驴妈妈旅游网
www.lvmama.com

KAFKA从入门到放弃

出境业务开发 储强

自在游天下，就找驴妈妈


Kafka
简介
设计
操作
监控
配置解读
存储结构
应用场景
其它MQ

Kafka简介

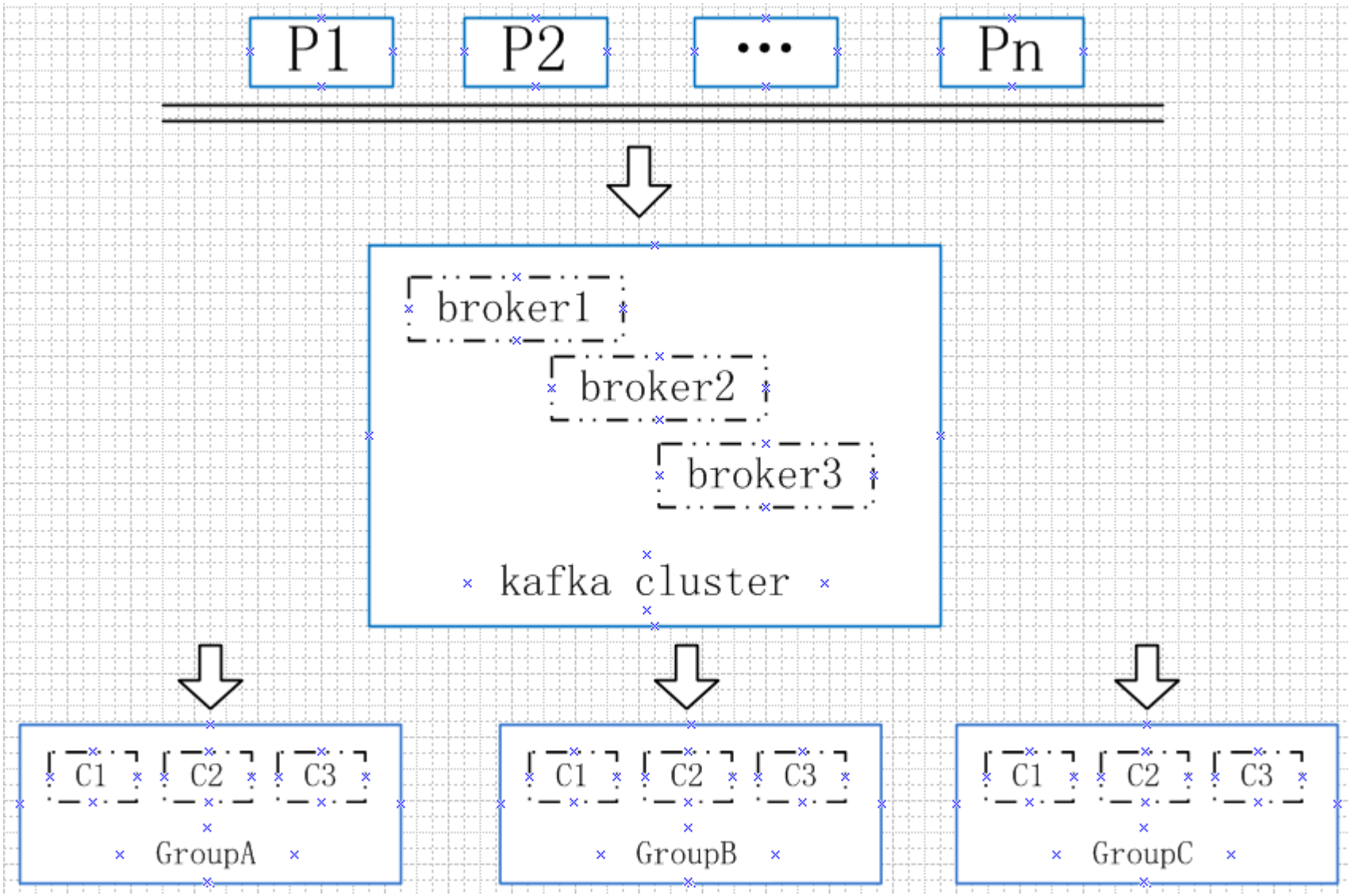
1. 同时为发布和订阅提供高吞吐量。Kafka每秒产生25万条消息，每秒处理55万条消息。

