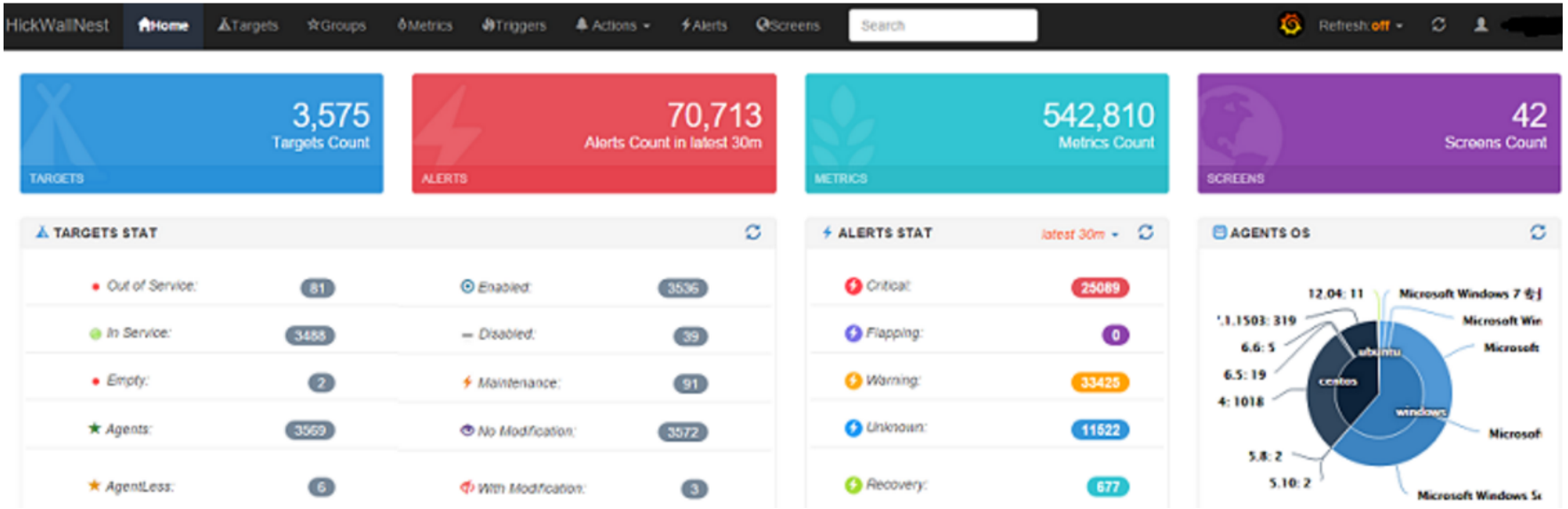
根据用户选择的range自动返回DS的数据

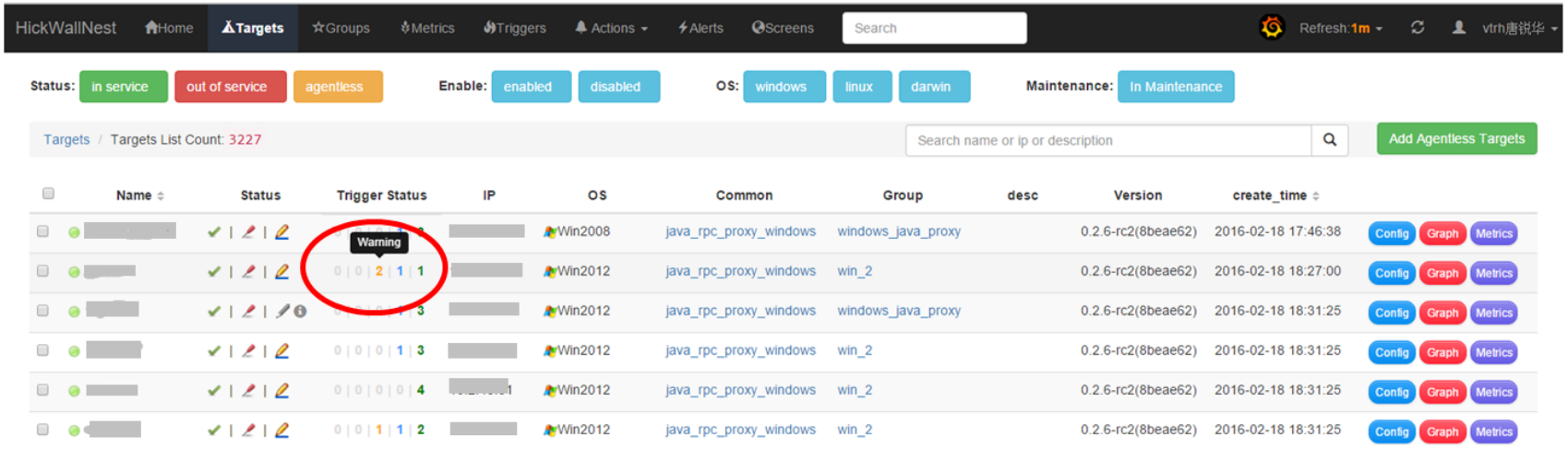
主要组件的介绍 - NEST

Nest – python & scala

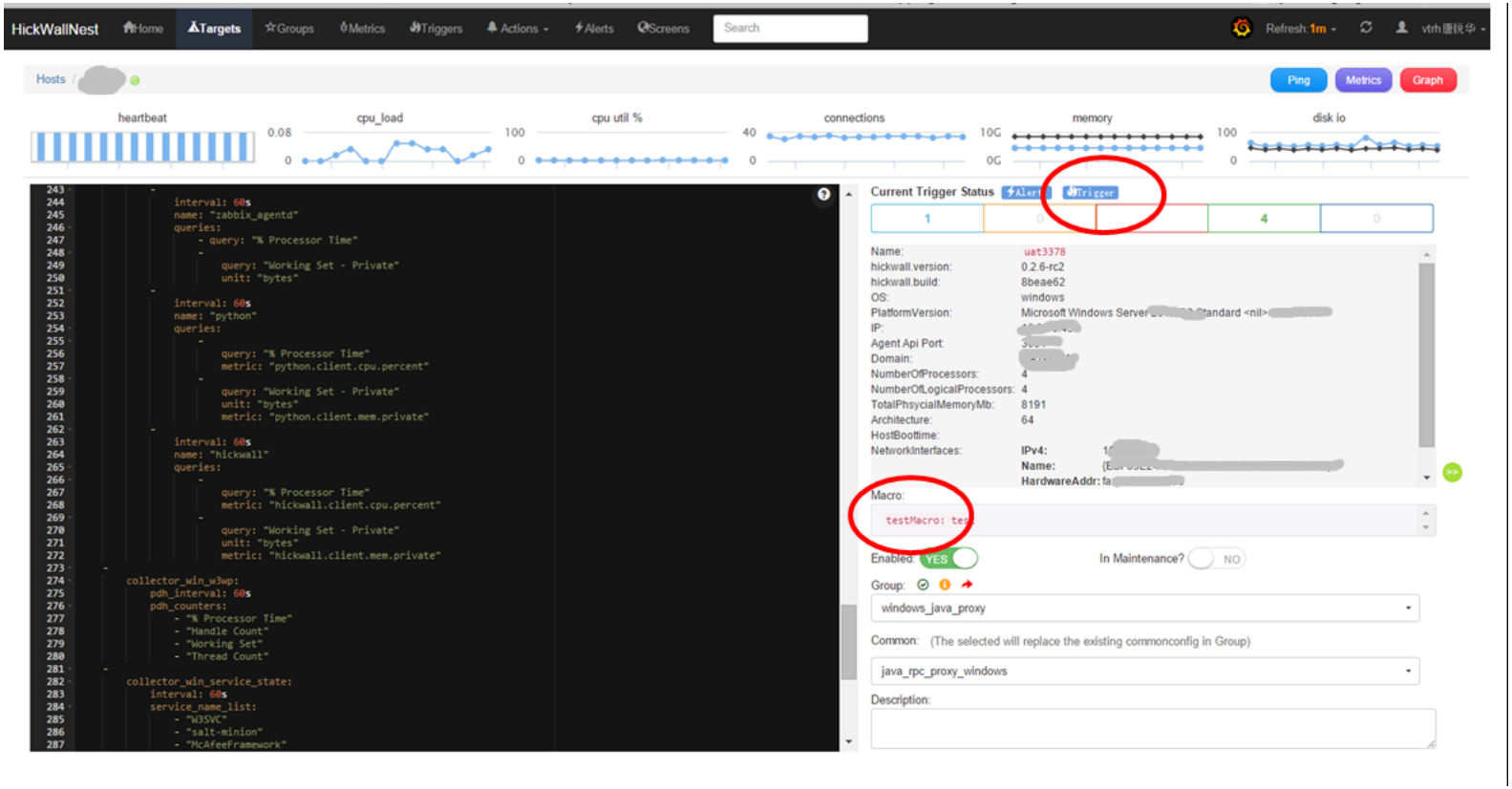
- 集中式的配置管理平台
- YAML 的采集配置方式
- Javascript 作为告警DSL, 支持宏定义
- 灵活的告警通知机制：邮件，短信等，工作日，按时间段等
- 便捷的告警搜索过滤
- screen模板管理
- RBAC权限管理
- 自服务
- 将来会提供更多实时数据查看工具以减少排障的登机时间
- Python 的部分将会慢慢被scala所替代

