内容大纲

- Kafka简介
- Kafka中角色与术语
- Kafka系统架构
- 生产者
- 消费者
- Kafka命令基本流程

Kafka简介

- Kafka是一个分布式发布-订阅消息系统。它最初由LinkedIn公司开发，之后成为Apache项目的一部分。
- Kafka主要特点

  (1) 高吞吐量。据了解，Kafka每秒可以生产约25万条消息（50 MB），每秒处理55万条消息（110 MB）。

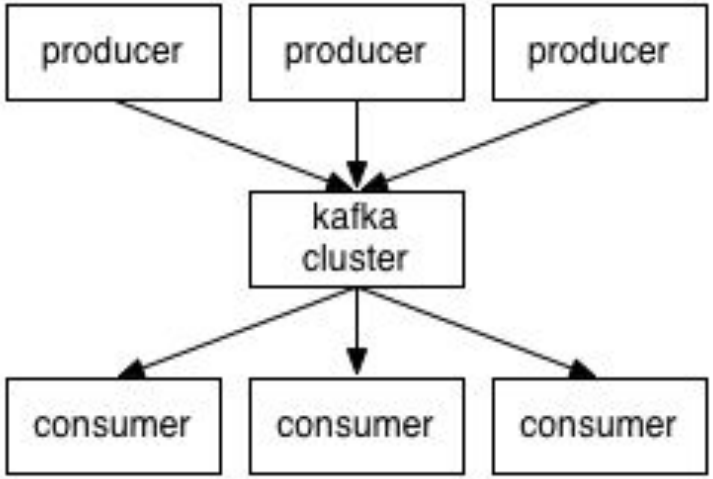
  (2) 持久化。将消息持久化到磁盘，因此可用于批量消费，例如ETL，以及实时应用程序。通过将数据持久化到硬盘以及replication防止数据丢失。

  (3) 分布式系统。所有的producer、broker和consumer都会有多个，均为分布式的。

  (4) 可扩展性。kafka使用zookeeper来实现动态的集群扩展，不需要更改客户端（producer和consumer）的配置。broker会在 zookeeper注册并保持相关的元数据（topic，partition信息等）更新。而客户端会在zookeeper上注册相关的watcher。

Kafka中角色与术语

- Producer
  向kafka中发布消息的进程
- Comsumer
  从Kafka中订阅消息的进程
- Broker
  Kafka集群中每一个kafka服务
- Topic
  保存在Kafka中的每一类消息
  它们之间的数据流程如右图所示