2. 可进行持久化操作。将消息批量持久化到磁盘，可以批量消费。同时消息备份，防止数据丢失。

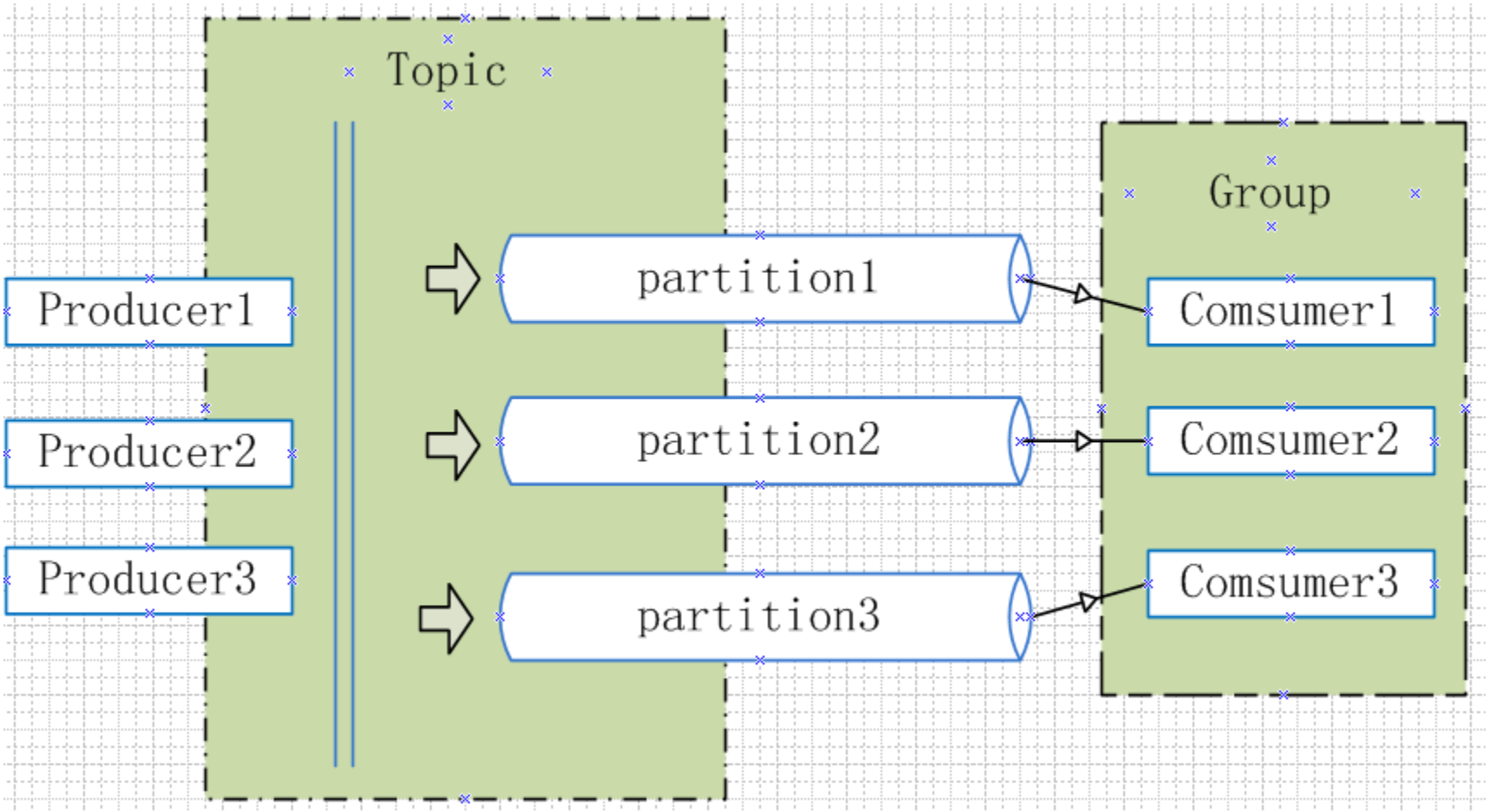
3. 分布式系统，高伸缩性。多个生产者，多个消费者，支持集群。无需停机，即可扩展。

