打造万亿级别的数据流水线

Steven Wu @ Netflix

个人简介

@Netflix Real-Time Data Infrastructure Team

@Netflix Cloud Platform Infrastructure Team

@Yahoo! Messenger Server Team

工作领域：大型分布式系统

大纲

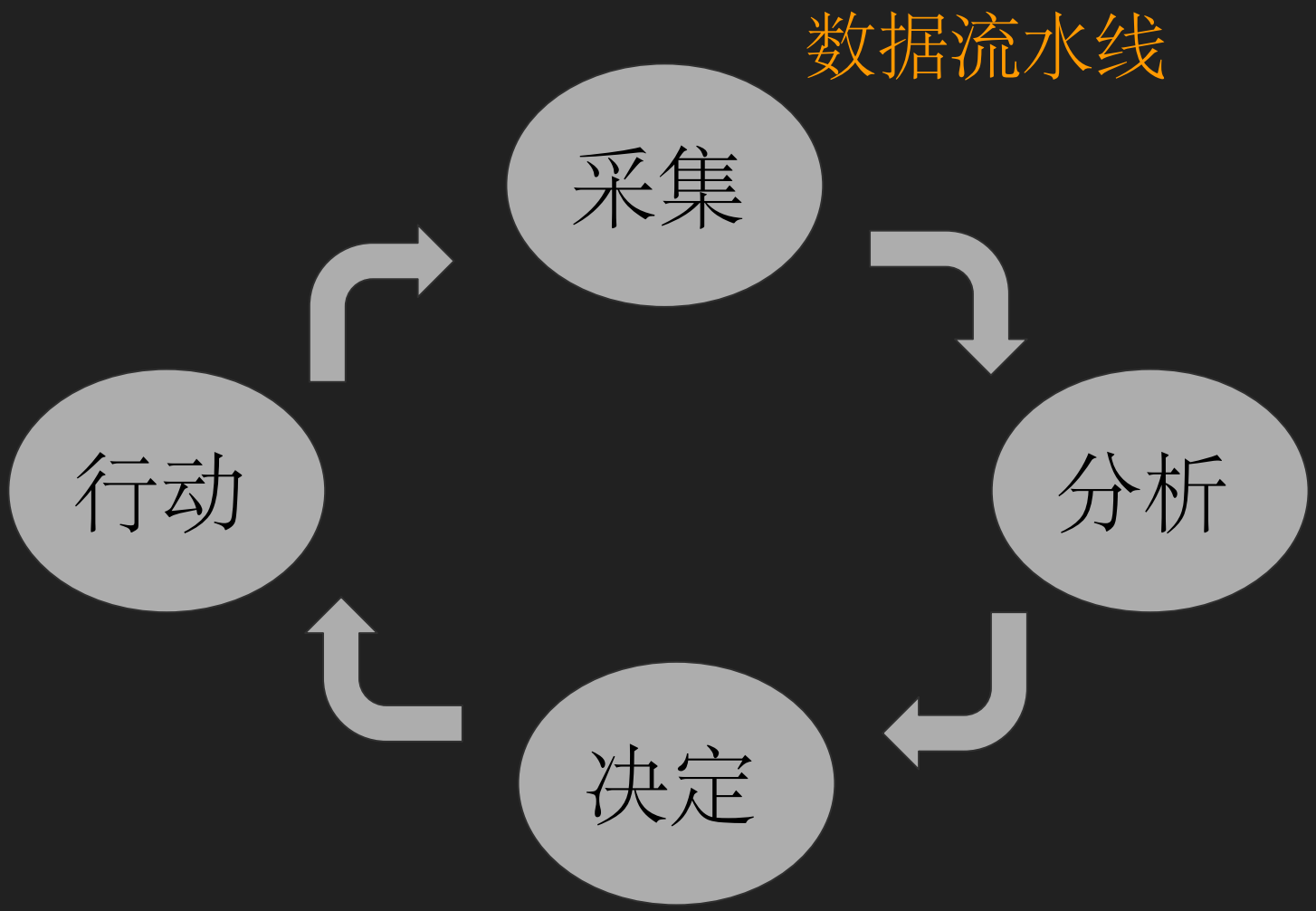
- Netflix数据流水线简介
- 演化之路
- Keystone架构
- 实战中的问题和教训
- 总结与展望

Netflix数据流水线简介

Netflix简介

- 按月付费的网络视频服务（无广告）
- >190个国家和地区
- >8千万用户
- >35%的北美Internet峰值下行流量
- 100%运行在AWS云

数据驱动的文化



Netflix是一家数据记录(data logging)公司

偶尔播放网络视屏

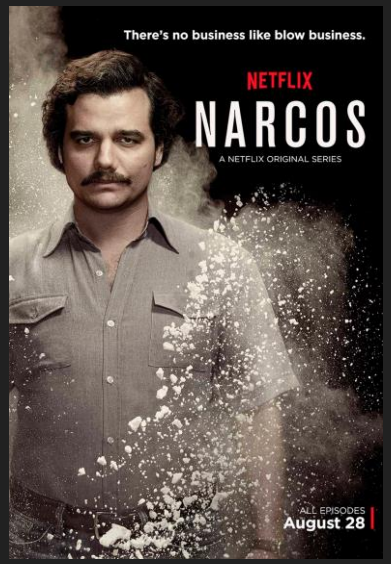

NETFLIX
HOUSE of CARDS


NETFLIX
ORANGE
is the new BLACK


NETFLIX
MARCO POLO


NETFLIX
A NETFLIX DOCUMENTARY SERIES
MAKING A MURDERER
ONLY ON NETFLIX
Dec 18


NETFLIX
A NETFLIX ORIGINAL SERIES
MARVEL
DAREDEVIL
ALL EPISODES
April 10


There's no business like blow business.
NETFLIX
NARCOS
A NETFLIX ORIGINAL SERIES
ALL EPISODES
August 28


NETFLIX
MARVEL
JESSICA JONES
Nov 20


NETFLIX
STRANGER THINGS
A NETFLIX ORIGINAL SERIES
ALL EPISODES
July 15

数据

- 数值监控指标(numeric metrics)
  - counter, gauge, meter, timer, histogram
- 结构化数据 (structural event)

用户：张三
视屏：纸牌屋
类型：电视剧
季数：1
集数：12
行为：开始播放

两套数据系统

- 数值监控指标
  - Atlas （Netflix开源代码）
- 结构化数据
  - Keystone数据流水线（今天的主题）

数据流水线当前规模

- 每日
  - 7千亿条消息（新年时曾经达到万亿）
  - 1.3 PB
- 峰值每秒
  - 1.6千万条消息
  - 24 GB

例子

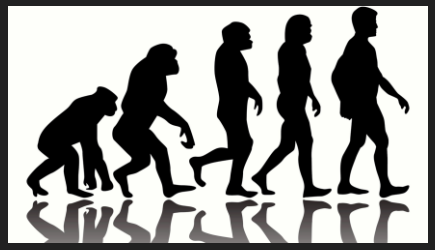
- 个性化推荐系统
- A/B测试
- 安全入侵检测
- 系统失败检测
- 分布式跟踪系统

数据收集和处理的开销

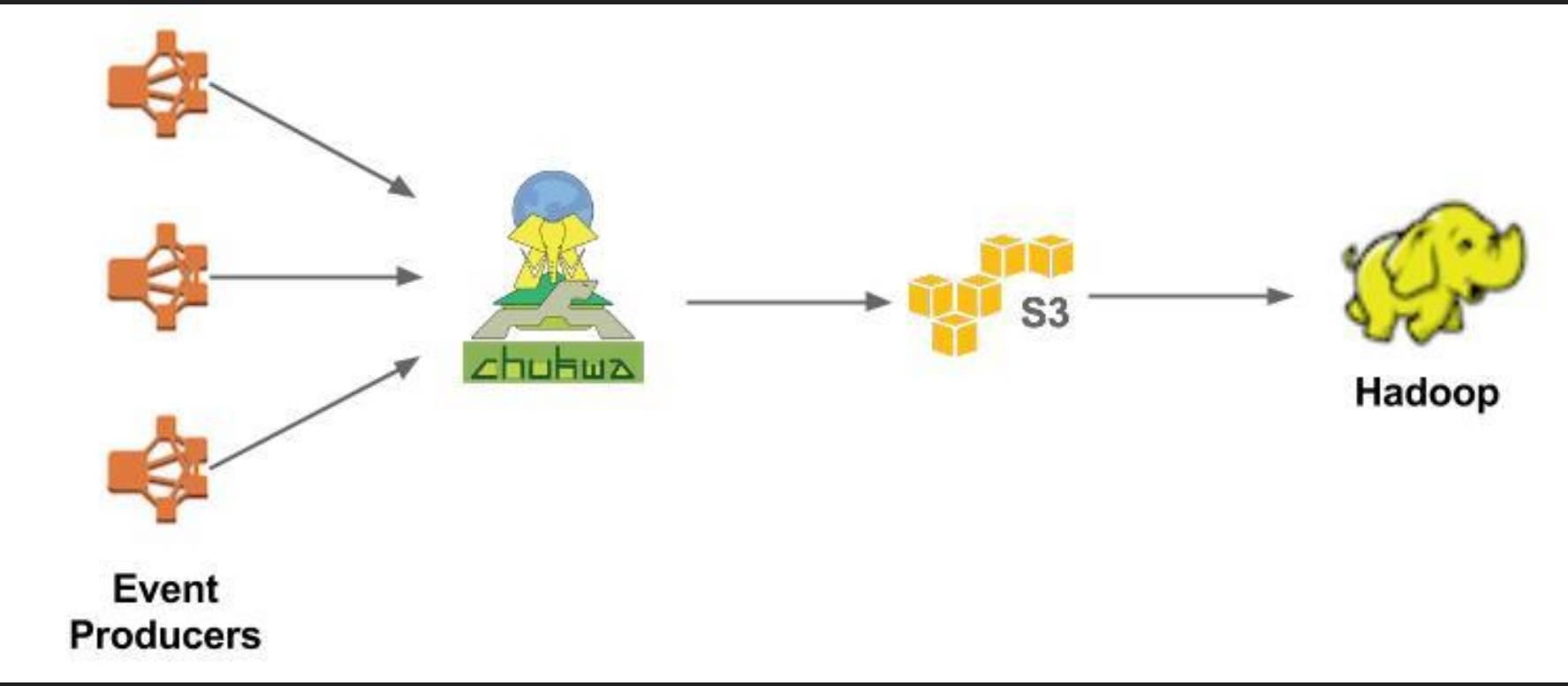
- ～20%的AWS费用［1］
  - 经过几年的不断优化后
- 曾经最高时～33%

