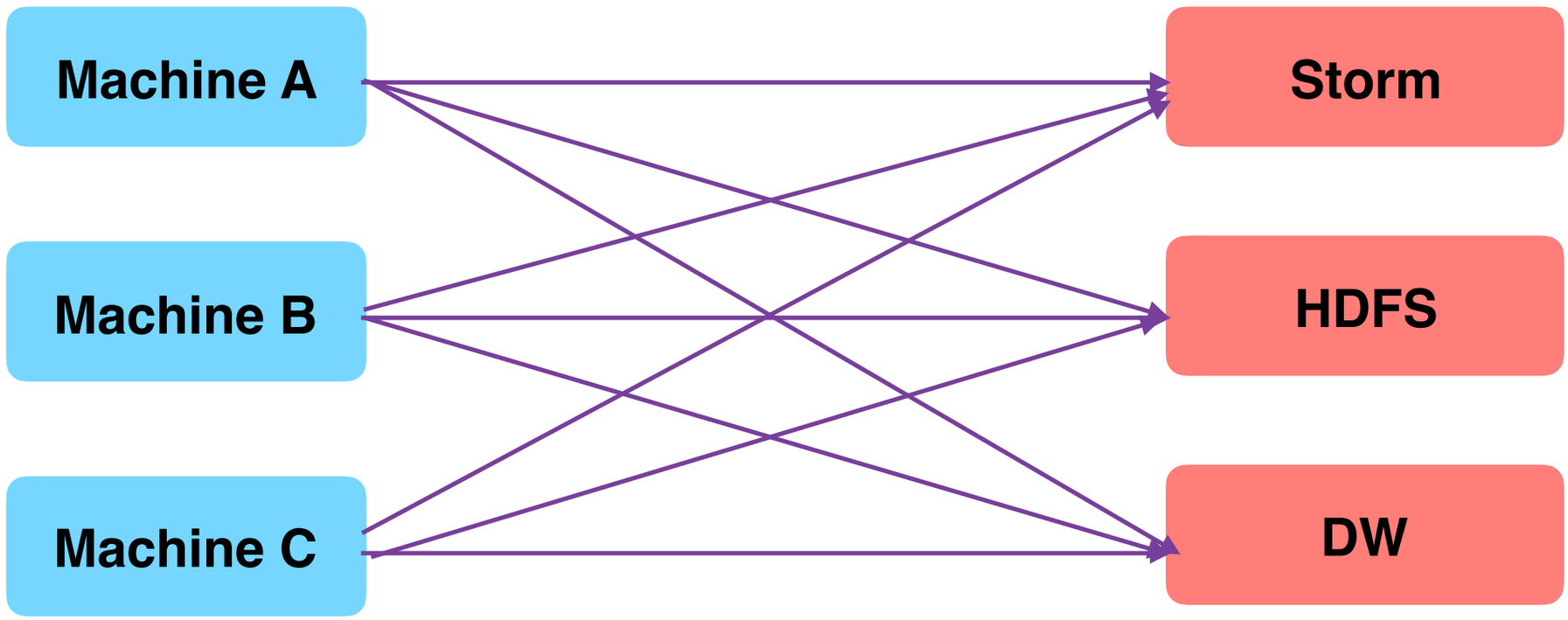
从 MQ 说到 Kafka

guoning02@猫眼

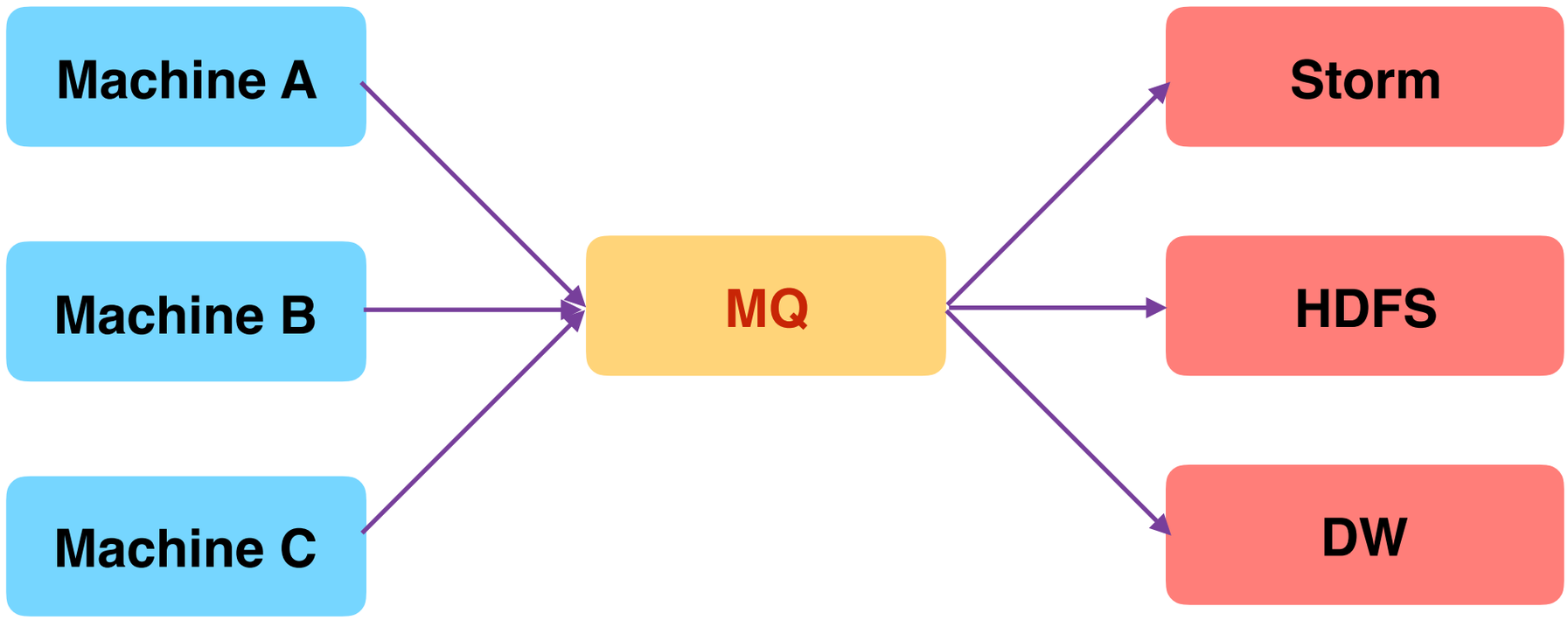
2015/12/23

目录

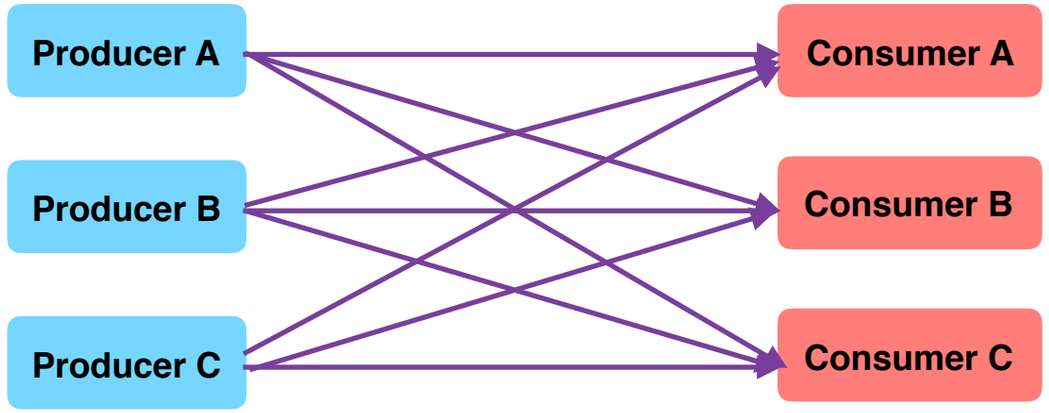