4. 消息被消费的状态是保存在消费者端，而不是在服务器端。当一个消费者挂掉，可以消费者重新平衡。

发布订阅模型


P1
P2
...
Pn
broker1
broker2
broker3
kafka cluster
C1
C2
C3
GroupA
C1
C2
C3
GroupB
C1
C2
C3
GroupC

Kafka的Topic和Partition



集群配置

broker.id=1

zookeeper.connect=10.113.10.190:2181,10.113.10.191:2181,10.113.10.192:2181

listeners=PLAINTEXT:10.113.10.190:9092

log.retention.hours=168

num.partitions=3

default.replication.factor=3

num.io.threads=8

num.network.threads=3

空闲连接

connections.max.idle.ms

空闲连接超时时间：socket处理线程会关闭空闲的socket链接。

集群服务器：connections.max.idle.ms=600000
生产者：connections.max.idle.ms=540000
消费者：connections.max.idle.ms=540000

生产者发消息 | 消费者同步偏移量 | 生产者同步元数据

Logging.scala:89- Failed to send producer request with correlation id 4126 to broker 3 with data for partitions [VST_LOG,1]java.io.EOFException: Received -1 when reading from channel, socket has likely been closed. at kafka.utils.Utils$.read(Utils.scala:381) at kafka.network.BoundedByteBufferReceive.readFrom(BoundedByteBufferReceive.scala:54)

生产者配置

bootstrap.servers=10.113.10.190:9092,10.113.10.191:9092,10.113.10.192:9092

key.serializer=

value.serializer=

acks=0/1/-1 应答机制

batch.size= 16384 生成者会将多个消息请求合并成一个大的消息发送给集群

linger.ms= 合并多个消息请求的时间限制

max.request.size=1048576 一个请求最大的大小，会影响batch.size.

request.timeout.ms=30000

消息应答

acks = 0 / 1 / -1

0：生产者不会等待任何应答，只要消息发送到 socket buffer 中，就认为消息发送成功。

1：leader会将消息保存起来，不会等待 follower 的应答。如果leader失败了，消息就会丢失。

-1：leader将会等待 **in-sync** 中的follower的应答。 in-sync 对应着集群配置 min.insync.replicas,默认1.

消费者配置

group.id=vst_log

zookeeper.connect=10.113.10.190:2181,10.113.10.191:2181,10.113.10.192:2181

fetch.message.max.bytes=1024 * 1024

num.consumer.fetchers=1　　消费者拉取数据的线程

auto.commit.enable=ture

auto.commit.interval.ms=60* 1000

rebalance.max.retries=4　　当一个新的消费者加入消费者组。

offsets.storage=zookeeper/kafka

文件系统IO慢？？？

Kafka非常的依赖文件系统(linux)来存储和缓存消息。

1. 磁盘的性能

线性读写的速度：600M/s　随机读写的速度：100k/s

操作系统极大的优化了 线性写和读.操作系统提供了读优先的技术,提供了批量读写数据.

