酷狗大数据平台架构重构

王 劲

2015.12

酷狗音乐

目录

重构原因--原有架构

重构原因

数据采集	• 数据收集接口众多，且数据格式混乱，基本每个业务都有自己的上报接口
数据接入	• 直接从接入服务通过rsync同步文件 • 没有数据监控服务
数据清洗	• ETL集中在作业计算前进行处理 • 存在重复清洗
作业调度	• 大部分作业通过crontab调度 • 经常出现作业调度冲突
平台监控	• 只有硬件与操作系统级监控

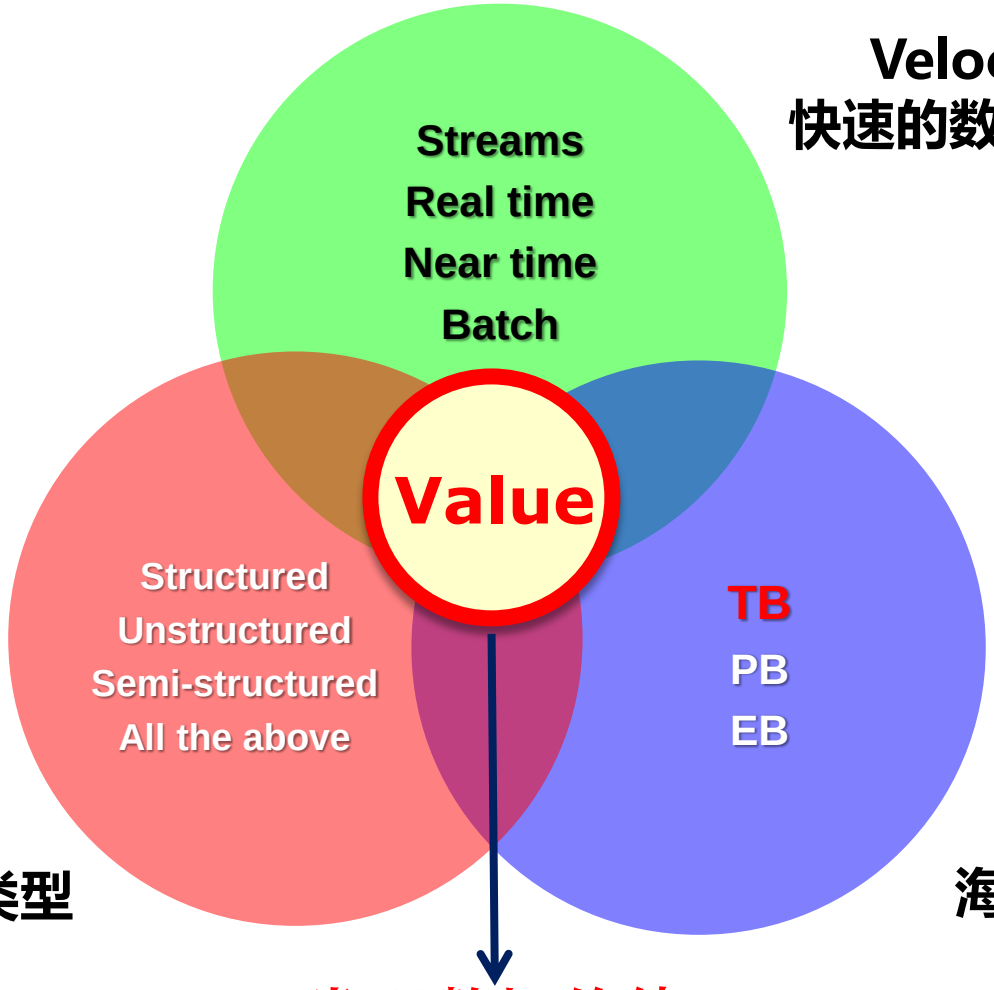
目录

技术架构--大数据的4V特征

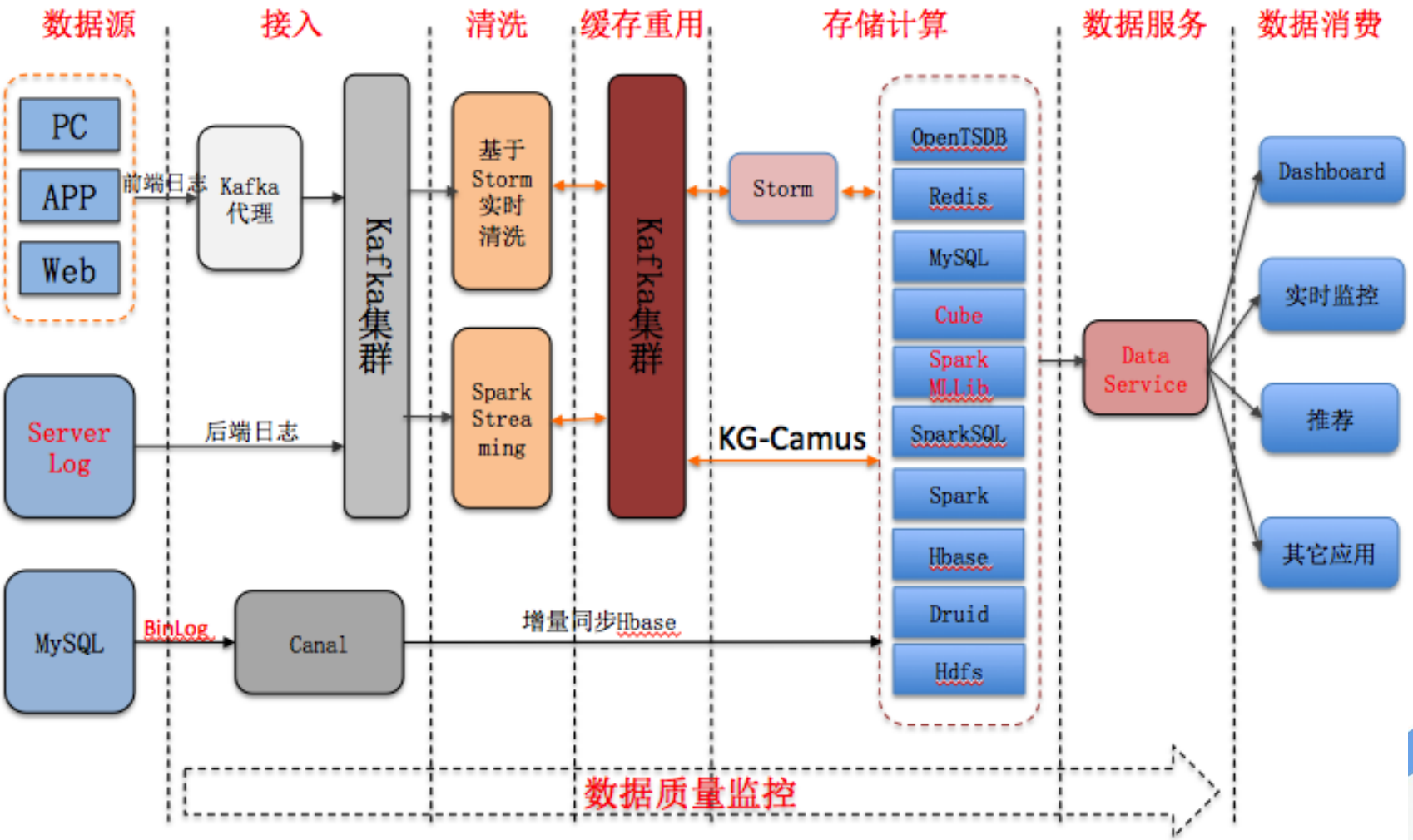
体量Volume	非结构化数据的超大规模和增长 总数据量的80~90% 比结构化数据增长快10倍到50倍 是传统数据仓库的10倍到50倍
多样性Variety	大数据的异构和多样性 很多不同形式（文本、图像、视频、机器数据） 无模式或者模式不明显 不连贯的语法或句义
价值密度Value	大量的不相关信息 对未来趋势与模式的可预测分析 深度复杂分析（机器学习、人工智能Vs传统商务智能(咨询、报告等）
速度Velocity	实时分析而非批量式分析 数据输入、处理与丢弃 立竿见影而非事后见效

“大量化(Volume)、多样化(Variety)、快速化(Velocity)、价值密度低（Value）”就是“大数据”的显著特征，或者说，只有具备这些特点的数据，才是大数据。