MQ 有什么用？



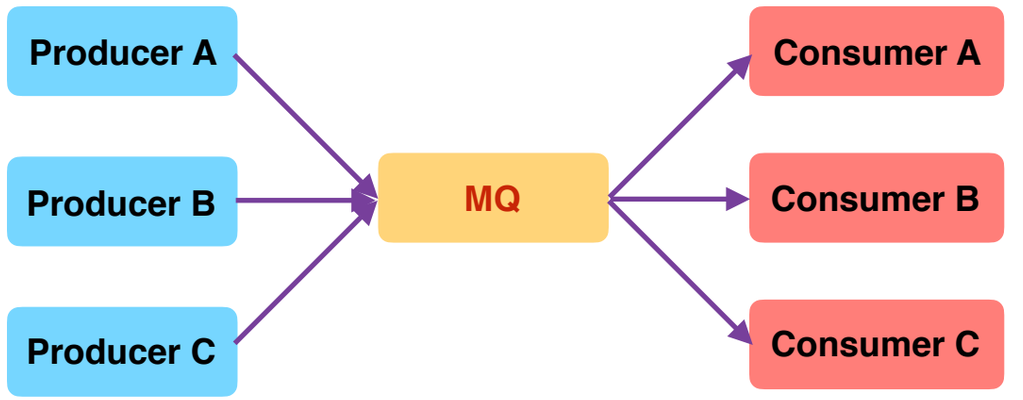
MQ 有什么用？



MQ 有什么用？



VS.



	可扩展性	可靠性
Producer	😀	😷
Consumer	😀	😀

MQ 有什么用？

- 其他应用场景：
  - 平滑突发峰值：系统突发处理能力不足，平均处理能力可以，MQ 平滑突发峰值