［1］Keystone数据流水线, Hadoop, ElasticSearch, Atlas。不包含Cassandra

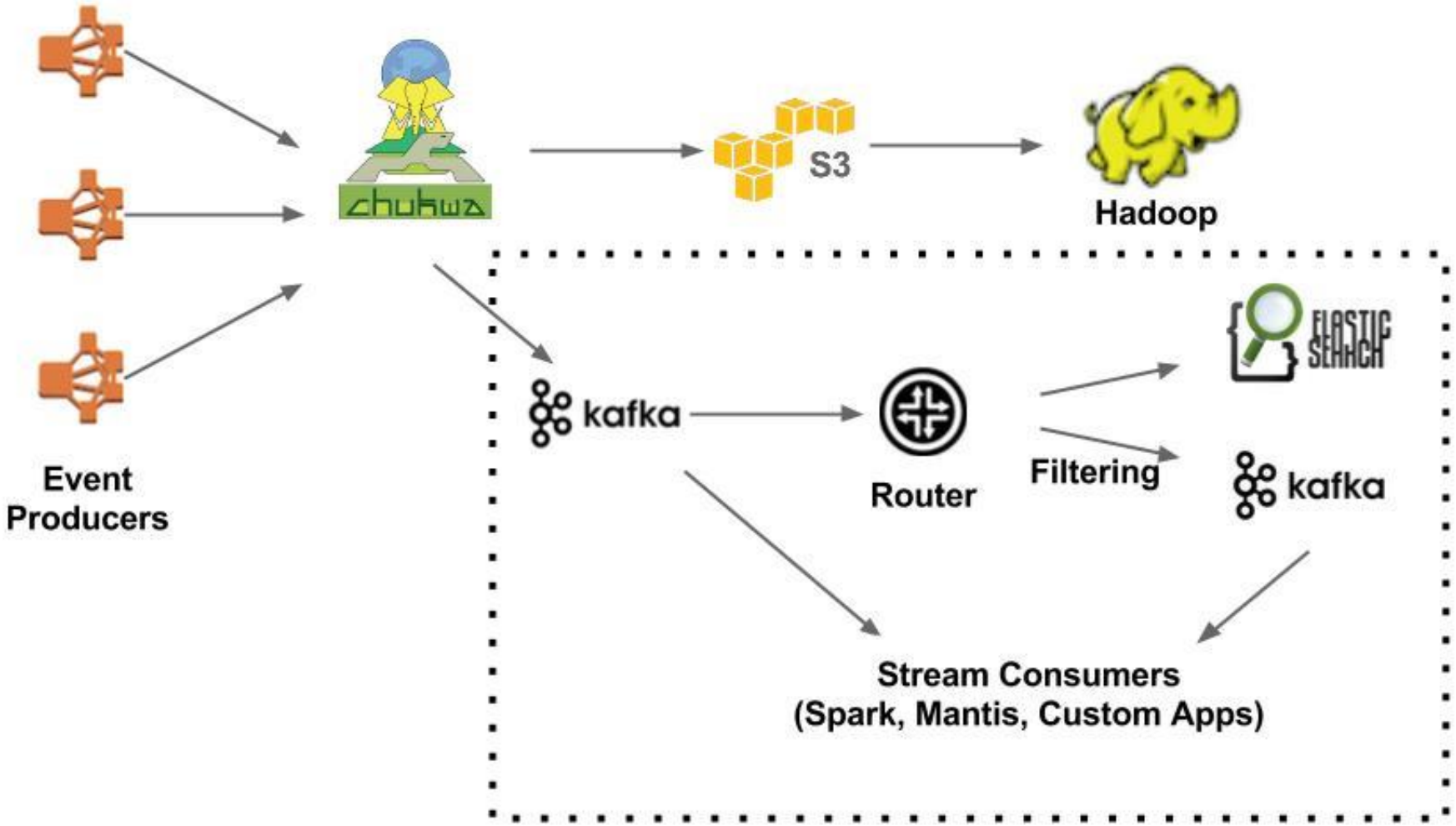
演化之路



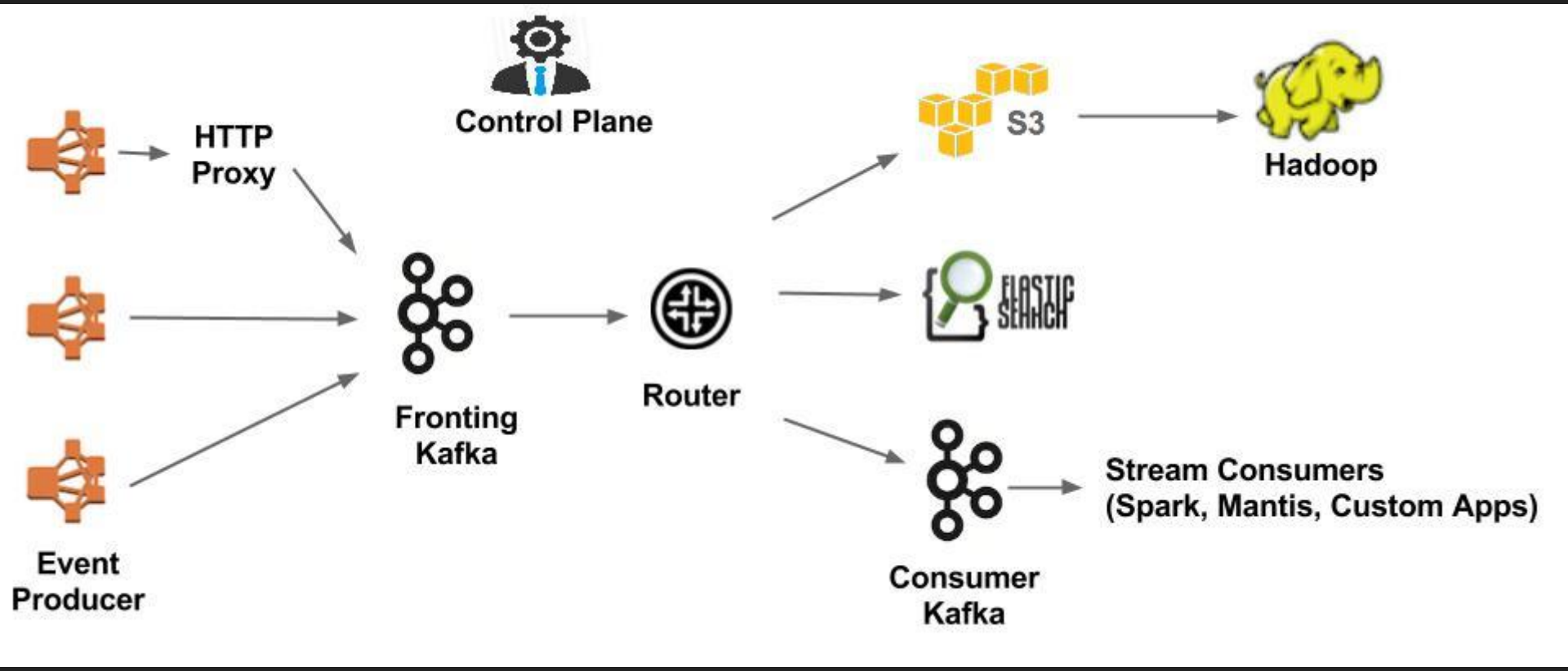
V1.0 Chukwa



V1.5 Chukwa + 实时分支



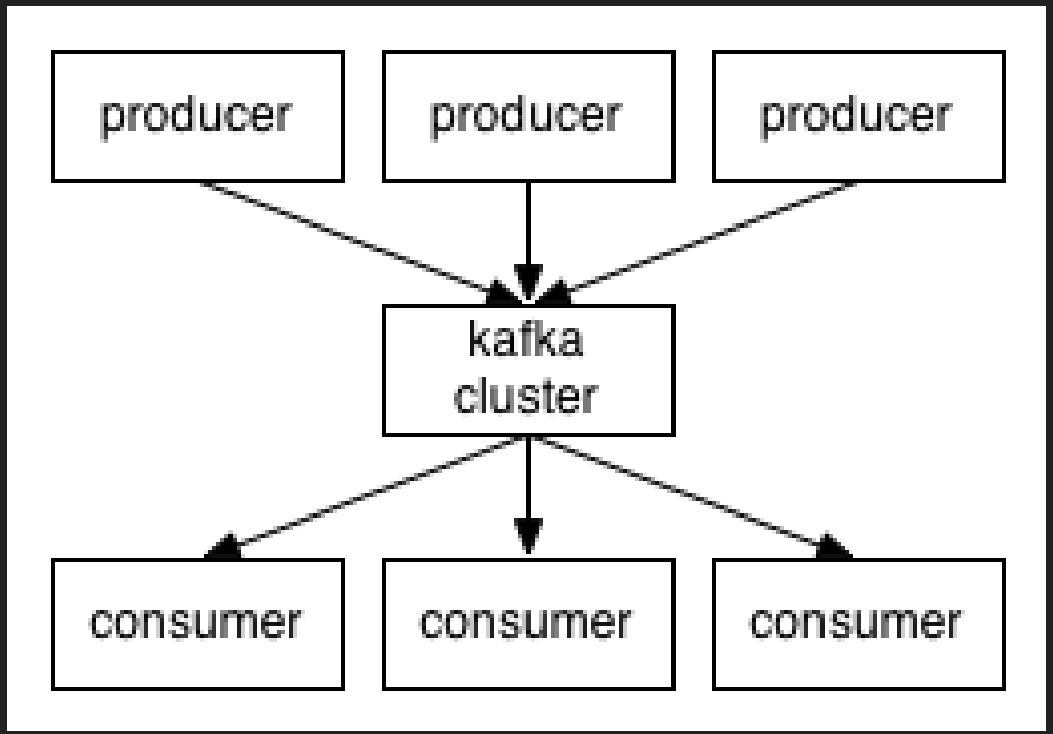
V2.0 Keystone



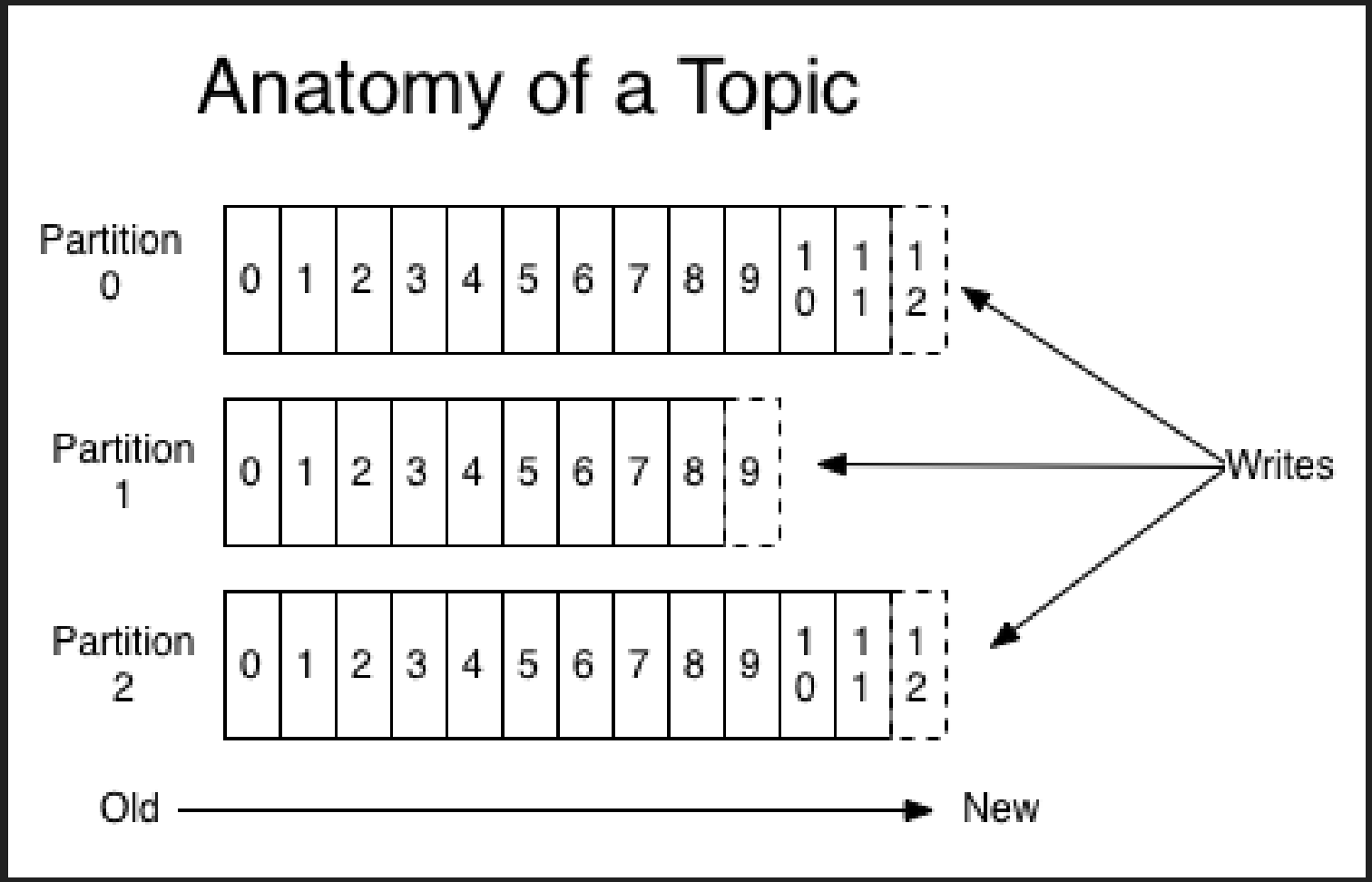
Keystone架构

Kafka简介

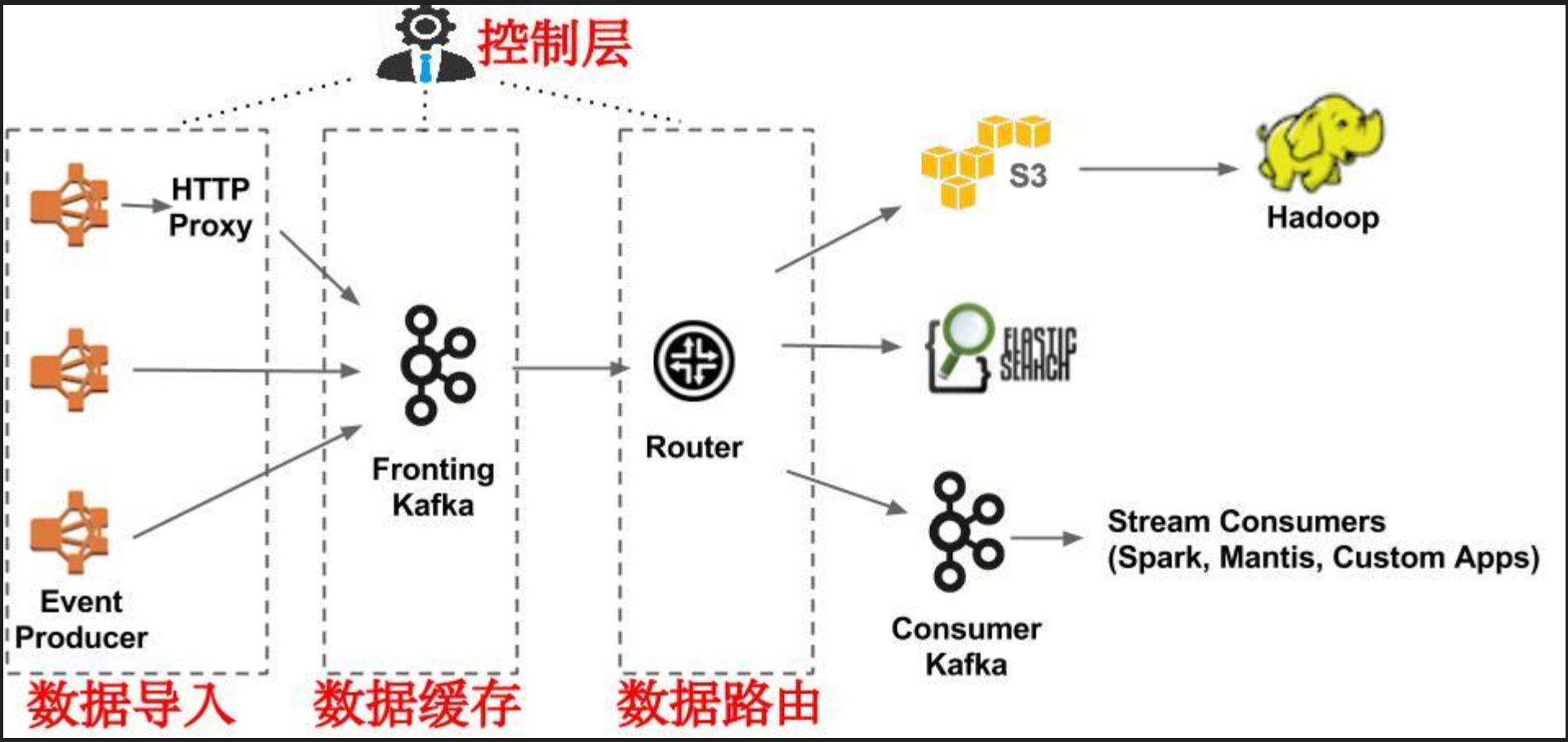
- 分布式消息队列系统
- 生产者
- 消费者
- 代理集群
- 推拉模型



话题(Topic)和分区(Partition)



架构


控制层
HTTP
Proxy
Event
Producer
Fronting
Kafka
Router
S3
Hadoop
ELASTIC
SEARCH
Consumer
Kafka
Stream Consumers
(Spark, Mantis, Custom Apps)