主要组件的介绍 - NEST

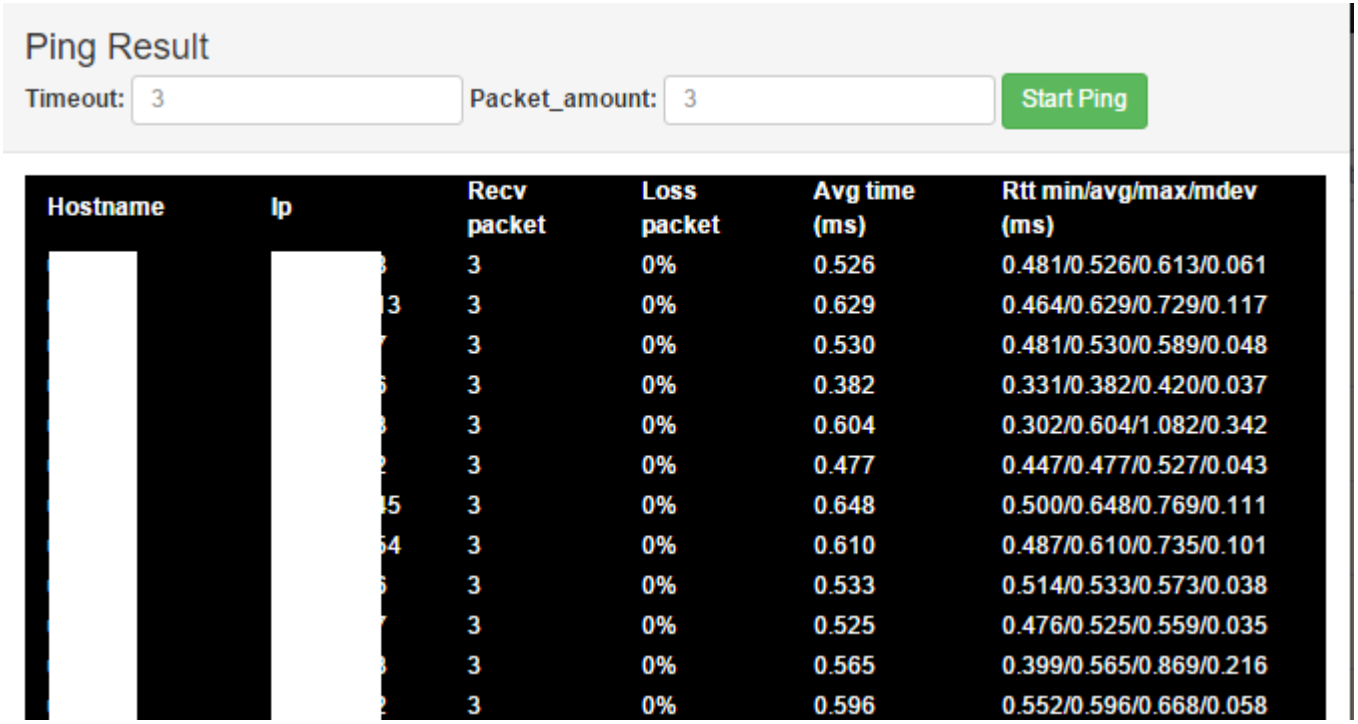




主要组件的介绍 - NEST



主要组件的介绍 - NEST



主要组件的介绍 - NEST

跳转到定制的dashboard

主要组件的介绍 - NEST

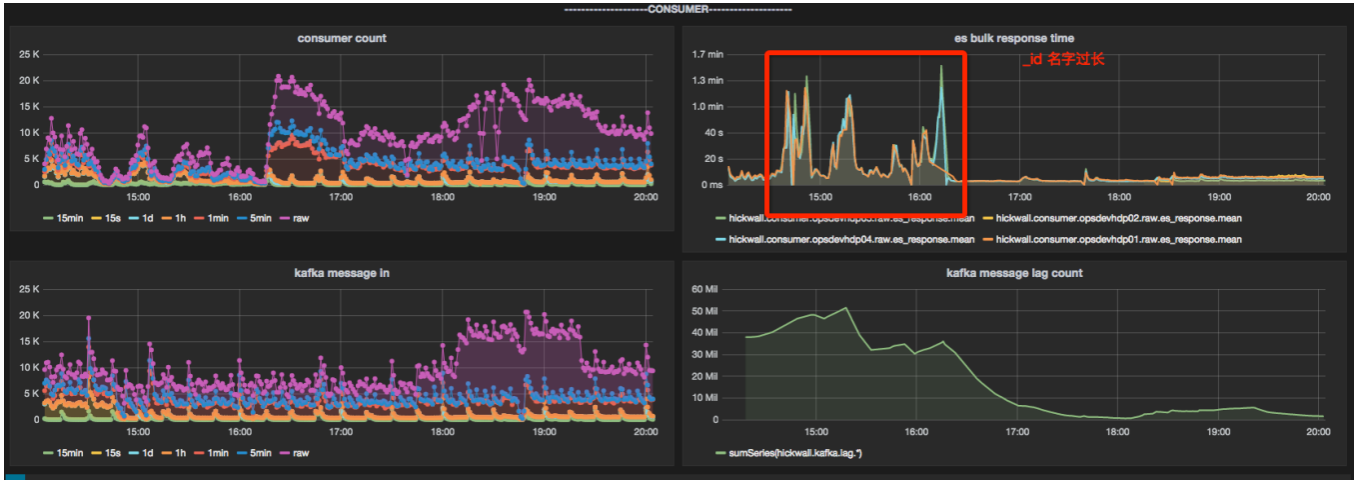
ES在使用过程中碰到的问题

- _id 定制存在性能问题
- 数据的延时比较大
- 查询cache利用率不好提高
- 无法简单实现 Latest Data，现在我们用redis缓存来实现
- 内嵌文档无法Add
- 静态Tag 和动态 Tag的取舍，topN

ES在使用过程中碰到的问题

_id 定制存在性能问题

- **主要是用来去重**
- **过长最不可取, 即使用缩短也有问题**
  - hash(metric + time)
  - hash(metric) + time
  - hex(hash(metric)) + time



ES在使用过程中碰到的问题

数据延时大

- **ES refresh time 对读写性能影响大，所以需要一个合适的值**
- **正常情况下**
  - 延迟 = 传输/处理时间 + ES refresh 时间
  - 传输/处理时间 ~= 8ms （经验值）
  - ES refresh 时间 30 秒
- **异常情况下**
  - ES出现任何异常，造成kafka堆积，数据延迟会被堆积时间放大很多
  - 譬如kafka堆积 5分钟， 数据延迟可能超过10分钟
  - 这是因为ES是最终的数据存储，尽量不希望数据丢失
  - 如果不考虑数据丢失的问题，延时就基本和正常情况一样