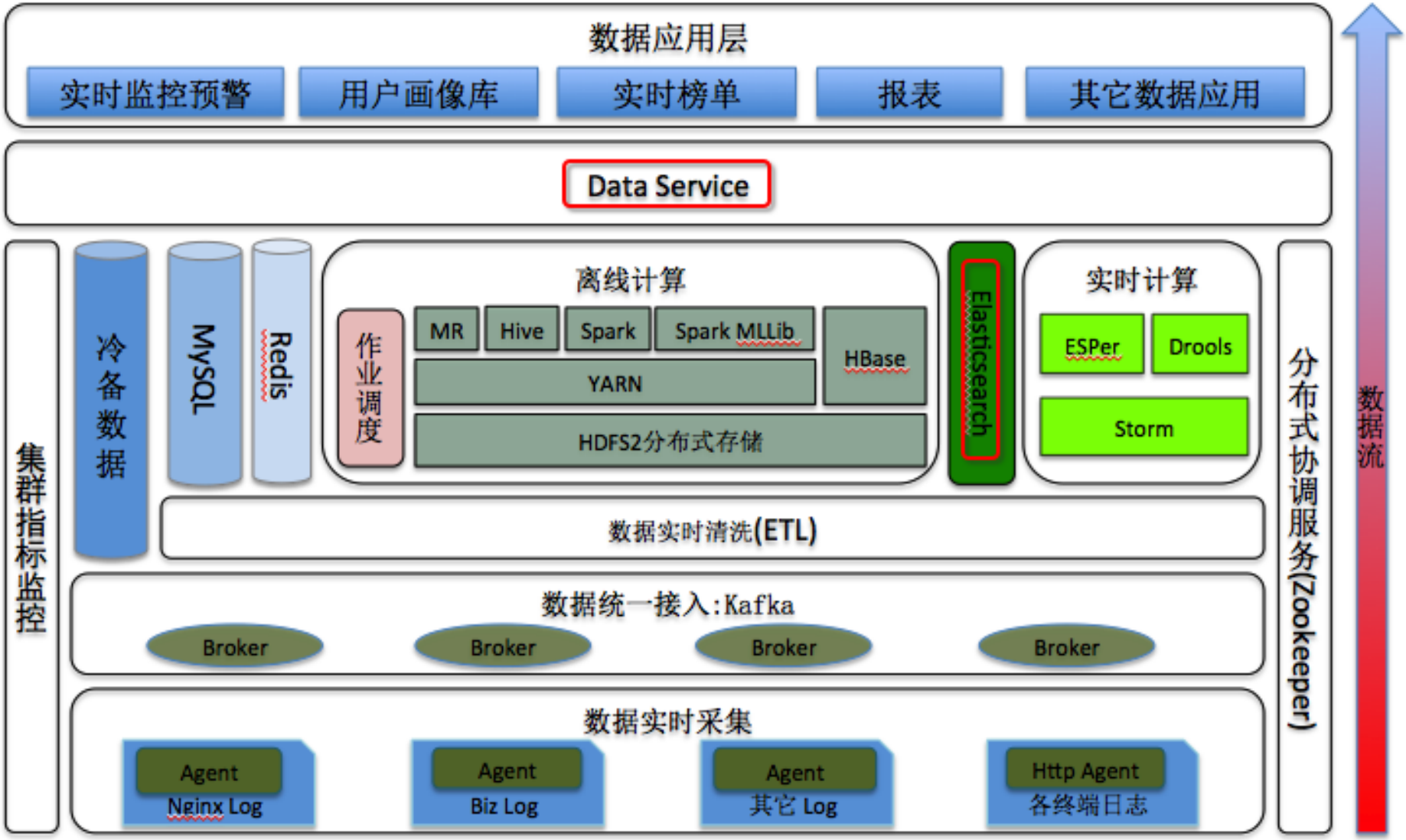
技术架构--要解决的问题



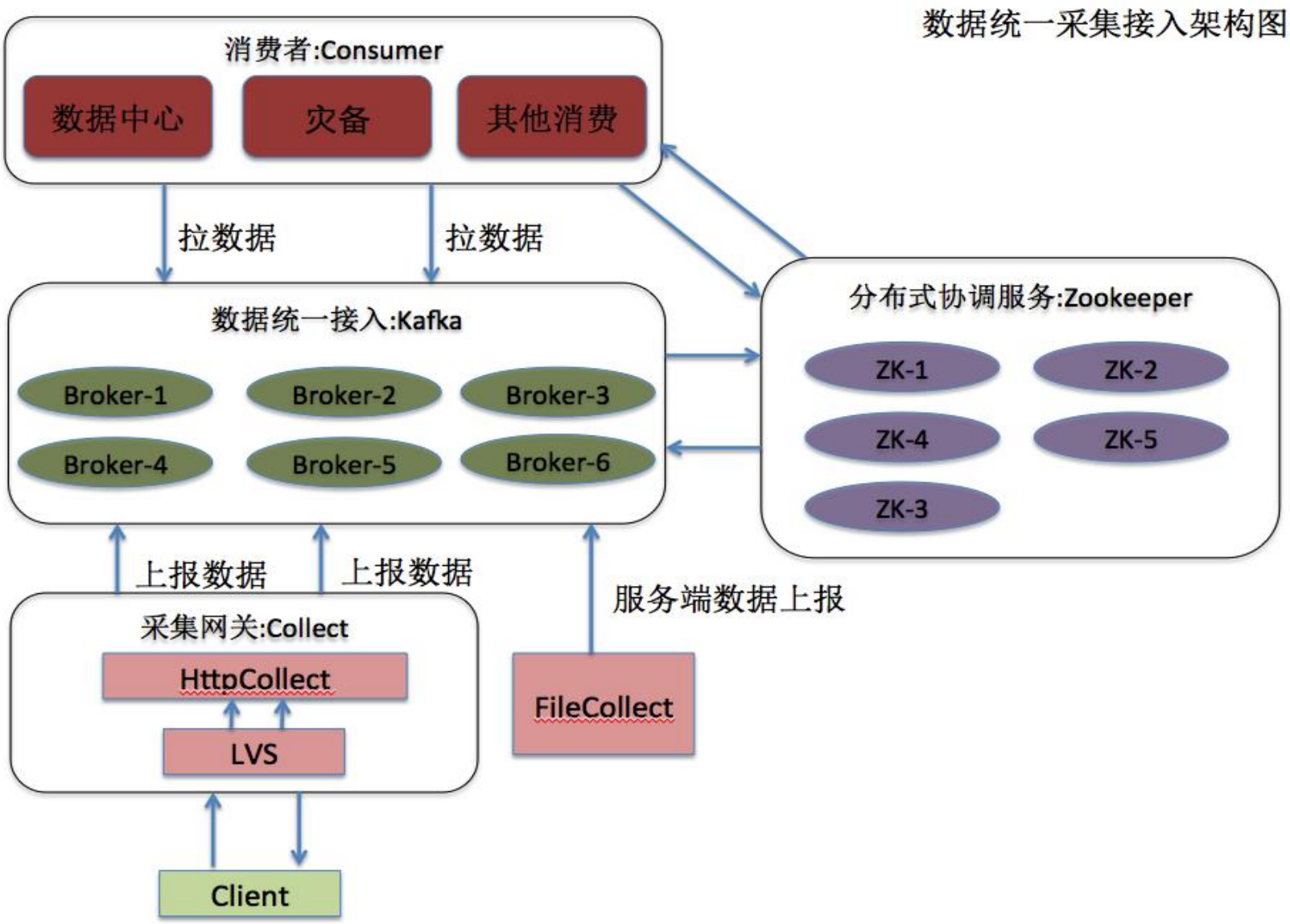
技术架构--数据流架构



技术架构--整体技术架构



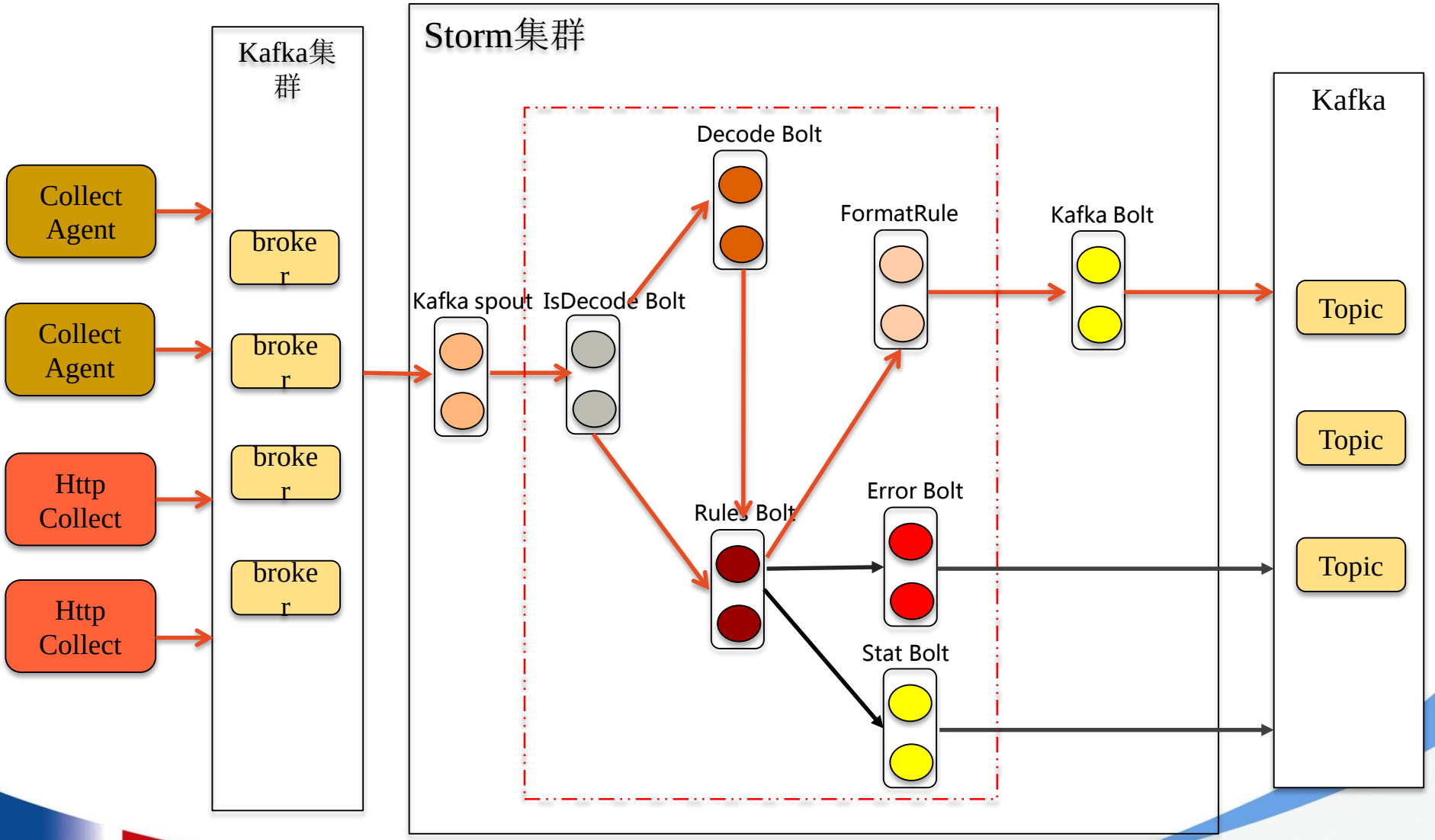
技术架构--数据采集接入


数据统一采集接入架构图
消费者:Consumer
数据中心
灾备
其他消费
拉数据
拉数据
数据统一接入:Kafka
Broker-1
Broker-2
Broker-3
Broker-4
Broker-5
Broker-6
分布式协调服务:Zookeeper
ZK-1
ZK-2
ZK-4
ZK-5
ZK-3
上报数据
上报数据
服务端数据上报
采集网关:Collect
HttpCollect
LVS
FileCollect
Client

技术架构--数据采集接入

	scribe	Chukwa	Kafka	Flume
公司	facebook	apache/yahoo	LinkedIn	Cloudera