数据导入
数据缓存
数据路由

数据导入

- Netflix生产者Java库
  - 包装Apache Kafka生产者Java库
  - 与Netflix生态系统整合
- HTTP代理
  - 支持非Java语言应用

最大努力发送，不破坏应用

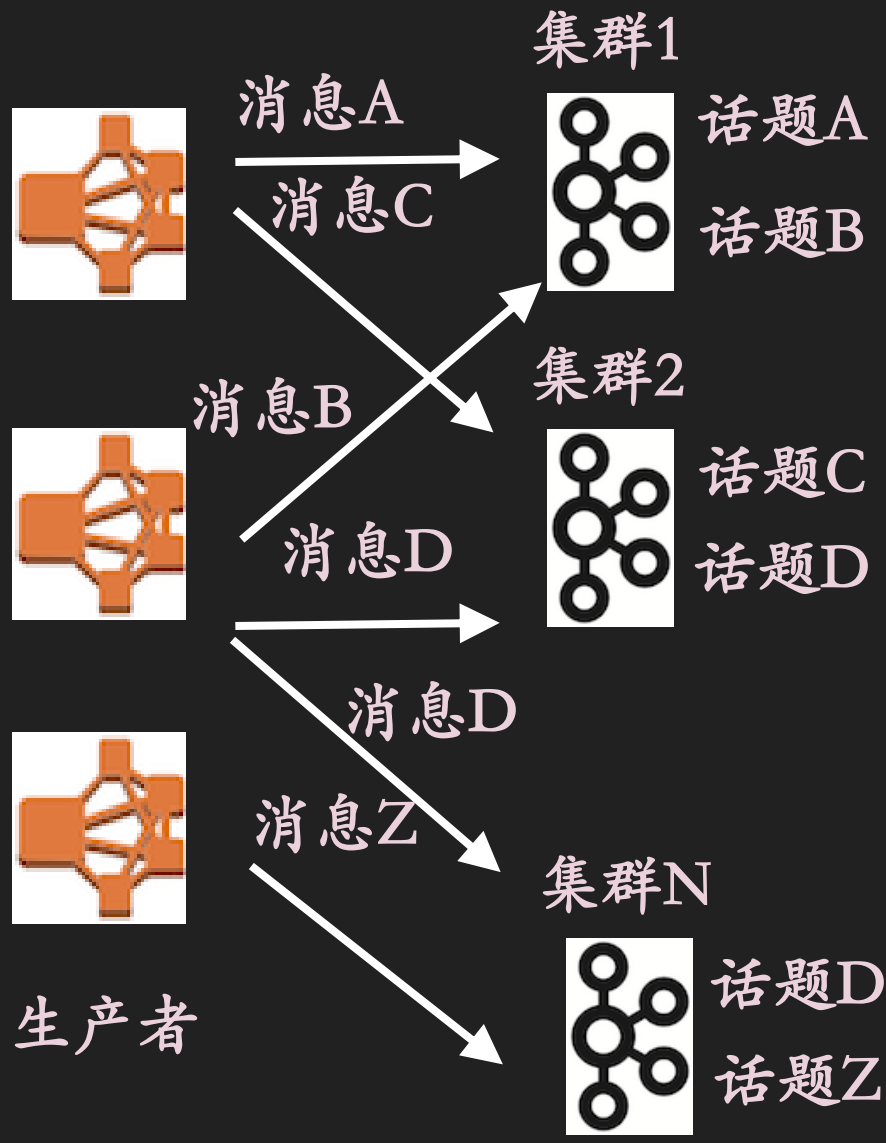
- 限制内存开销（32-512 MB）
- 不阻塞用户线程
  - block.on.buffer.full=false（丢弃）
  - 减少或者消除线程锁竞争
  - 处理第一次发送等待元数据阻塞线程的情况
- kafka代理故障不影响生产者应用

Kafka代理故障

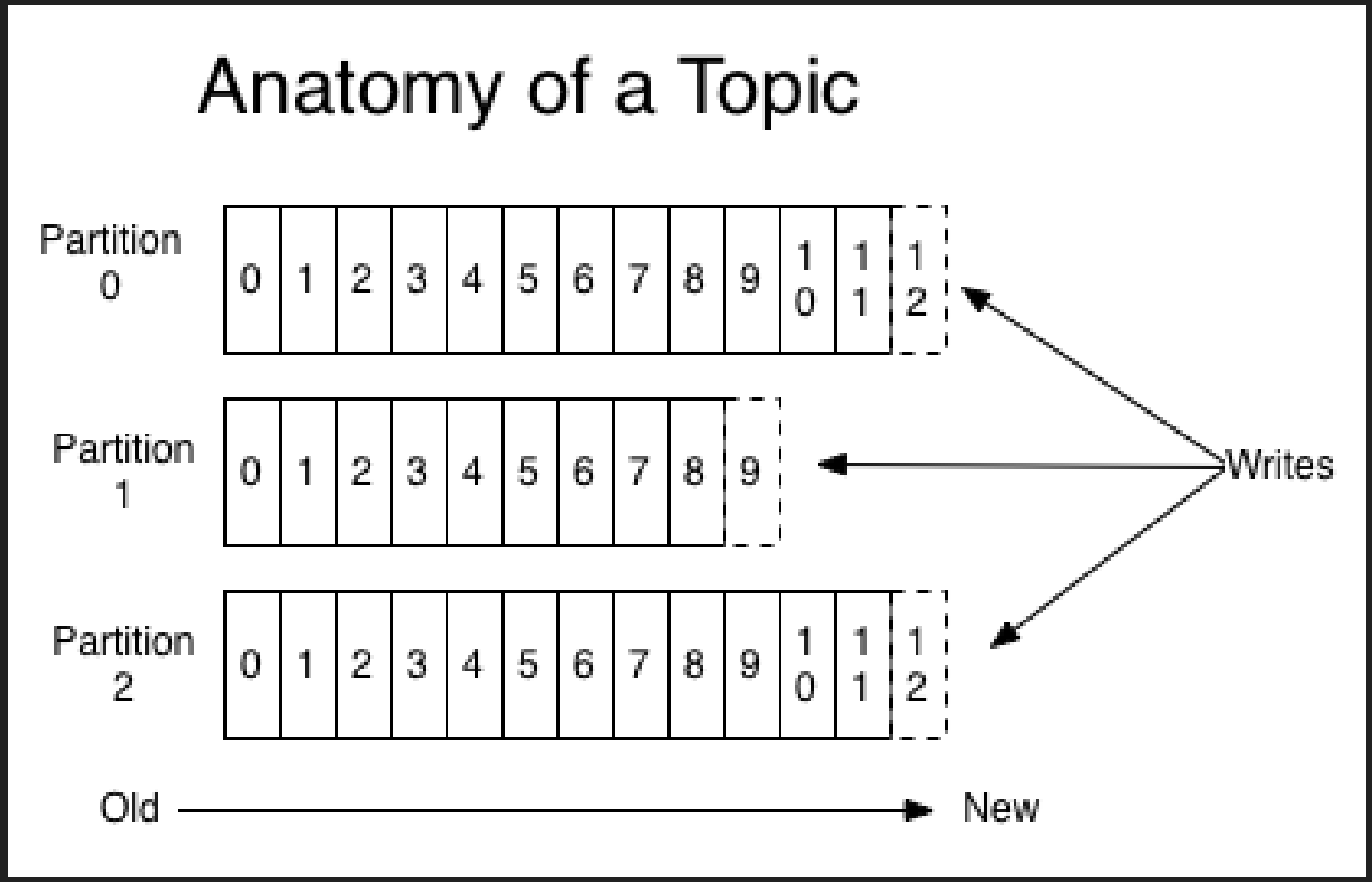
- 丢弃消息
- 运行中的生产者可继续服务
- 尽管Kafka模块初始化失败，新的生产者可以启动并可正常服务
- 自动恢复发送（当代理恢复时）
  - 定期检测Kafka模块状态并重新初始化

动态路由

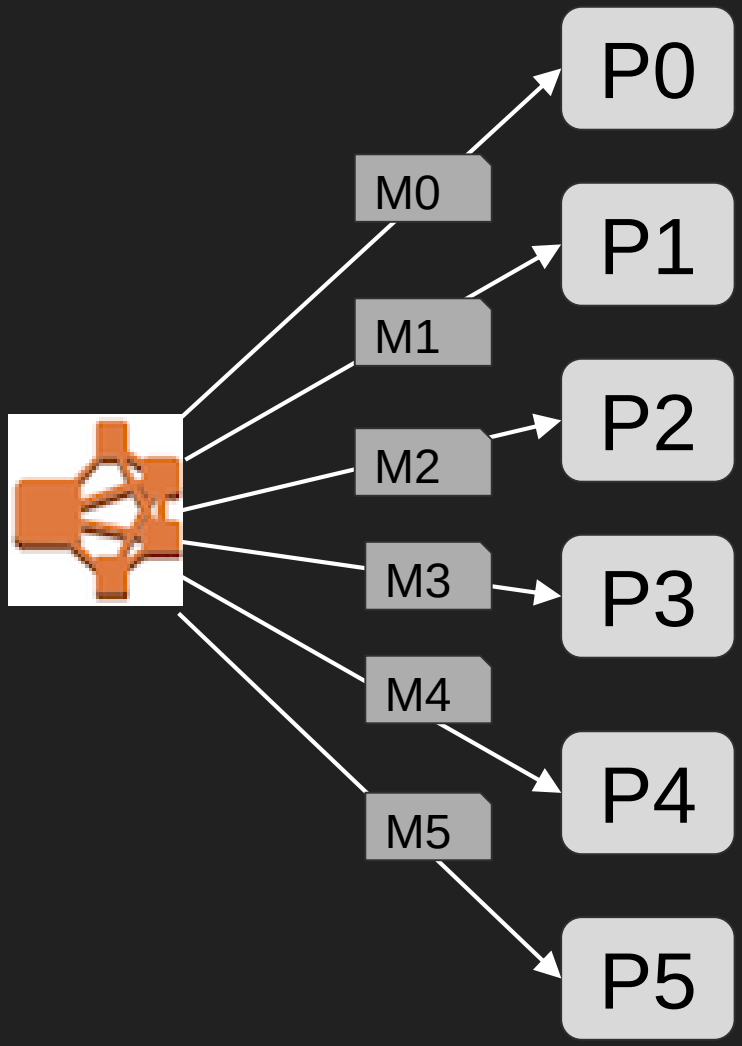
- 多个Kafka集群
- 路由表描述那个集群有哪个话题
- 路由表可动态更改。无需重启生产者服务（Archaius）