ES在使用过程中碰到的问题

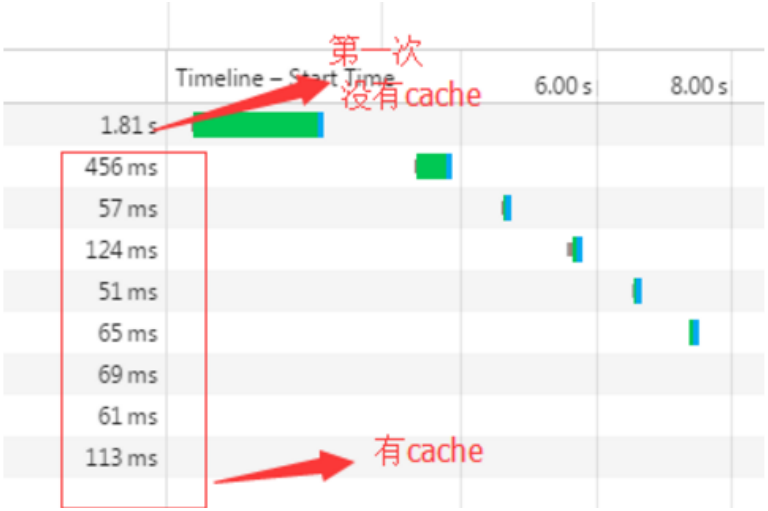
查询cache利用率不好提高

- 通过truncate 时间的方式增加 连续多次query cache的利用率

  05分 31 秒 -> 06 分

  05分 49 秒 -> 06 分

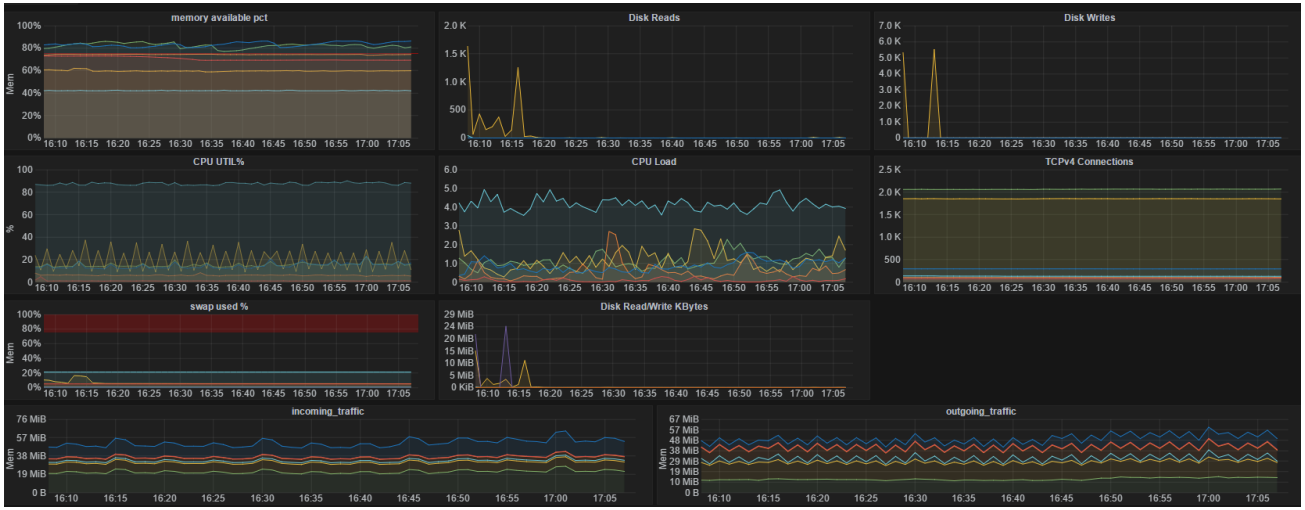
- 查询的目标，时间跨度等随机性比较高
- 查询返回的数据量比较大



ES在使用过程中碰到的问题

查询cache利用率不好提高

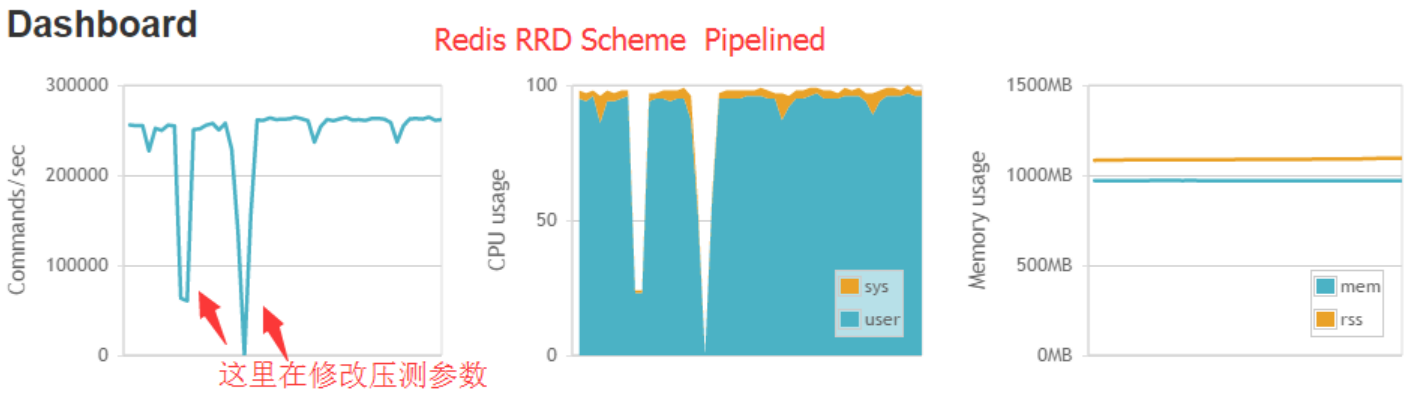
- 通过truncate 时间的方式增加 连续多次query cache的利用率
- 查询的目标，时间跨度等随机性比较高
- 查询返回的数据量比较大



ES在使用过程中碰到的问题

无法简单实现 Latest Data，现在我们用redis缓存来实现

- **数据到达时间不一致，无法简单用时间段来过滤最近N个点**
- **单这种查询在告警规则中非常有用**
- **Redis cache 实现效果更好**
  - metric: updatedon, latest, 0, 1, 2, 3, 4, …. 60
  - Codis 集群方案, 1000 pipline 性能最好



time	us	sy	cl	bcl	mem	rss	keys	cmd/s	exp/s	evt/s	hit%/s	hit/s	mis/s	aofcs