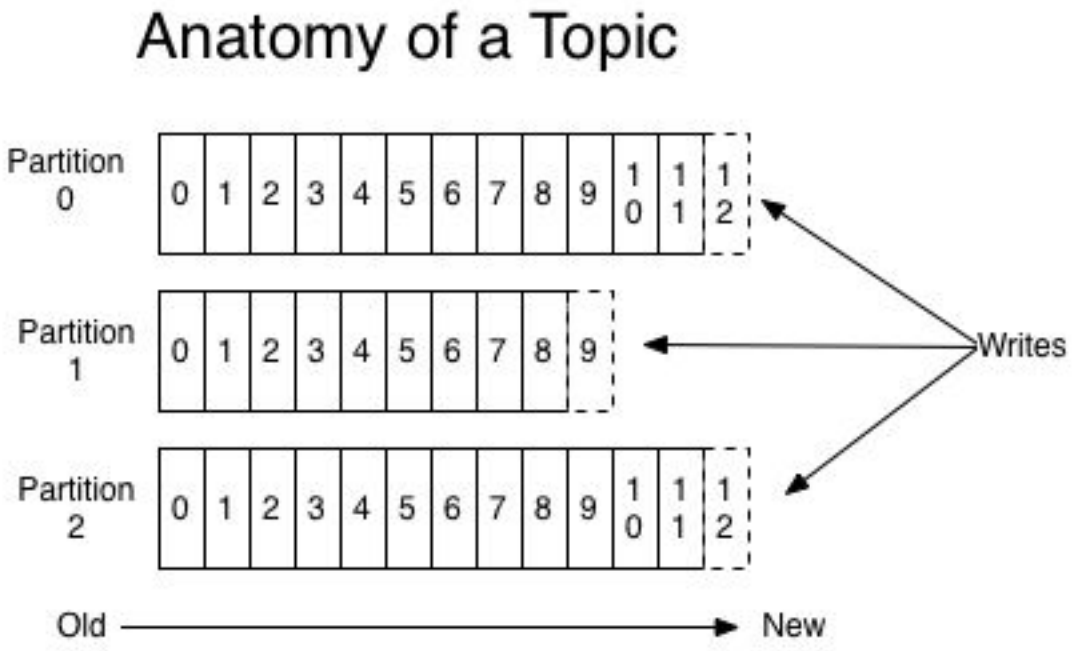


Kafka中角色与术语

- Partition

  每一个topic可以被分成多个partition(分区)