分区选择（Partitioner）

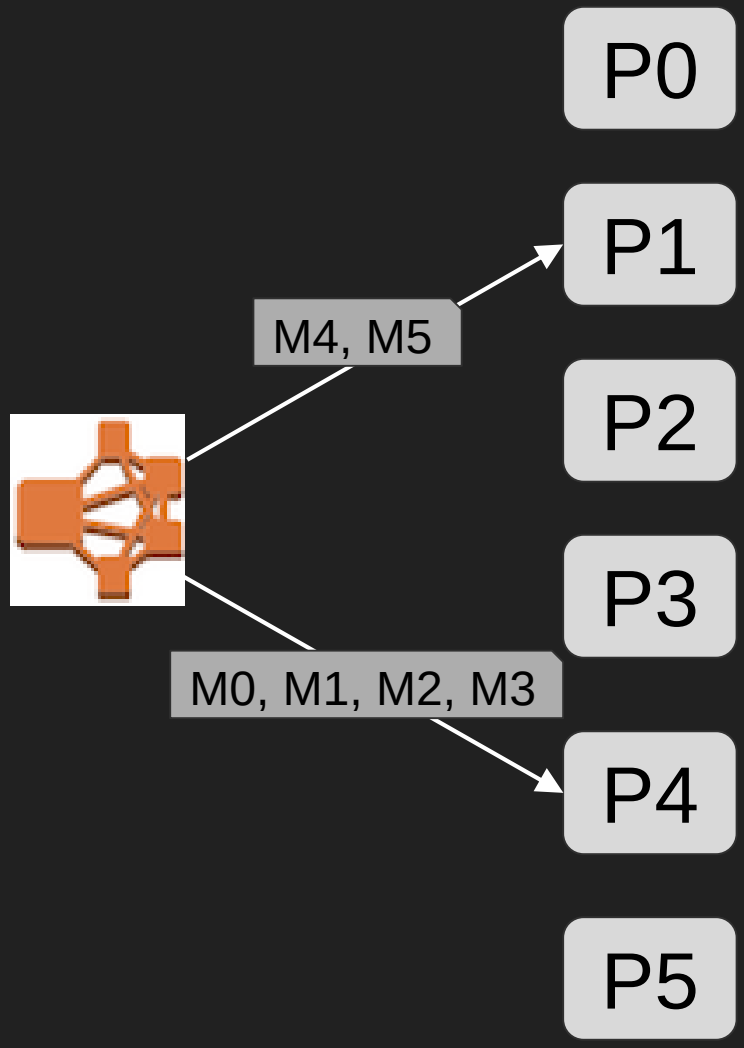


Round-Robin分区选择（缺省）



- 优点：简单
- 缺点：批处理效果不好（即使有linger.ms）

Sticky分区选择



- 优点：与linger.ms搭配的批处理效果非常好
- 注意：sticky时间不宜过长

批处理可以减少代理的开销

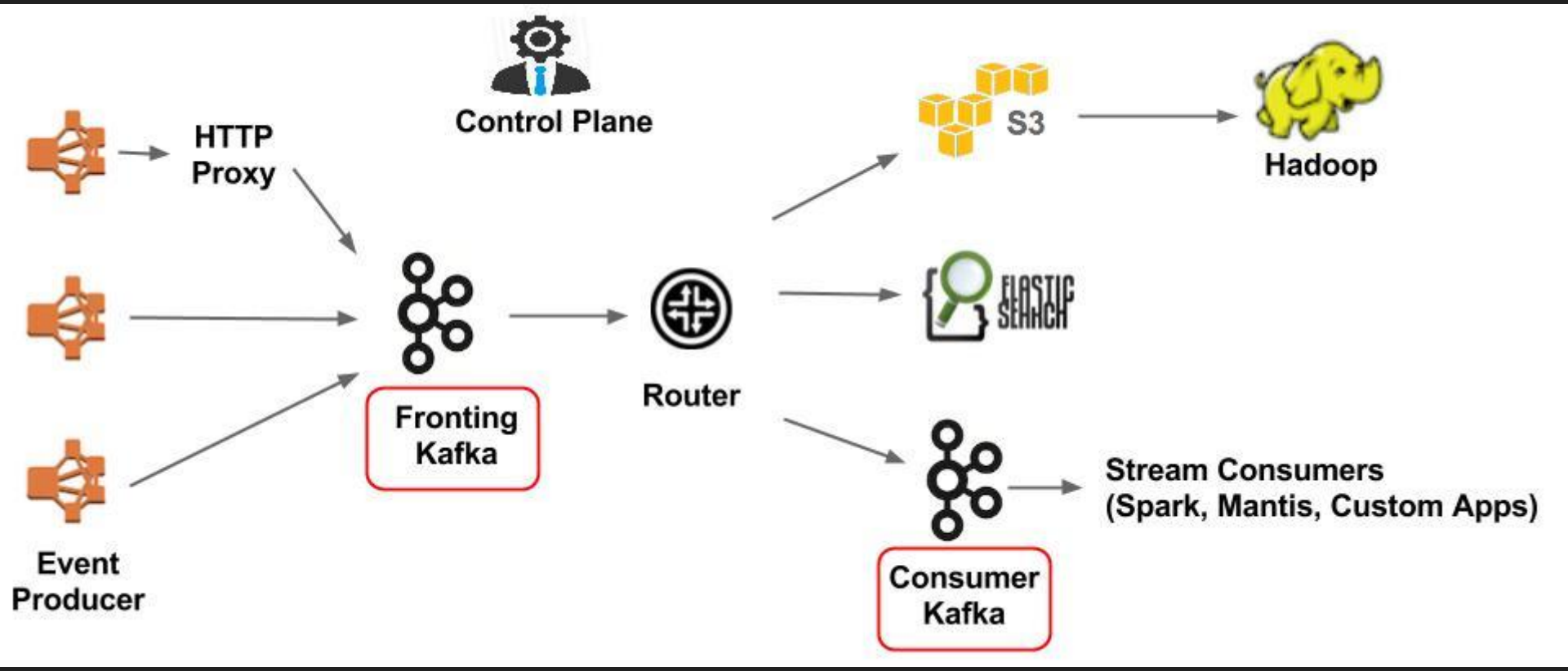
partitioner	batched records per request	broker cpu util [1]
round-robin with or without lingering	1.25	75%
sticky without lingering	2.0	50%
sticky with 100ms lingering	15 (full batch)	33%

[1] 10 MB & 10K msgs / second per broker, 1KB per message

数据缓存（Kafka）

- >4,000个Kafka代理
- 版本0.9

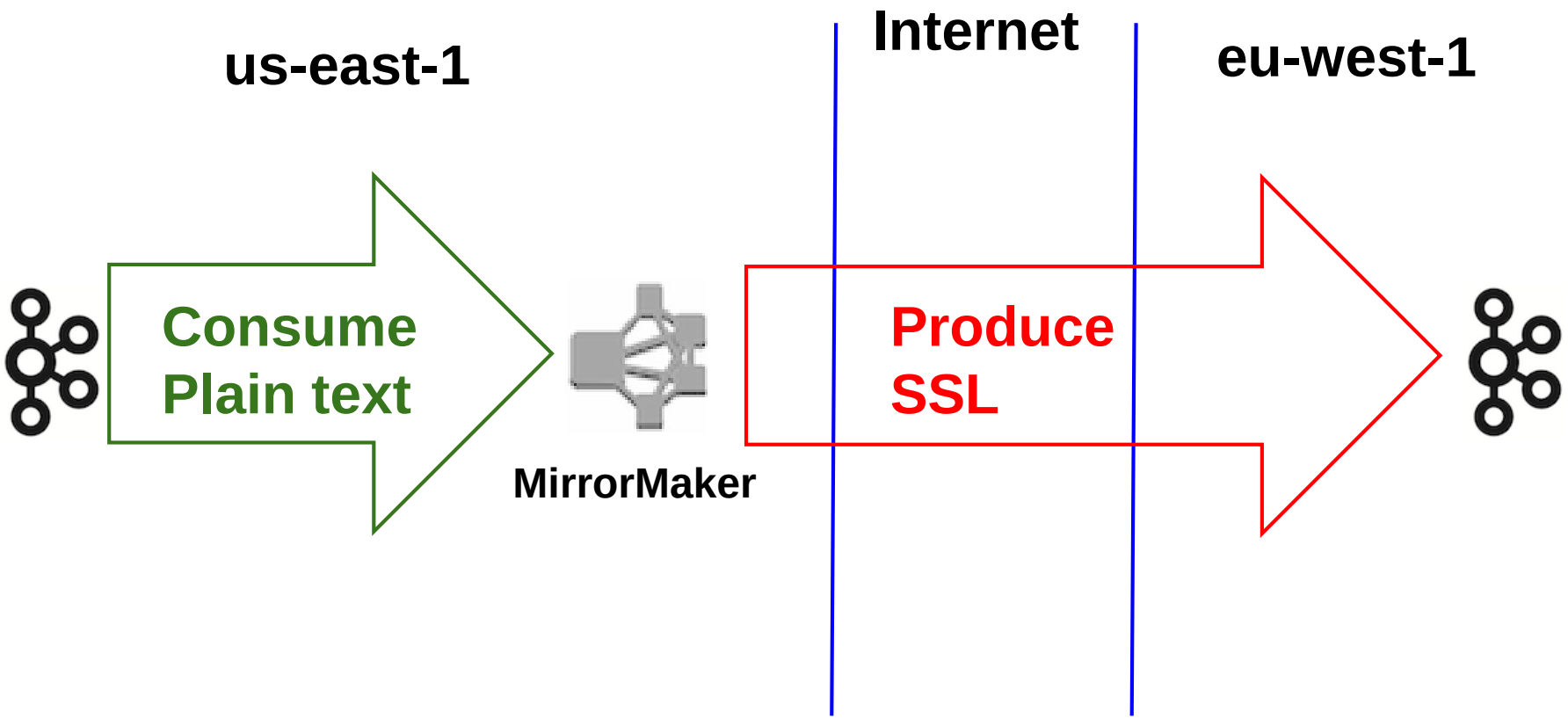
前置和消费者Kafka集群



部署策略

- 限制集群大小（<200个代理）
  - 限制集群的分区总数 <10,000
  - 集群不跨越AWS区域
  - 每个集群三个ASG（每个可用区一个ASG）

跨AWS区域复制


us-east-1
Internet
eu-west-1
Consume
Plain text
MirrorMaker
Produce
SSL

均衡的分区分配

- 集群**N**个代理，那么每个话题分配**N** x **K**个分区
- 每个代理**K**个分区
- **K**根据话题数据流量选择，每个分区控制在100 KB - 1 MB之间