  - 异步任务：耗时的任务

- 更多参考：
  - Top 10 Uses For A Message Queue

小结

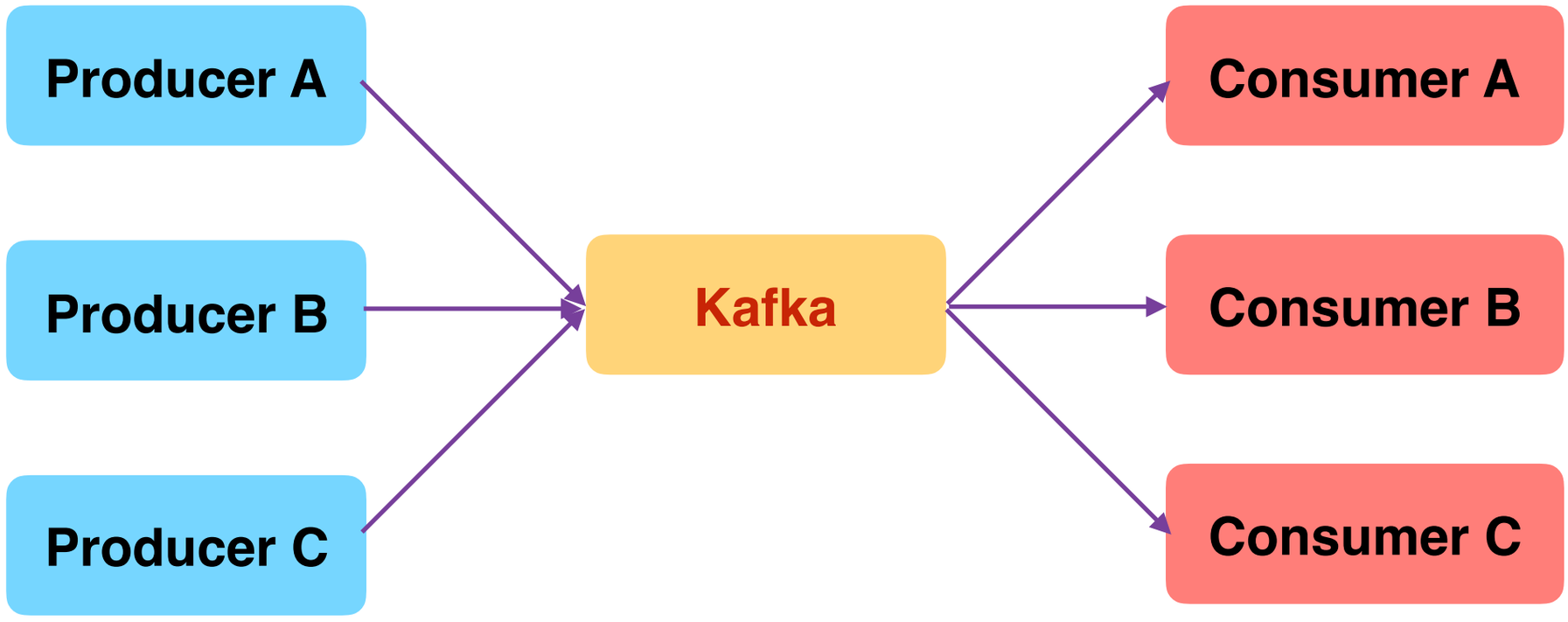
- MQ 有什么用？
  - 可扩展性
  - 可靠性
  - 平滑突发峰值
  - 异步任务