开源时间	2008 年 10 月	2009 年 11 月	2010 年 12 月	2009 年 7 月
实现语言	C/C++	JAVA	SCALA	JAVA
框架	push/push	push/push	push/pull	push/push
容错性	collector 和 store 之间有容错机制，而 agent 和 collector 之间的容错需用户自己实现	Agent 定期记录已送给 collector 的数据偏移量，一旦出现故障后，可根据偏移量继续发送数据。	Agent 可用通过 collector 自动识别机制获取可用 collector。store 自己保存已经获取数据的偏移量，一旦 collector 出现故障，可根据偏移量继续获取数据。	Agent 和 collector，collector 和 store 之间均有容错机制，且提供了三种级别的可靠性保证。
负载均衡	无	无	使用 zookeeper	使用 zookeeper
可扩展性	好	好	好	好
agent	Thrift client，需自己实现	自带一些 agent，如获取 hadoop logs 的 agent	用户需根据 Kafka 提供的 low-level 和 high-level API 自己实现。	提供了各种非常丰富的 agent
collector	实际上是一个 thrift server	--	使用了 sendfile，zero-copy 等技术提高性能	系统提供了很多 collector，直接可以使用。
store	直接支持 HDFS	直接支持 HDFS	直接支持 HDFS	直接支持 HDFS
总体评价	设计简单，易于使用，但容错和负载均衡方面不够好，且资料较少。	属于 hadoop 系列产品，直接支持 Hadoop，目前版本升级比较快，但还有待完善。	设计架构（push/pull）非常巧妙，适合异构集群，但产品较新，其稳定性有待验证。	非常优秀

技术架构--数据清洗



技术架构--数据清洗

Storm UI

Topology summary

Name	Id	Status	Uptime	Num workers	Num executors	Num tasks
etl-offline-mobile-action	etl-offline-mobile-action-236-1449748276	ACTIVE	3d 22h 10m 15s	25	340	340