Zone-Aware话题分区分配

- 6分区，2份复制
- 每个分区的2份复制不在同一可用区

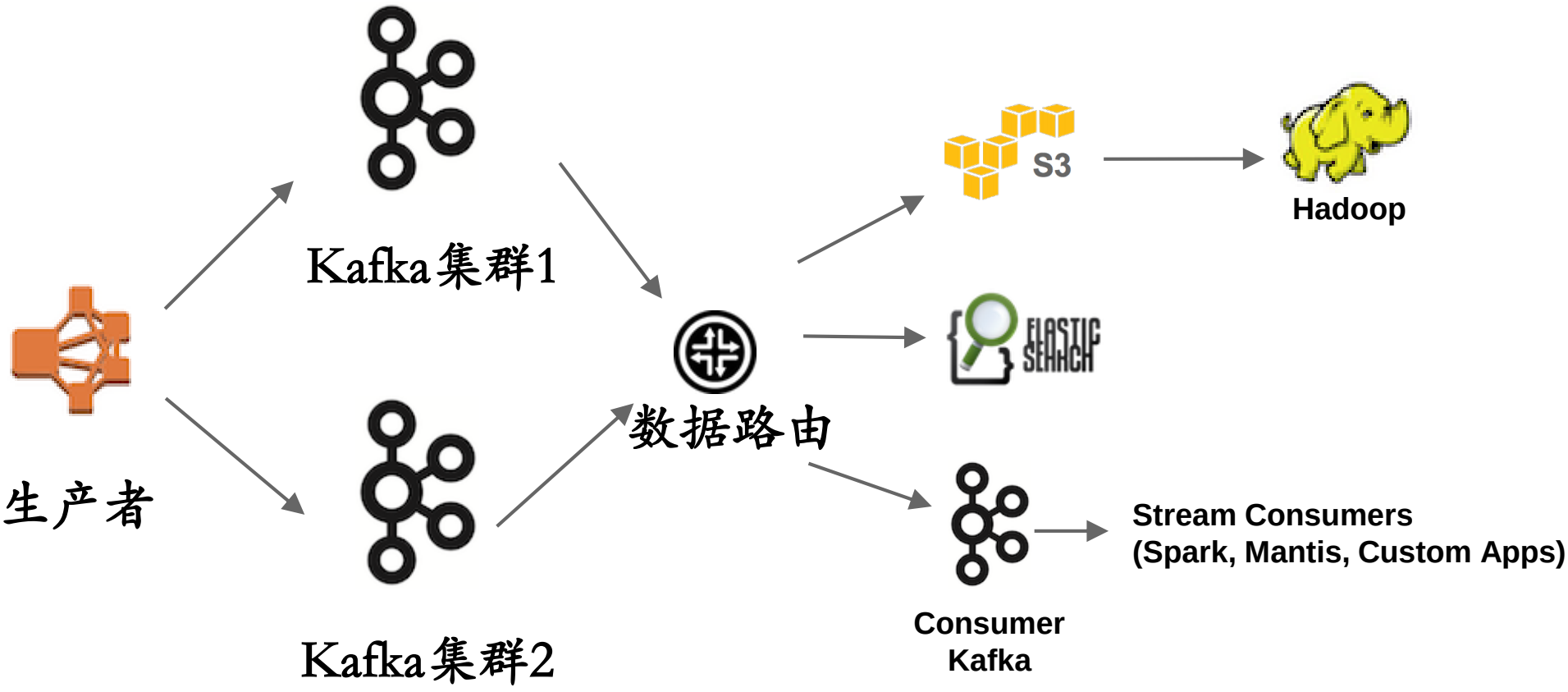
回馈开源代码

- KIP-36: Rack aware replica assignment
- Apache Kafka Github Pull Request #132
- Part of 0.10 release

扩展策略

- 容量规划考虑峰值流量和区域故障切换的流量
- 如果流量增长
  - 增加新集群
  - 扩展现有集群

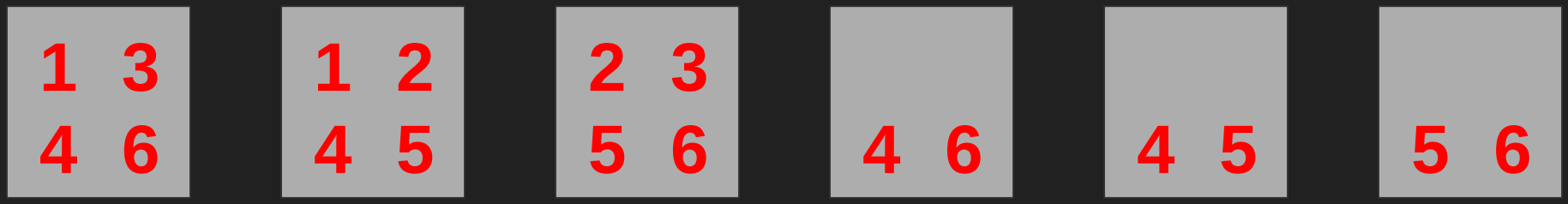
增加新集群：话题迁移


Kafka集群1
S3
Hadoop
数据路由
生产者
Stream Consumers
(Spark, Mantis, Custom Apps)
Consumer
Kafka
Kafka集群2

扩展现有集群

- 移动分区
- 增加分区

移动分区



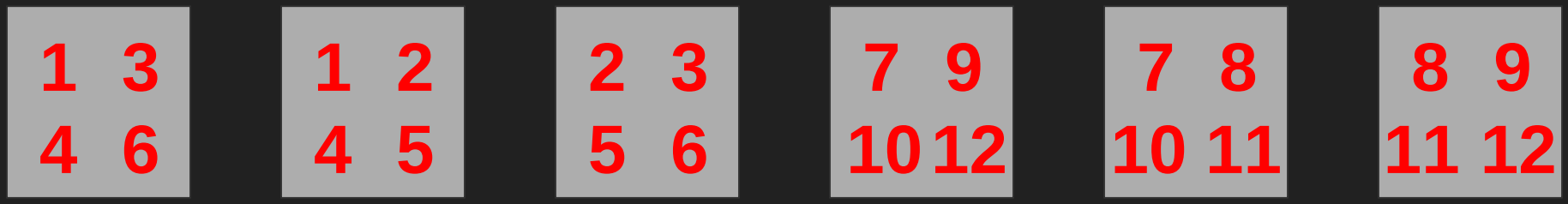
移动分区

- 优点： 对生产者和消费者透明
- 缺点
  - 非常慢（大规模数据移动）
  - 数据复制没有流量控制，可能会影响代理性能

移动分区策略

- 创建工具实行多次小批的分区移动以限制数据移动的流量

增加分区



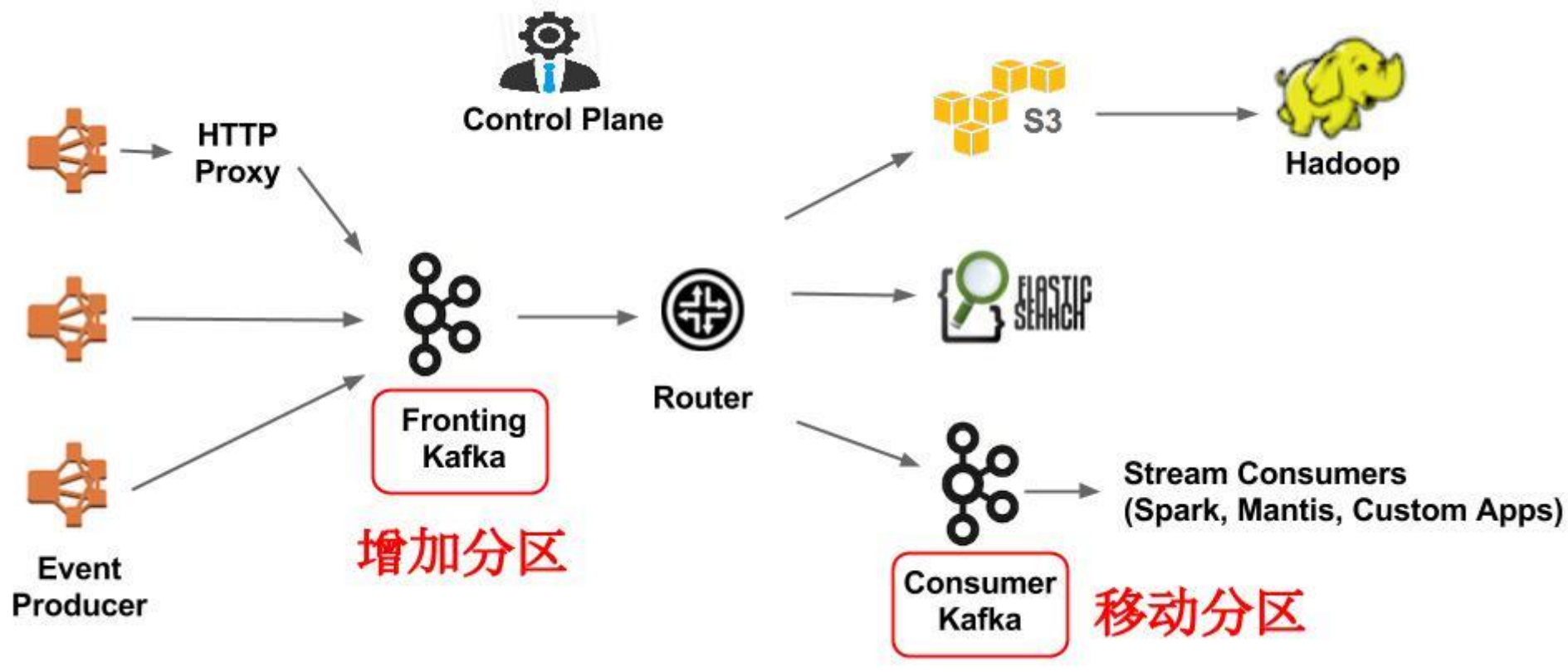
增加分区

- 优点：非常快（无数据移动）
- 缺点／前提
  - 没有keyed消息
  - 生产者和消费者可以应付分区增加

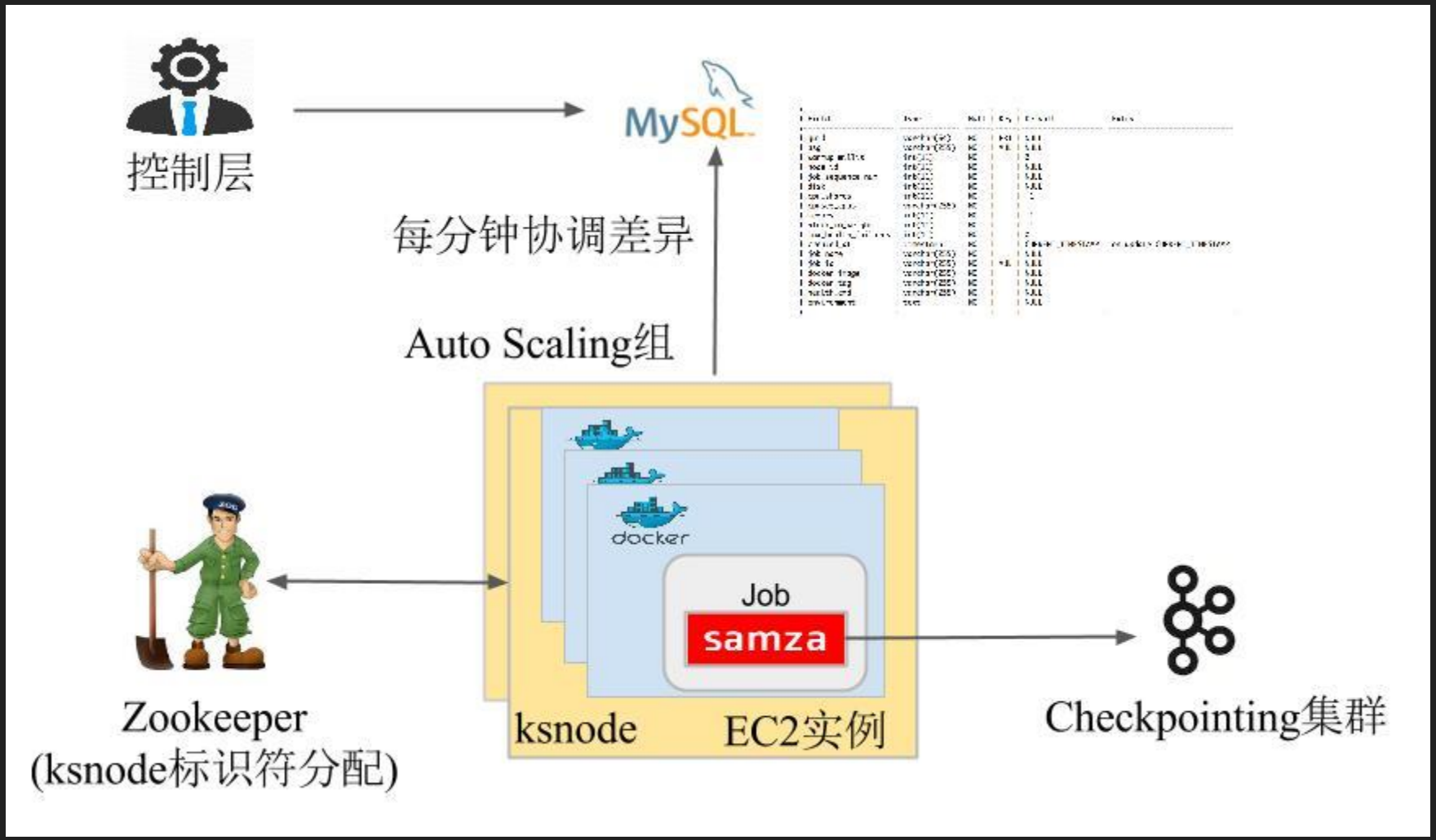
增加分区策略

- 创建工具保证新增加的分区只分配到新代理

应用场合


Control Plane
HTTP
Proxy
S3
Hadoop
Router
Fronting
Kafka
增加分区
Event
Producer
Stream Consumers
(Spark, Mantis, Custom Apps)
Consumer
Kafka
移动分区

数据路由


控制层
MySQL
每分钟协调差异
Auto Scaling组
docker
Job
samza
Zookeeper
(ksnode标识符分配)
ksnode
EC2实例
Checkpointing集群