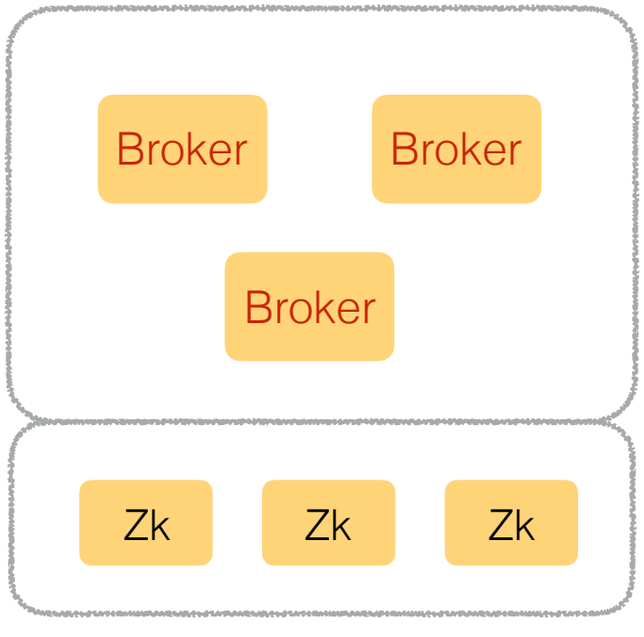
Kafka 设计原理

- 整体结构
- 关键术语
  - topic
  - consumer group
  - partition
  - replica
- 相对于其他 MQ 的典型特点