16:18:33	96	2	12	0	970MB	1.07GB	1.20M	262k	0	0	-	0	0	0B
16:18:28	96	2	12	0	970MB	1.07GB	1.20M	261k	0	0	-	0	0	0B

ES在使用过程中碰到的问题

内嵌文档无法Add

- **内嵌文档只能读出来之后再把新的数据一块写进去**
  - Time1, Doc1 [subdoc1]
  - Time2, Doc1 [subdoc1, subdoc2]
- **想到的可行的做法：前边有一个聚合的引擎（譬如 现成的redis），每小时用内嵌文档写一批数据进去**
  - 优点：内嵌文档能共享很多字段，数据量减少，性能都会提高
  - 缺点：数据需要分成 `实时数据` 和 `非实时数据`，复杂性会提高。查数据需要查两部分数据拼接起来。

ES在使用过程中碰到的问题

静态Tag 和动态 Tag的取舍，topN

- **静态Tag变动少，可以和数据分开存放，性能好**
- **动态Tag变动频繁，需要和数据在一块，性能差，但是灵活性高**
- **TopN的采集就是利用动态Tag的方式**
  - cpu-util，time1，value1, top1=xxxx, top2=xxxx,top3=xxx
  - cpu-util，time2，value2, top1=xxxx, top2=xxxx,top3=xxx

Q&A