数据路由

- >14,000 docker容器
- >1,400 c3.4xl实例

Samza v.s. Spark Streaming

- Spark Streaming不支持背压（backpressure）机制
- Spark Streaming性能不如Samza

注：2015年4月做的比较

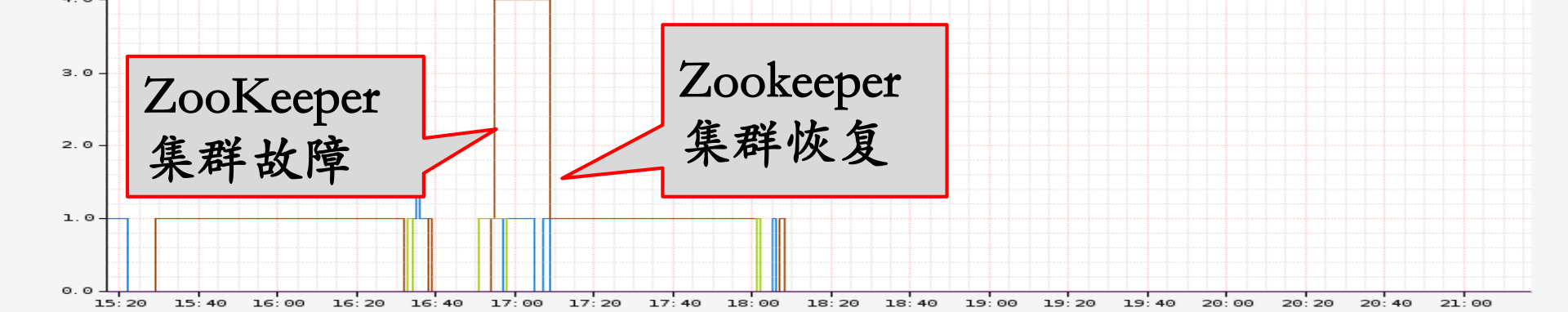
控制层

- 自助式服务用户界面
- 管理工具（如故障切换）
- 配置管理（如生产者动态路由表）
- 路由任务管理
- ...

实战中的问题和教训

一个真实的故事

Keystone上线的第二天



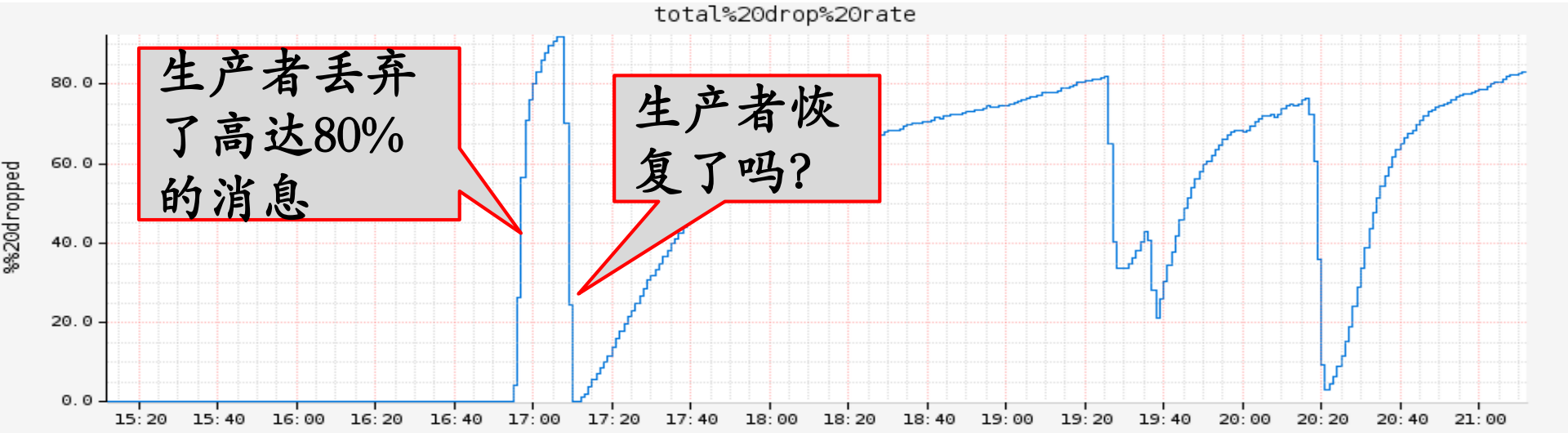
(name=DiscoveryStatus_NOT_REGISTERED)
Max : 2.000 Min : 0.000
Avg : 61.111m Last : 0.000
Tot : 22.000 Cnt : 360.000

(name=DiscoveryStatus_zookeeperserverdatapipeline_DOWN)
Max : 4.000 Min : 0.000
Avg : 483.333m Last : 0.000
Tot : 174.000 Cnt : 360.000

(name=DiscoveryStatus_zookeeperserverdatapipeline_OUT_OF_SERVICE)
Max : 1.000 Min : 0.000
Avg : 25.000m Last : 0.000
Tot : 9.000 Cnt : 360.000

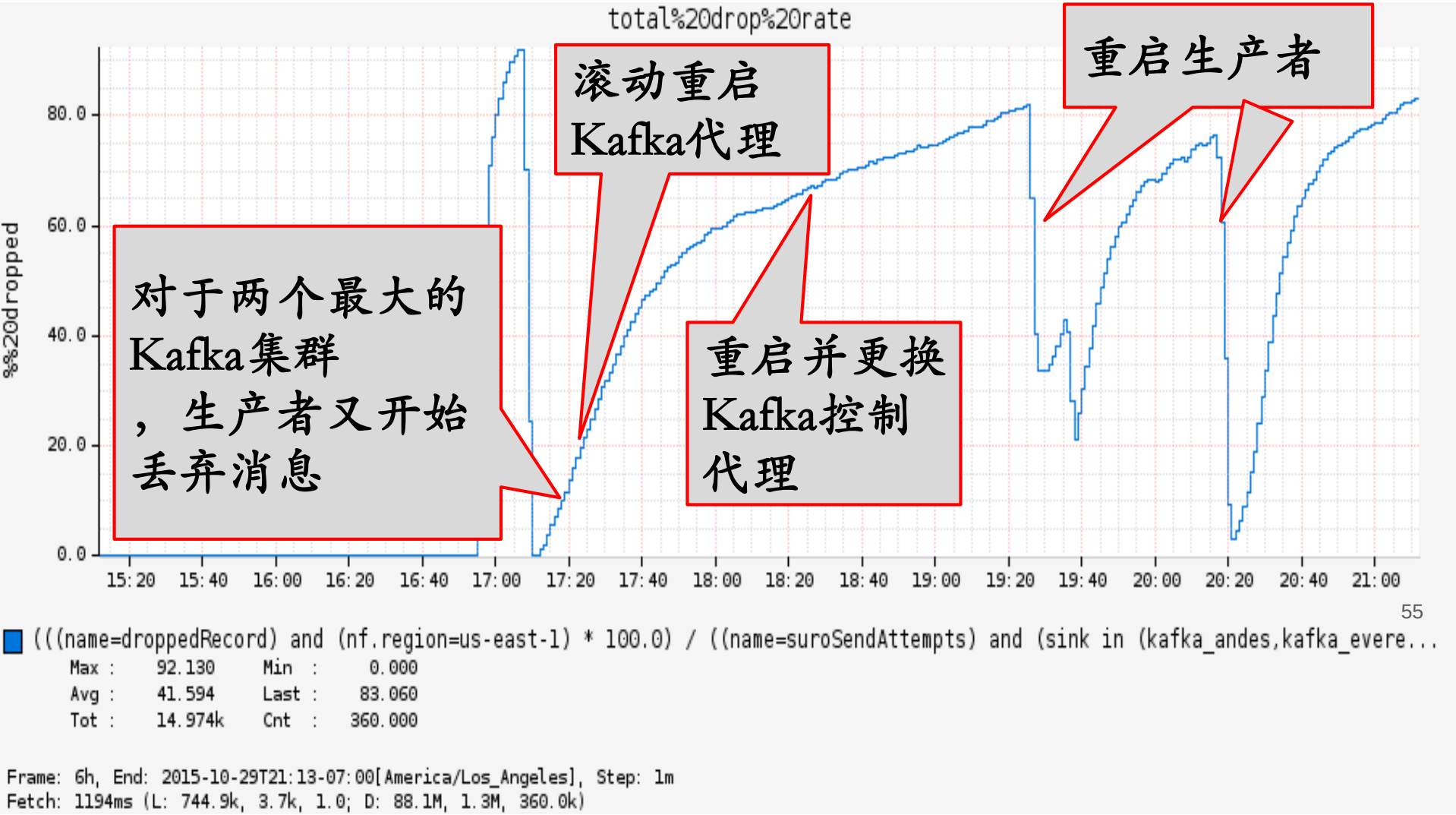
(name=DiscoveryStatus_zookeeperserverdatapipeline_UNKNOWN)
Max : 0.000 Min : 0.000
Avg : 0.000 Last : 0.000
Tot : 0.000 Cnt : 360.000

Frame: 6h, End: 2015-10-29T21:18-07:00[America/Los_Angeles], Step: 1m
Fetch: 491ms (L: 122.0, 65.0, 4.0; D: 13.6k, 23.5k, 1.4M)



(((name=droppedRecord) and (nf.region=us-east-1) * 100.0) / ((name=suroSendAttempts) and (sink in (kafka_andes,kafka_evere...
Max : 92.130 Min : 0.000
Avg : 41.594 Last : 83.060
Tot : 14.974k Cnt : 360.000

Frame: 6h, End: 2015-10-29T21:13-07:00[America/Los_Angeles], Step: 1m


total%20drop%20rate
%%20dropped
80.0
60.0
40.0
20.0
0.0
15:20
15:40
16:00
16:20
16:40
17:00
17:20
17:40
18:00
18:20
18:40
19:00
19:20
19:40
20:00
20:20
20:40
21:00
滚动重启Kafka代理
重启生产者
对于两个最大的Kafka集群，生产者又开始丢弃消息
重启并更换Kafka控制代理
55
(((name=droppedRecord) and (nf.region=us-east-1) * 100.0) / ((name=suroSendAttempts) and (sink in (kafka_andes,kafka_evere...
Max : 92.130 Min : 0.000
Avg : 41.594 Last : 83.060
Tot : 14.974k Cnt : 360.000
Frame: 6h, End: 2015-10-29T21:13-07:00[America/Los_Angeles], Step: 1m
Fetch: 1194ms (L: 744.9k, 3.7k, 1.0; D: 88.1M, 1.3M, 360.0k)