2. 使用内存

操作系统使用空闲的主内存来做磁盘缓存，所有的读写都基于这些未使用的缓存。

3. Jvm内存

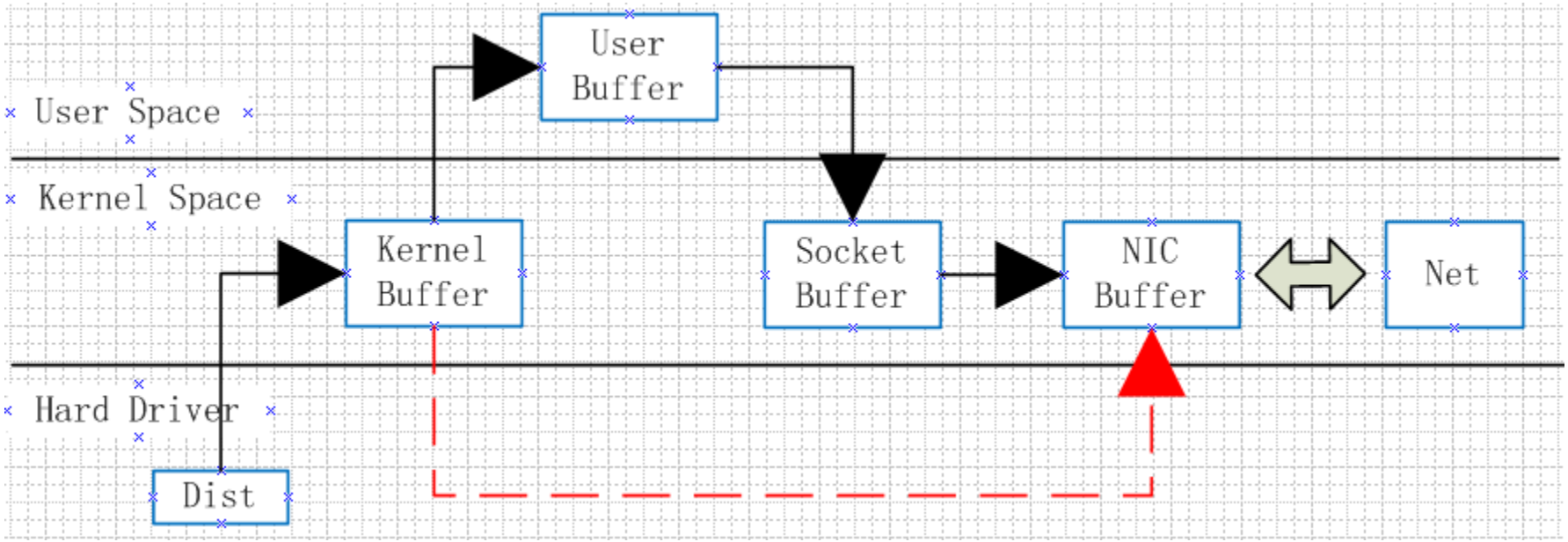
对象的内存瓶颈很高,两倍于数据存储.

当Jvm内存使用很大,GC就会很慢.

Kafka Bytes Zero-Copy

1. 生产者、kafka服务器和消费者 之间 采用统一的二进制消息格式.

2. 消息网络传输的优化.