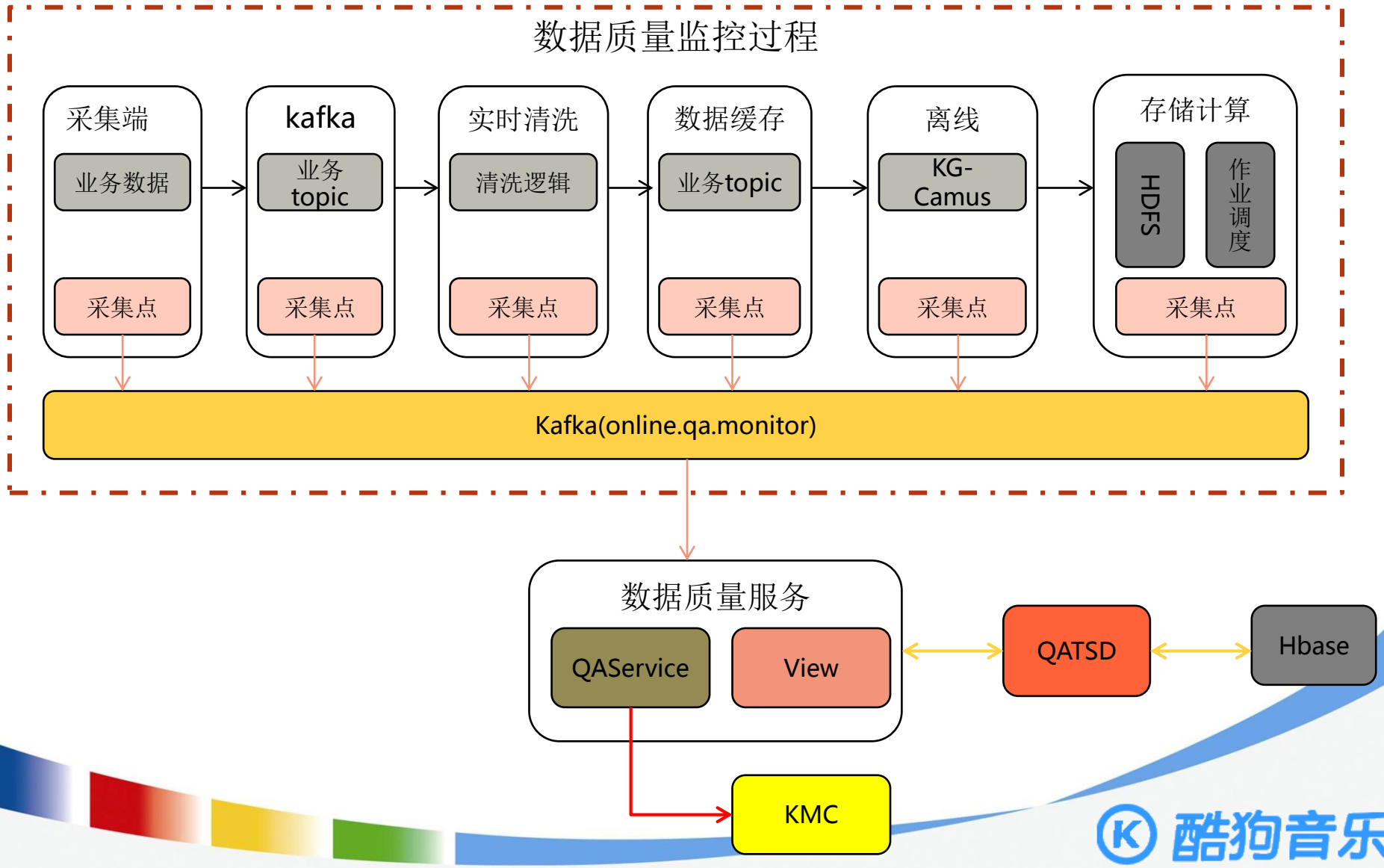
Topology actions

Activate | Deactivate | Rebalance | Kill

Topology stats

Window	Emitted	Transferred	Complete latency (ms)	Acked	Failed
10m 0s	45331840	45331840	17.336	21029480	0
3h 0m 0s	644609940	644609940	18.279	300807860	0
1d 0h 0m 0s	4489975380	4489975380	17.299	2104960960	0
All time	20039011160	20039011160	21.607	9338696620	0

技术架构--数据质量监控



目录

踩过的坑

- 那些掉过的坑

踩过的坑--zookeeper

磁盘空间爆了

```
# The number of snapshots to retain in dataDir
autopurge.snapRetainCount=100
# Purge task interval in hours ,  Set to "0" to disable auto purge feature
autopurge.purgeInterval=2
```

```
#决定了每个IP地址可以发起的socket连接个最大个数
maxClientCnxns=800

#最大会话超时时间
maxSessionTimeout=120000
```

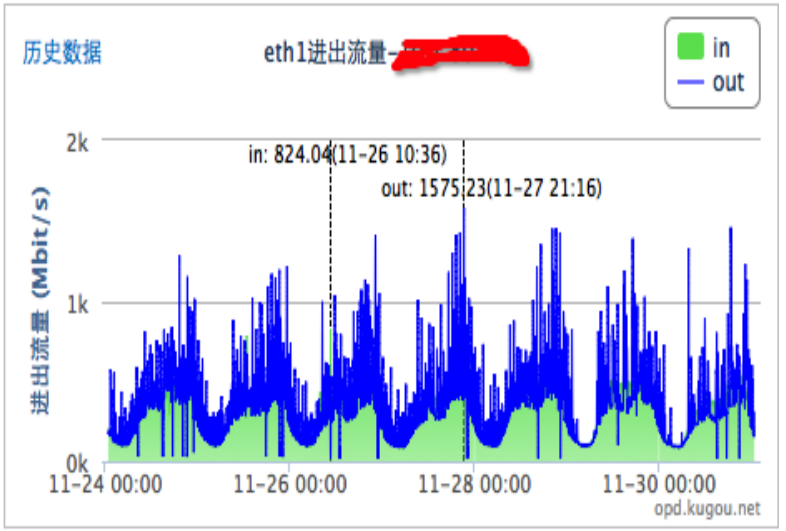
踩过的坑--Kafka 吞吐量

- # The number of threads handling network requests
- num.network.threads=25（默认10）