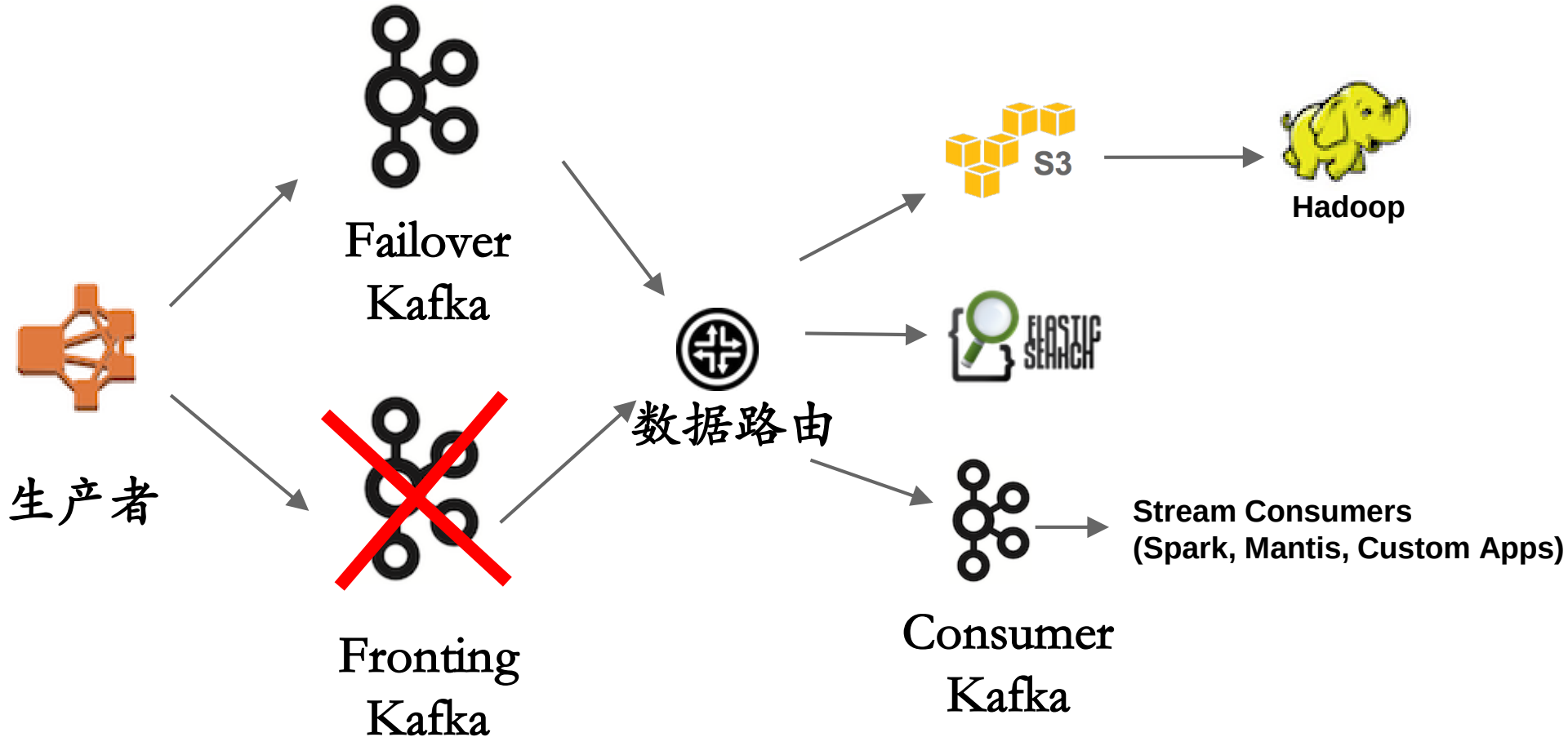
改变部署策略

- 每个前置Kafka集群有自己的Zookeeper
- 限制Kafka集群的大小（<200代理）
- 限制集群的总分区数（<10,000分区）

教训

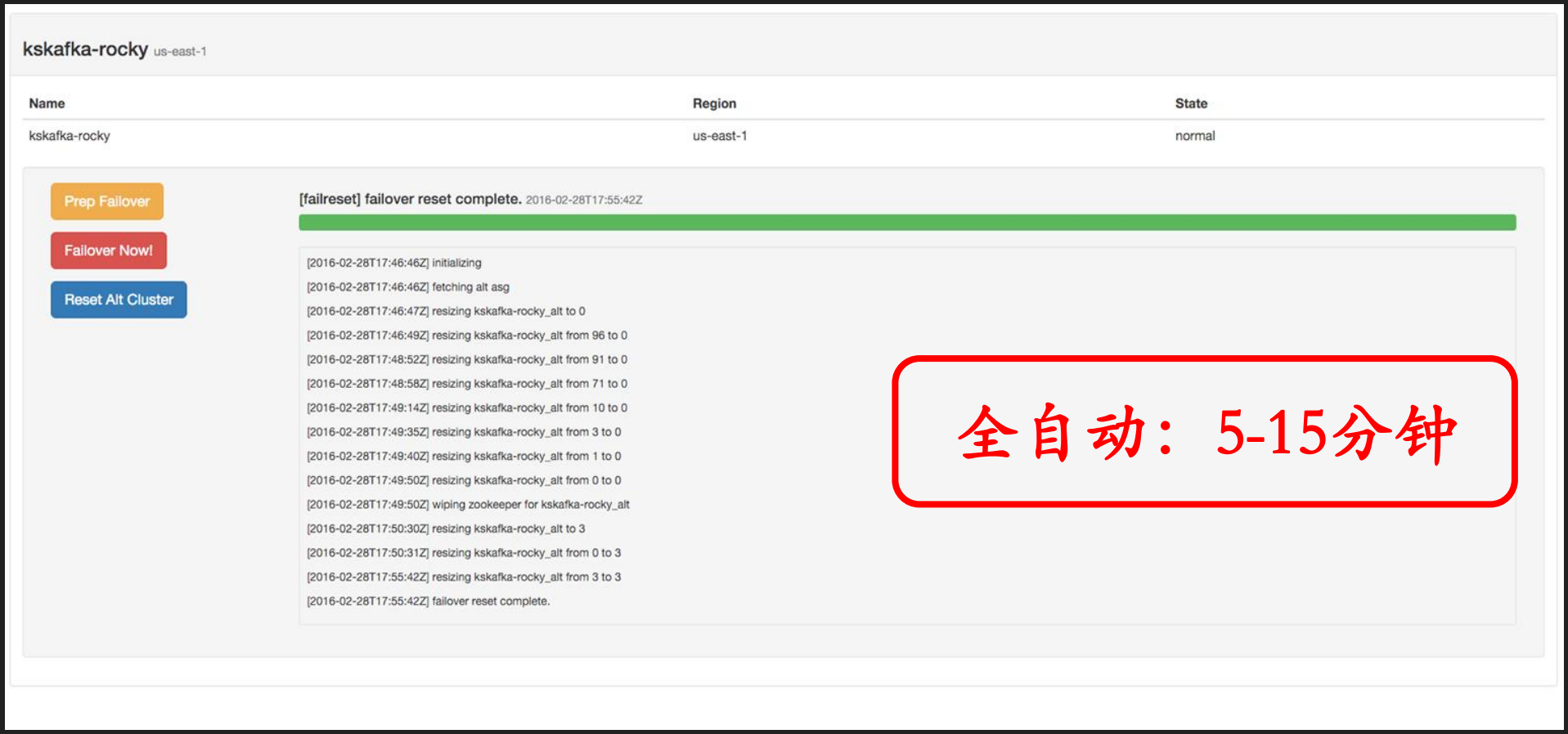
- 期待不可预测的故障并有所准备
- 有时重建是更快的恢复策略

故障切换（failover）



- 生产者动态路由
- 消费者Kafka集群

时间就是生命


kskafka-rocky us-east-1
Name
Region
State
kskafka-rocky
us-east-1
normal
Prep Failover
Failover Now!
Reset Alt Cluster
[failreset] failover reset complete. 2016-02-28T17:55:42Z
[2016-02-28T17:46:46Z] initializing
[2016-02-28T17:46:46Z] fetching alt asg
[2016-02-28T17:46:47Z] resizing kskafka-rocky_alt to 0
[2016-02-28T17:46:49Z] resizing kskafka-rocky_alt from 96 to 0
[2016-02-28T17:48:52Z] resizing kskafka-rocky_alt from 91 to 0
[2016-02-28T17:48:58Z] resizing kskafka-rocky_alt from 71 to 0
[2016-02-28T17:49:14Z] resizing kskafka-rocky_alt from 10 to 0
[2016-02-28T17:49:35Z] resizing kskafka-rocky_alt from 3 to 0
[2016-02-28T17:49:40Z] resizing kskafka-rocky_alt from 1 to 0
[2016-02-28T17:49:50Z] resizing kskafka-rocky_alt from 0 to 0
[2016-02-28T17:49:50Z] wiping zookeeper for kskafka-rocky_alt
[2016-02-28T17:50:30Z] resizing kskafka-rocky_alt to 3
[2016-02-28T17:50:31Z] resizing kskafka-rocky_alt from 0 to 3
[2016-02-28T17:55:42Z] resizing kskafka-rocky_alt from 3 to 3
[2016-02-28T17:55:42Z] failover reset complete.
全自动：5-15分钟