高效

1. 消息压缩格式的一致性、支持消息批量的压缩。

2. 生产者直接将消息发送到对应分区的leader所在机器，没有任何的转发代理。PS：这一点和redis集群类似。

3. 生产者可以选择将消息发送到某一个分区。

4. 批处理：生产者将多个消息请求合并成一个大的消息请求。这里的合并受限于 消息的数量和延迟时间。

Push vs. Pull

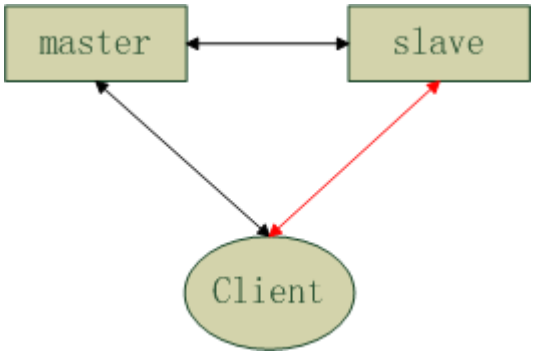
1. Kafka：生产者将消息push到集群，消费者从集群pull消息。

2. 集群push到消费者 OR 消费者从集群pull消息？

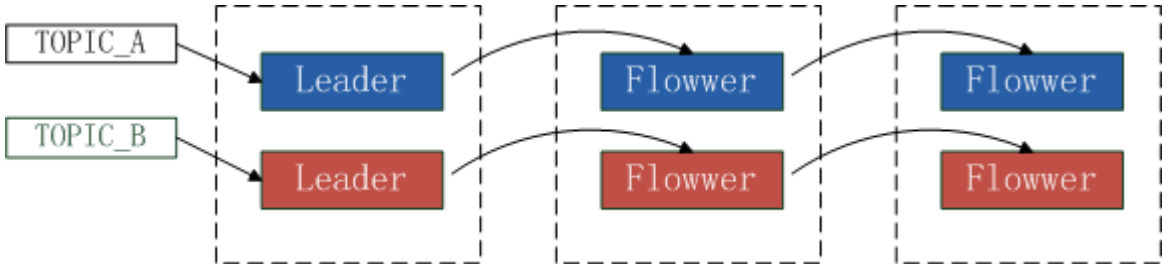
Push模型	实时性好	服务端维护消息状态	服务端需要做流量控制，无法最大化客户端的处理能力。 其次，在客户端故障情况下，无效的push对服务端有一定负载。	缺点:服务器端的状态存储是个难点，可以将这些状态转移到DB或者key-value存储，来减轻server压力。
Pull模型	依赖于轮询间隔时间	消费端维护消息状态	客户端的请求可能很多无效或者没有数据可供传输，浪费带宽和服务器处理能力 优点：批量获取消息，减少带宽	缺点:针对实时性的问题，可以将push加入进来，push小数据的通知信息，让客户端再来主动pull。 针对无效请求的问题，可以设置逐渐延长间隔时间的策略，以及合理设计协议尽量缩小请求数据包来节省带宽。

复制—容灾

主备模式(activemq)



分区模式(kafka)



消息存储方式

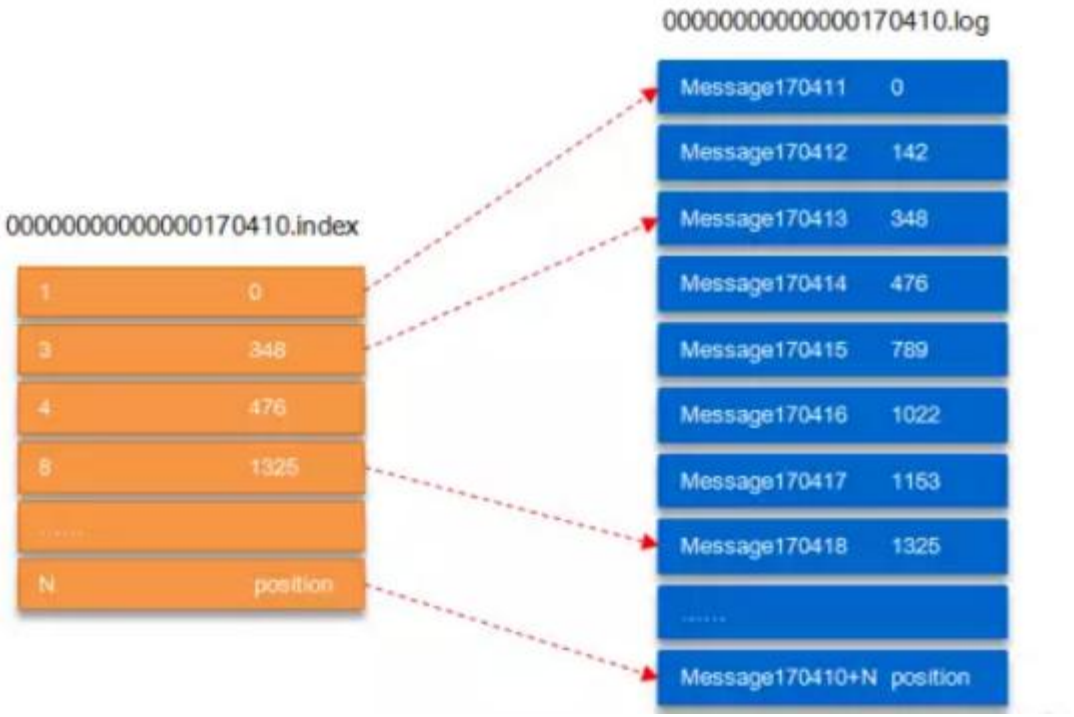
topic：VST_LOG　　partition: 3

```
[root@chujing190 kafka-logs]#
[root@chujing190 kafka-logs]# pwd
/tmp/kafka-logs
[root@chujing190 kafka-logs]# ll | grep VST_LOG | grep -v VST_LOG_ORDER
drwxr-xr-x 2 root root  107 Jul  7 06:22 VST_LOG-0
drwxr-xr-x 2 root root  107 Jul  7 06:42 VST_LOG-1
drwxr-xr-x 2 root root  107 Jul  7 03:12 VST_LOG-2
[root@chujing190 kafka-logs]#
```