MQ 的 基本原理

Kafka 整体结构

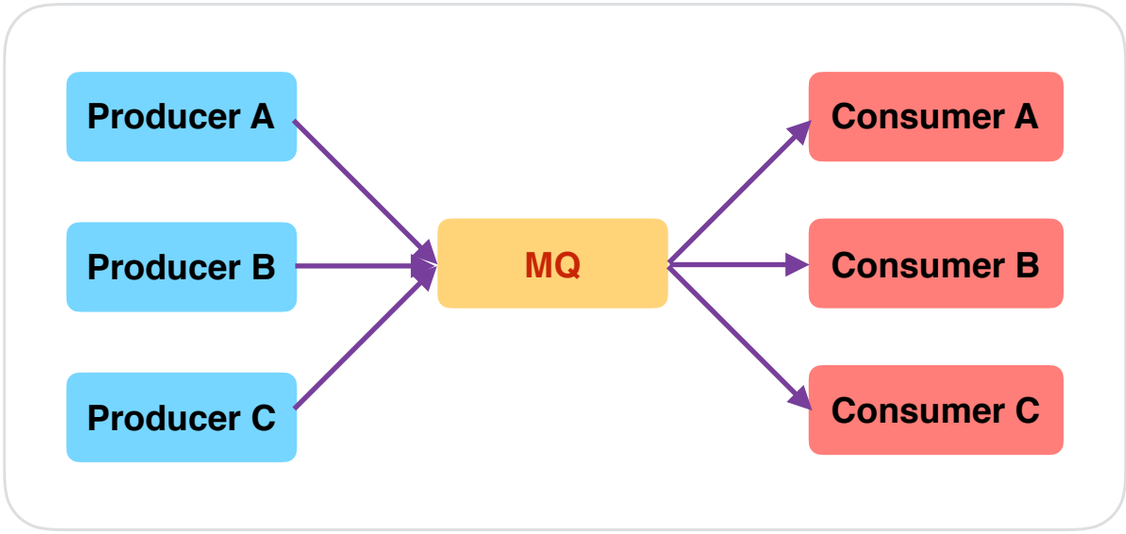

Producer A
Producer B
Producer C
Kafka
Consumer A
Consumer B
Consumer C


Broker
Broker
Broker
Zk
Zk
Zk

Kafka 关键术语

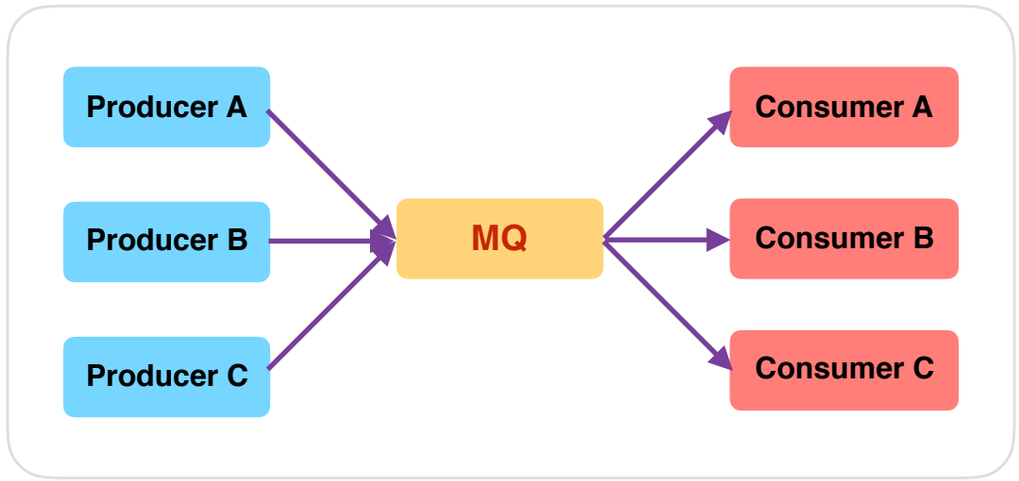
- topic
- consumer group
- partition
- replica