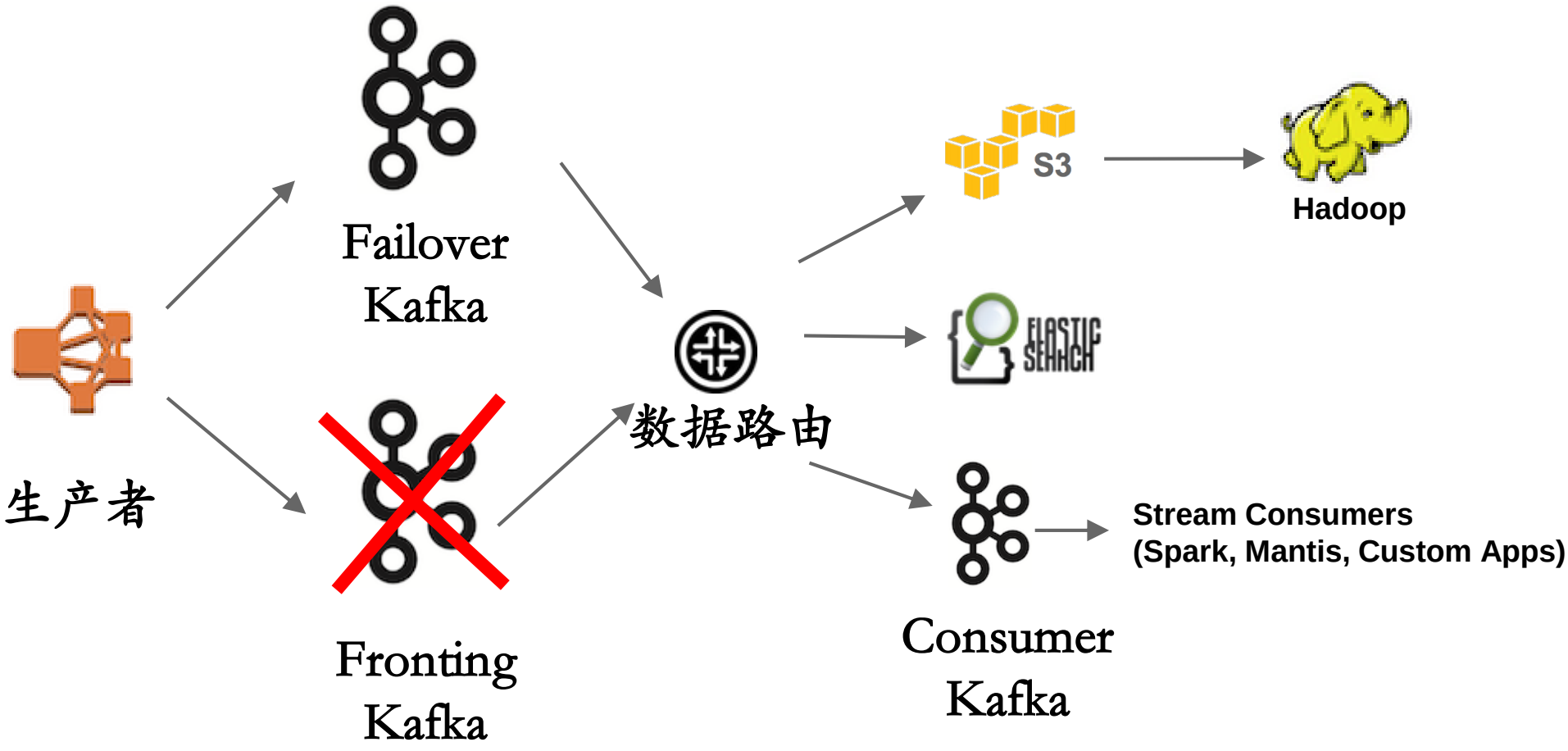
切换后的故障集群

- 解决问题
- 重建集群

利用切换做代码更新

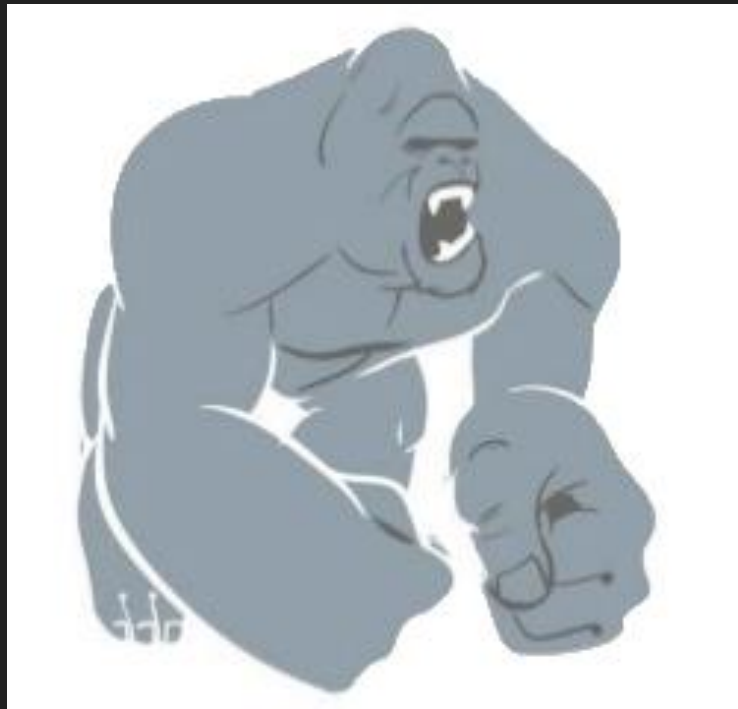
- 滚动更新可长达十几天
- 切换并重建小于一个小时

故障复原（failback）



Kafka Kong

- On-call工程师每周切换一个集群
- 目的
  - On-call熟悉切换过程
  - 测试切换工具

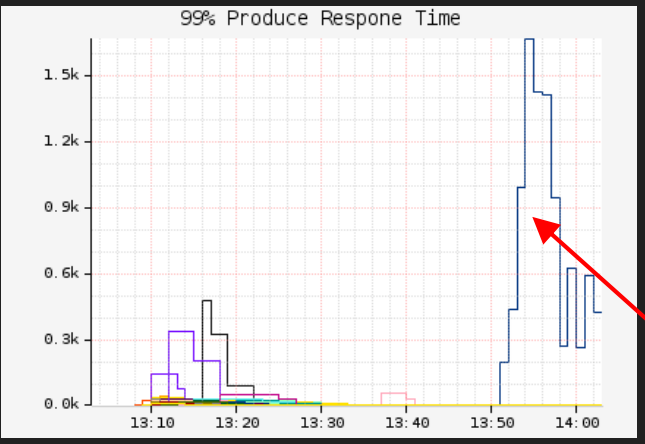


异常（Outlier）代理

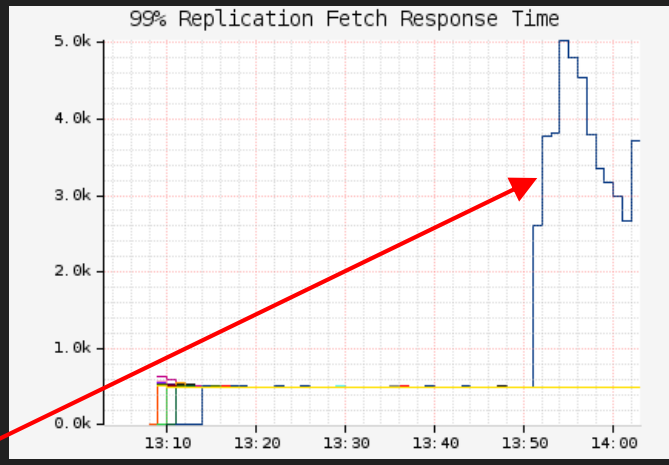
原因

- EC2实例硬件问题
- 网络堵塞

基于指标的检测

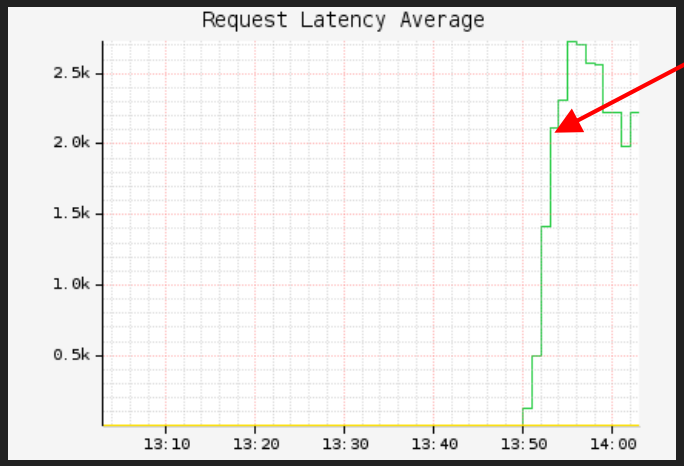


代理端回复延迟

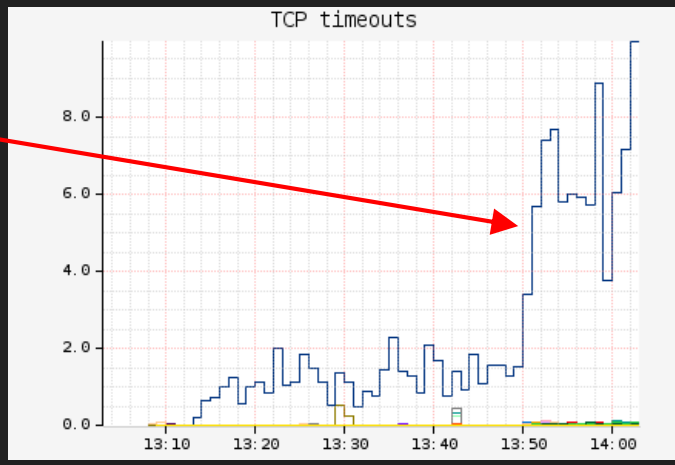


代理端复制回复延迟

同一个代理在不同的指标显示异常

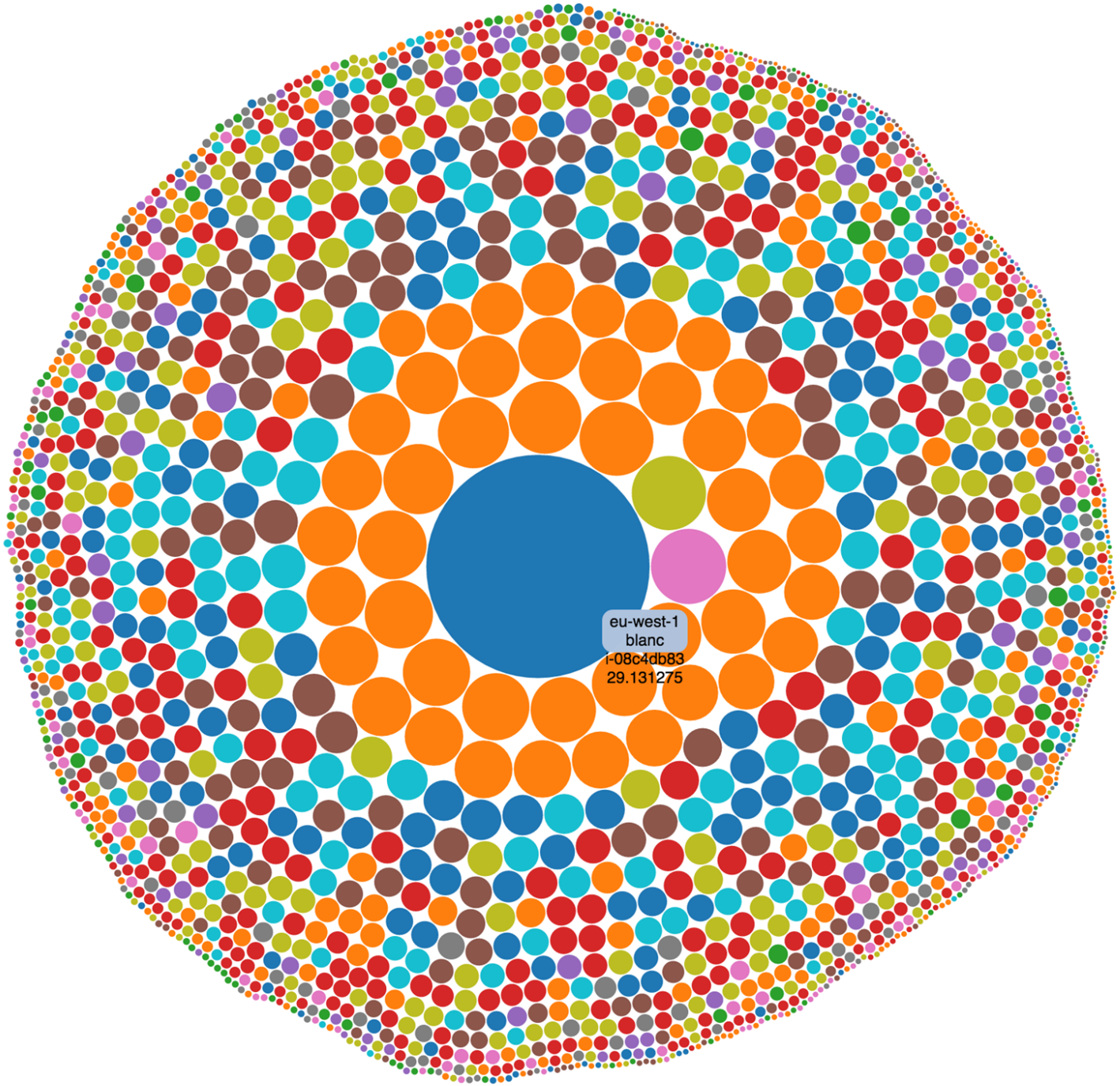


生产者端回复延迟



代理端TCP超时

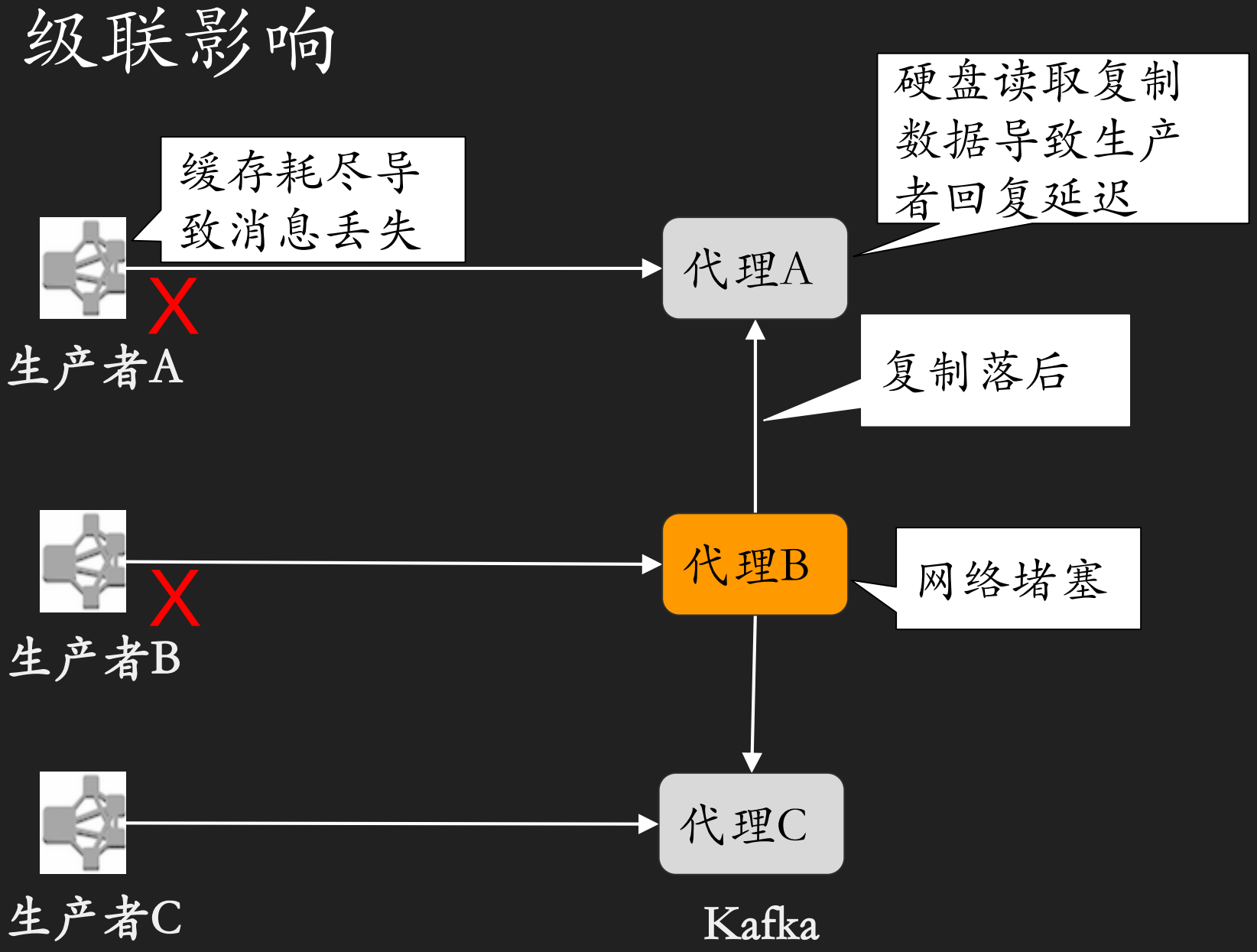
可视化



直接影响

延迟的回复导致生产者的缓存（32-512MB）耗尽，进而消息丢失

级联影响


硬盘读取复制数据导致生产者回复延迟
缓存耗尽导致消息丢失
代理A
复制落后
生产者A
代理B
网络堵塞
生产者B
代理C
生产者C
Kafka

总结与展望

总结

- 设计灵活的系统
- 拥抱失败并准备恢复策略

生产者

- JSON -> Schema（avro/protobuf）
- 流量配额（Quota）

路由器

- Titus：Netflix容器管理和调度平台
- Apache Beam with Flink runner

控制层

- 完善自助式服务
- Skynet：DevOps -> NoOps

Keystone数据流水线博客:

http://techblog.netflix.com/search?q=keystone

Questions?

Twitter：@stevenzwu

LinkedIn: https://www.linkedin.com/in/stevenzhenwu