- #一些后台任务处理的线程数，例如过期消息文件的删除等，一般情况下不需要去做修改
- background.threads=12（默认：5）

- #等待IO线程处理的请求队列最大数，若是等待IO的请求超过这个数值，那么会停止接受外部消息，应该是一种自我保护机制。
- queued.max.requests=1000

- # The number of threads doing disk I/O
- num.io.threads=48 （最初默认：12）



踩过的坑--Kafka Log Flush Policy

- # The number of messages to accept before forcing a flush of data to disk
- log.flush.interval.messages=10000 (默认注释)

- # The maximum amount of time a message can sit in a log before we force a flush
- log.flush.interval.ms=1000 (默认注释)

踩过的坑--Kafka log segment &Replication

- # The maximum size of a log segment file. When this size is reached a new log segment will be created.
- log.segment.bytes=1073741824 （默认注释）

- # The interval at which log segments are checked to see if they can be deleted according
- # to the retention policies
- log.retention.check.interval.ms=300000 （默认注释）

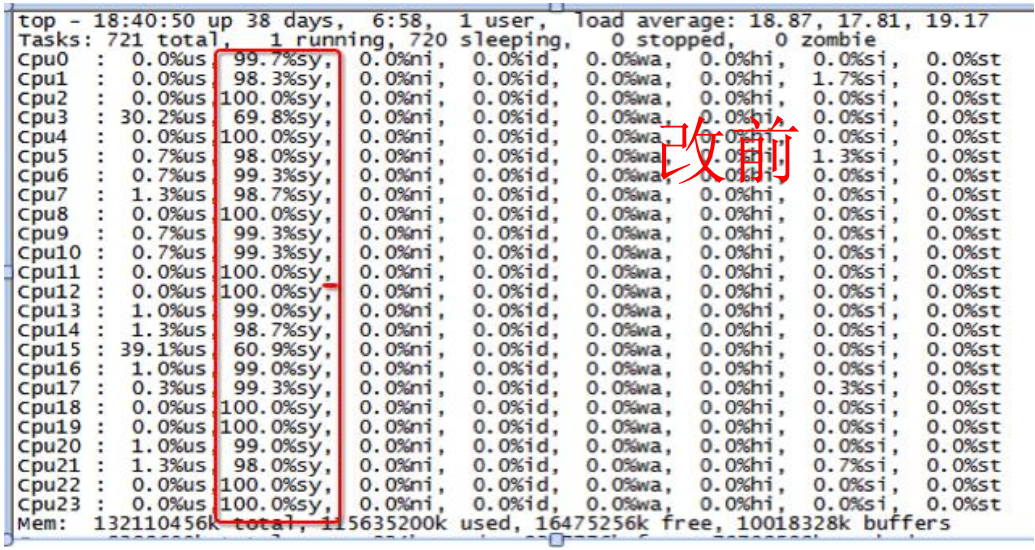
- ################Replication Configuration##############
- num.replica.fetchers=2

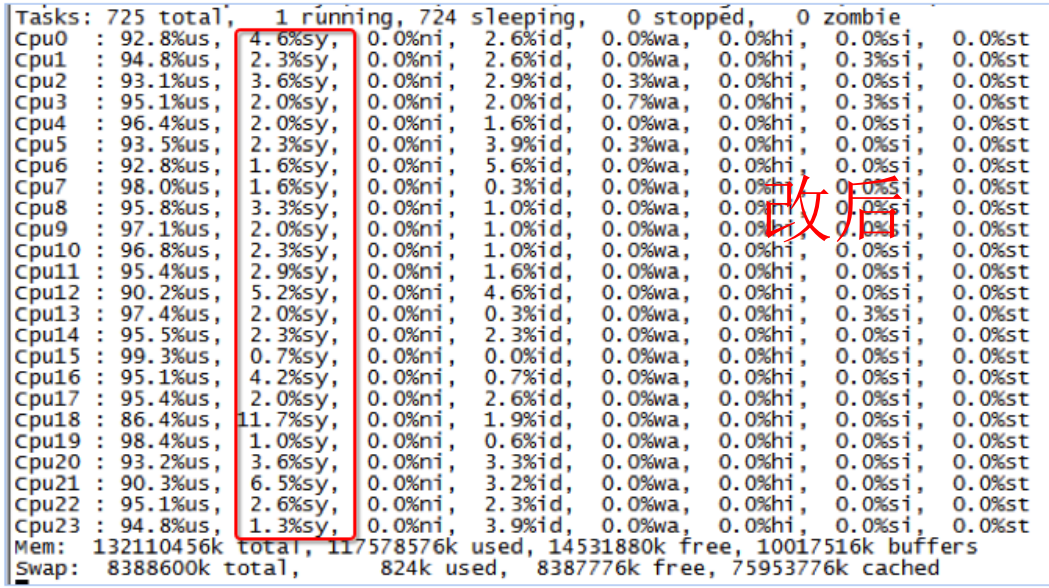
踩过的坑--Kafka Replication

- ################Replication Configuration##############
- num.replica.fetchers=2