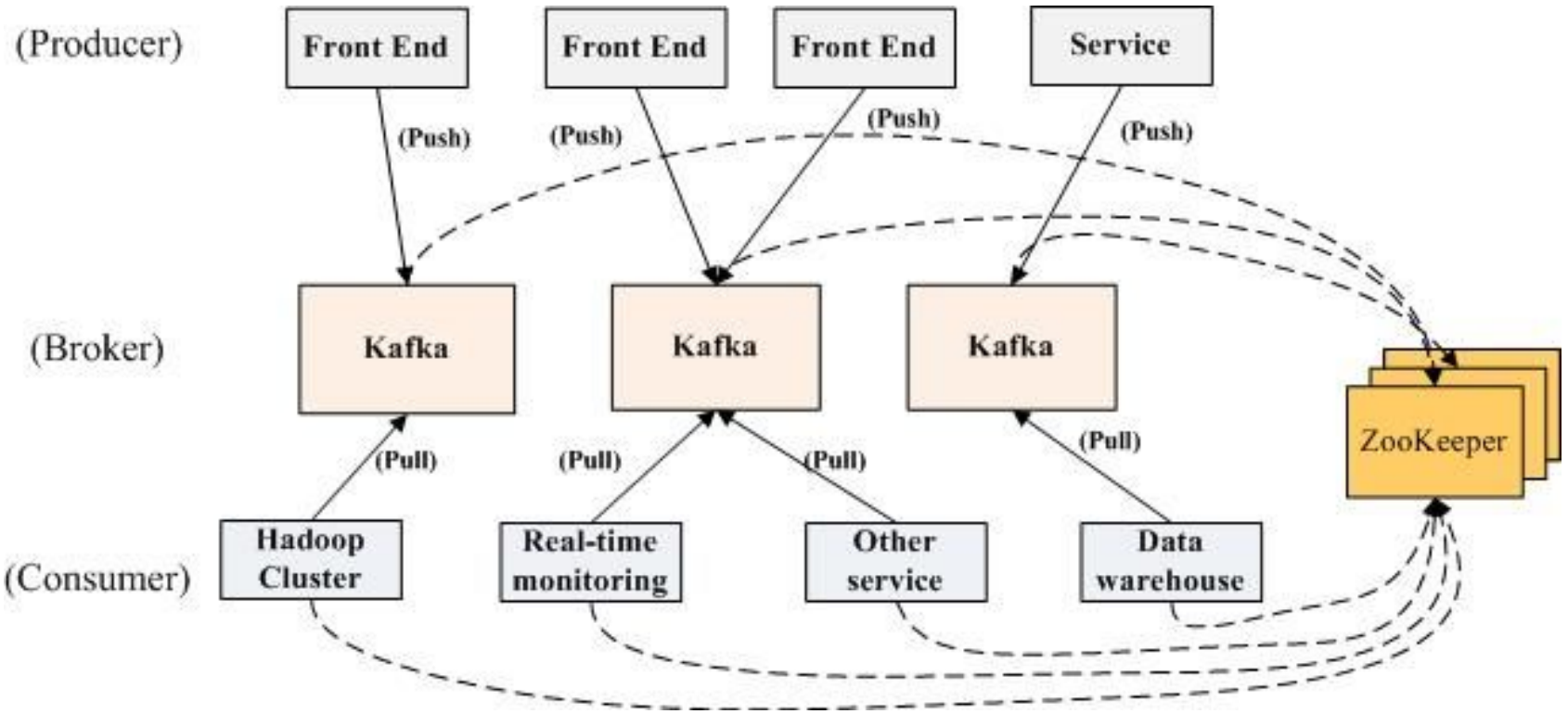
Kafka集群中维护的partitions如图所示：



Kafka中角色与术语

- Replication
  一个partition可以有多个备份,默认为1。
- Leader/Follower
  每个partition都有一个唯一的leader，所有的读写操作都在leader上完成。
- Offset
  每个消息在partition中的位置叫做offset。
- Consumer Group
  同一个Consumer Group中的consumers，Kafka将相应Topic中每个消息只发送给其中一个Consumer。

Kafka系统架构



kafka是显式分布式架构，producer、broker（Kafka）和consumer都可以有多个。Kafka的作用类似于缓存，即活跃的数据和离线处理系统之间的缓存

负载均衡

- 负载均衡可以分为两个部分：producer发消息的负载均衡和consumer读消息的负载均衡。
- producer有一个到当前所有broker的连接池，当一个消息需要发送时，需要决定发到哪个broker（即partition）。这是由 partitioner实现的，partitioner是由应用程序实现的。
- broker和consumer之间利用zookeeper进行负载均衡。所有broker和consumer都会在zookeeper中进行注册，且 zookeeper会保存他们的一些元数据信息。如果某个broker和consumer发生了变化，所有其他的broker和consumer都会得到 通知。

生产者（Producer）

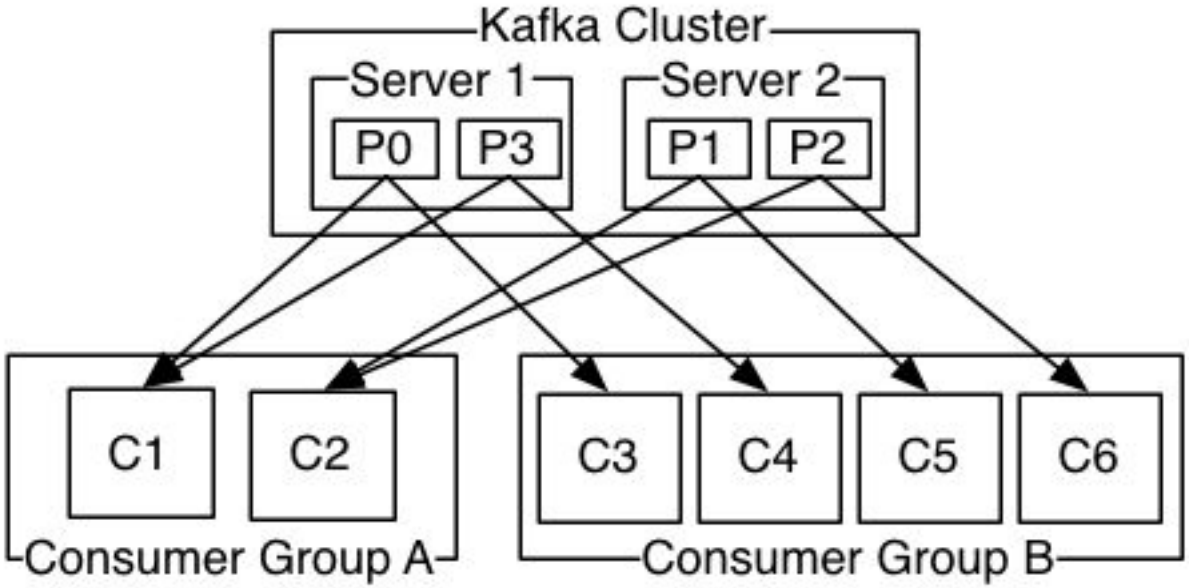
- Producer将消息发布到指定的Topic中。
- 同时Producer也能决定将此消息归属于哪个partition;比如基于"random","round-robin", "key-hash"方式或者通过其他的一些算法等。