消息存储方式

topic：VST_LOG　partition: 3　VST_LOG-0

00000000000000000000.index
00000000000000000000.log
00000000000000170410.index
00000000000000170410.log
00000000000000239430.index
00000000000000239430.log



使用场景

日志收集中心

消息系统

网站活性跟踪

流处理

事件源

监控

Summary

# of Topics	6
# of Partitions	14
% of Messages	0.000
% of Incoming	0.000
% of Outgoing	0.000

Metrics

Rate	Mean	1 min	5 min	15 min
Messages in /sec	0.00	0.00	0.00	0.00
Bytes in /sec	0.02	0.00	0.00	0.00
Bytes out /sec	0.05	0.00	0.00	0.00
Bytes rejected /sec	0.00	0.00	0.00	0.00
Failed fetch request /sec	0.00	0.00	0.00	0.00
Failed produce request /sec	0.00	0.00	0.00	0.00

Messages count

Per Topic Detail

Show 10 entries Search:

Topic	Replication	Total Partitions	Partitions on Broker	Partitions	Skewed?
my-replicated-topic	3	1	1	(0)	false
PROD_QUERY	3	3	3	(0,1,2)	false
test	2	1	1	(0)	false
TEST	3	3	3	(0,1,2)	false
VST_LOG	3	3	3	(0,1,2)	false
VST_LOG_ORDER	3	3	3	(0,1,2)	false

Showing 1 to 6 of 6 entries Previous 1 Next

RocketMQ

1. 数据可靠性

RocketMQ支持异步实时刷盘，同步刷盘，同步Replication，异步Replication

Kafka使用异步刷盘方式，异步Replication

2. 性能对比

Kafka单机写入TPS约在百万条/秒，消息大小10个字节

RocketMQ单机写入TPS单实例约7万条/秒，单机部署3个Broker，可以跑到最高12万条/秒，消息大小10个字节

3. 单机支持的队列数

Kafka单机超过64个队列/分区，Load会发生明显的飙高现象，队列越多，load越高，发送消息响应时间变长

RocketMQ单机支持最高5万个队列，Load不会发生明显变化

4. 有序性

Kafka支持消息顺序，但是一台Broker宕机后，就会产生消息乱序

RocketMQ支持严格的消息顺序，在顺序消息场景下，一台Broker宕机后，发送消息会失败，但是不会乱序

5. 分布式事务

Kafka不支持分布式事务消息

阿里云ONS支持分布式定时消息，未来开源版本的RocketMQ也有计划支持分布式事务消息

. . . . . .

ActiveMQ

1. 实现了Java消息服务(JMS)，传统的消息队列模型。Kafka是一种伪MQ。
2. 支持发布订阅和点到点消息，而kafka不支持点到点消息。
3. 竞争关系的消费者模型。
4. 消息是推送给消费者的，而kafka是消费者主动拉取消息的。
5. 支持主备服务器集群，而kafka是多服务器、多分区、多备份、多数选举的集群。

Thank you !