踩过的坑--Hadoop2.7在CentOS6.3的内核CPU过高

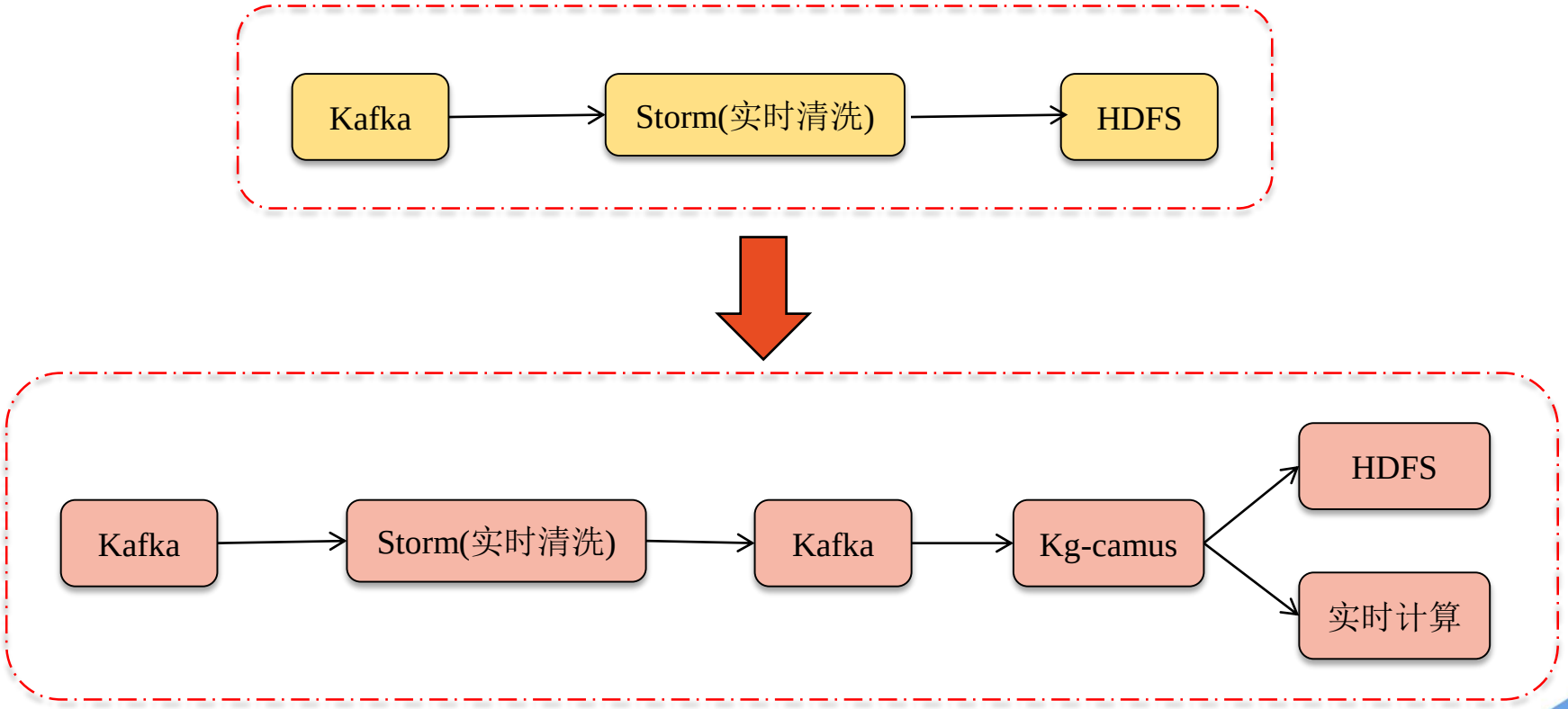
- 经过多种方法的测试验证，发现CentOS6优化了内存申请的效率，引入了THP的特性，而Hadoop是高密集型内存运算系统，这个改动给hadoop带来了副作用。通过以下内核参数优化关闭系统THP特性:
- echo never > /sys/kernel/mm/redhat_transparent_hugepage/enabled
- echo never > /sys/kernel/mm/redhat_transparent_hugepage/defrag





踩过的坑—Storm实时写入HDFS

- 每天7T的数据实时写入HDFS，经常导致HDFS客户端不稳定。



目录

后续规划

后续数据平台的持续改进点：

- 数据存储
  - ☑ 分布式内存文件系统(Tachyon)
  - ☑ 数据分层存储
  - ☑ 数据列式存储
- 即席查询(OLAP)
- 资源隔离
- 数据安全

Q & A

欢迎加入“酷狗音乐”

邮件：wangjin@kugou.net