生产者（Producer）

```
1.  ##对于开发者而言,需要通过broker.list指定当前producer需要关注的broker列表
2.  ##producer通过和每个broker链接,并获取partitions,
3.  ##如果某个broker链接失败,将导致此上的partitons无法继续发布消息
4.  ##格式:host1:port,host2:port2,其中host:port需要参考broker配置文件.
5.  ##对于producer而言没有使用zookeeper自动发现broker列表，非常奇怪。(0.8V和0.7有区别)
6.  metadata.broker.list=
7.  ##producer接收消息ack的时机.默认为0.
8.  ##0: producer不会等待broker发送ack
9.  ##1: 当leader接收到消息之后发送ack
10. ##2: 当所有的follower都同步消息成功后发送ack.
11. request.required.acks=0
12. ##producer消息发送的模式,同步或异步.
13. ##异步意味着消息将会在本地buffer,并适时批量发送
14. ##默认为sync,建议async
15. producer.type=sync
16. ##消息序列化类,将消息实体转换成byte[]
17. serializer.class=kafka.serializer.DefaultEncoder
18. key.serializer.class=${serializer.class}
19. ##partitions路由类,消息在发送时将根据此实例的方法获得partition索引号.
20. partitioner.class=kafka.producer.DefaultPartitioner
21.
22. ##消息压缩算法,none,gzip,snappy
23. compression.codec=none
24. ##消息在producer端buffer的条数.仅在producer.type=async下有效
25. ##batch.num.messages=200
```

消费者（Consumer）

- 消费者需要获取数据时,向broker发送fetch请求，并告知所要获取消息的offset，随后consumer会得到相应的数据。Consumer端也可以重置offset来重新消费消息。
- Kafka集群中对应Topic中的一个分区只能被Consumer Group中一个Consumer消费



消费者（Consumer）

```
1.  ##当前消费者的group名称,需要指定
2.  group.id=
3.  ##consumer作为zookeeper client,需要通过zk保存一些meta信息,此处为zk connectString
4.  zookeeper.connect=hostname1:port,hostname2:port2
5.  ##当前consumer的标识,可以设定,也可以有系统生成.
6.  conusmer.id=
7.  ##获取消息的最大尺寸,broker不会像consumer输出大于此值的消息chunk
8.  ##每次feth将得到多条消息,此值为总大小
9.  fetch.messages.max.bytes=1024*1024
10. ##当consumer消费一定里的消息之后,将会自动向zookeeper提交offset信息
11. ##注意offset信息并不是每消费一次消息,就像zk提交一次,而是现在本地保存,并定期提交
12. auto.commit.enable=true
13. ##自动提交的时间间隔,默认为1分钟.
14. auto.commit.interval.ms=60*1000
```

http://kafka.apache.org/documentation.html#quickstart

Kafka命令基本流程

- 创建topic

bin/kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 1 --partitions 1 --topic test

- 查看topic

bin/kafka-topics.sh --list --zookeeper localhost:2181

bin/kafka-topics.sh --describe --zookeeper localhost:2181 --topic test

Kafka命令基本流程

- 发布消息

bin/kafka-console-producer.sh --broker-list localhost:9092 --topic test

- 消费消息

bin/kafka-console-consumer.sh --zookeeper localhost:2181 --topic test --from-beginning