- **问题1**：不同业务数据，分开存放
  - OS：CPU、内存、网络
  - APP：pv、响应时间

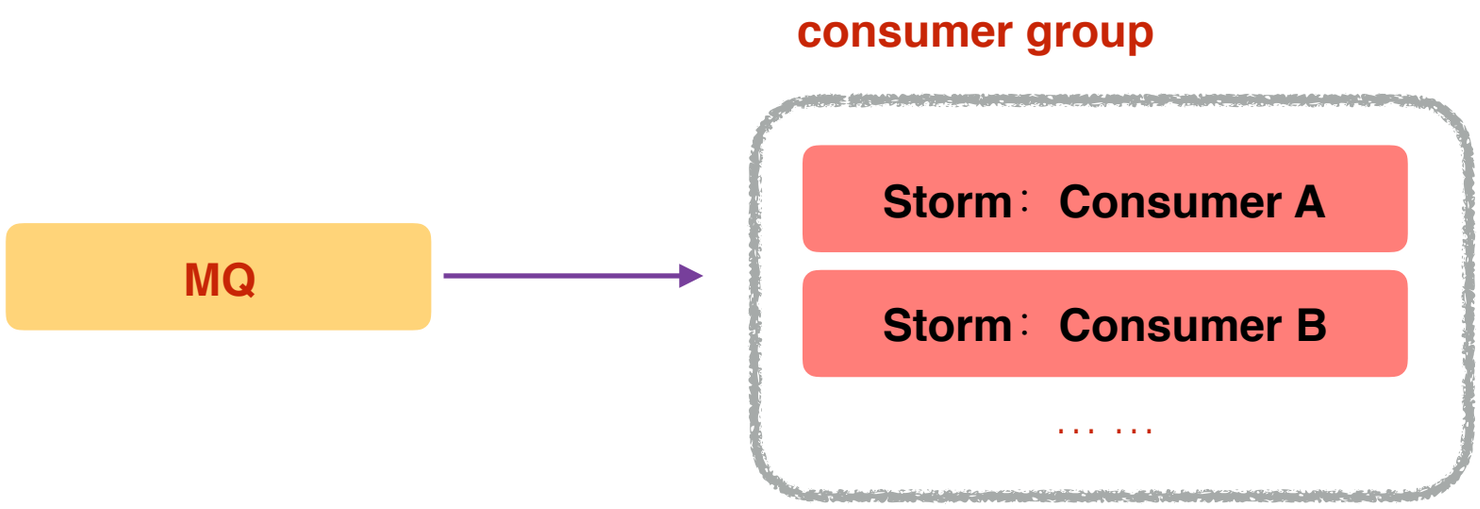


- **解决方式**：topic
  - 同一种数据放入同一个 topic
  - 不同数据，通过 topic 分离

- **问题2**：消息的单播、多播?
  - 单播：一条 msg，一个 consumer
  - 多播：一条 msg，多个 consumer

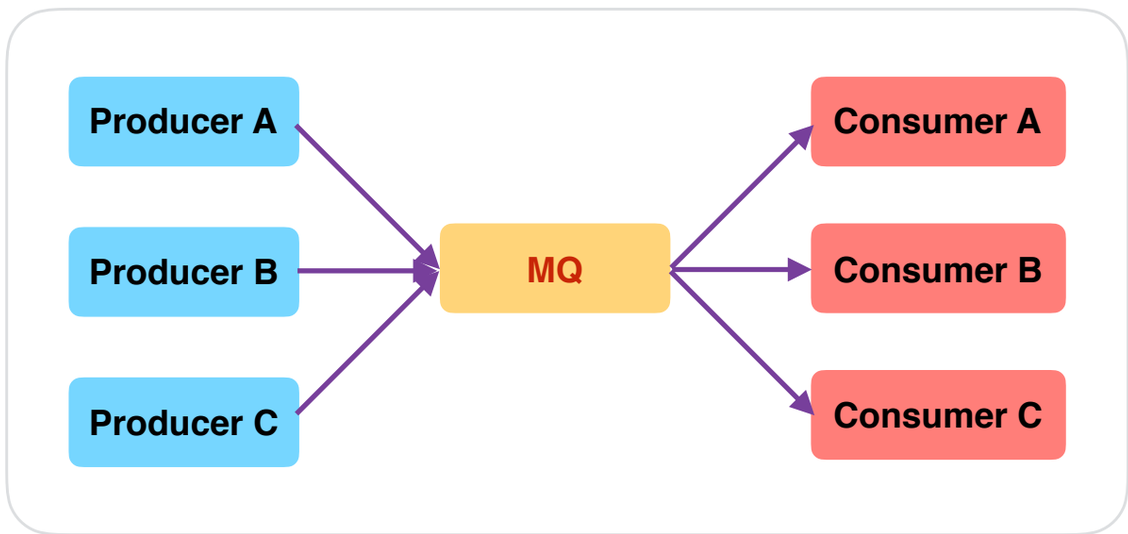


- **解决方式**： consumer group
  - 一个 msg，送入多个 consumer group
  - 一个 consumer group 中，只能有一个 consumer 处理某一条 msg



- **问题3**：并发效率
  - 同一类业务数据，多 producer，多 consumer
  - 并行处理?

- **解决方式**： partition
  - 一个 topic，划分为 多个 partition
  - partition 之间能够并行处理



Producer

Producer

Producer

partition A

6 5 4 3 2 1 0

partition B

4 3 2 1 0

... ...

Topic A

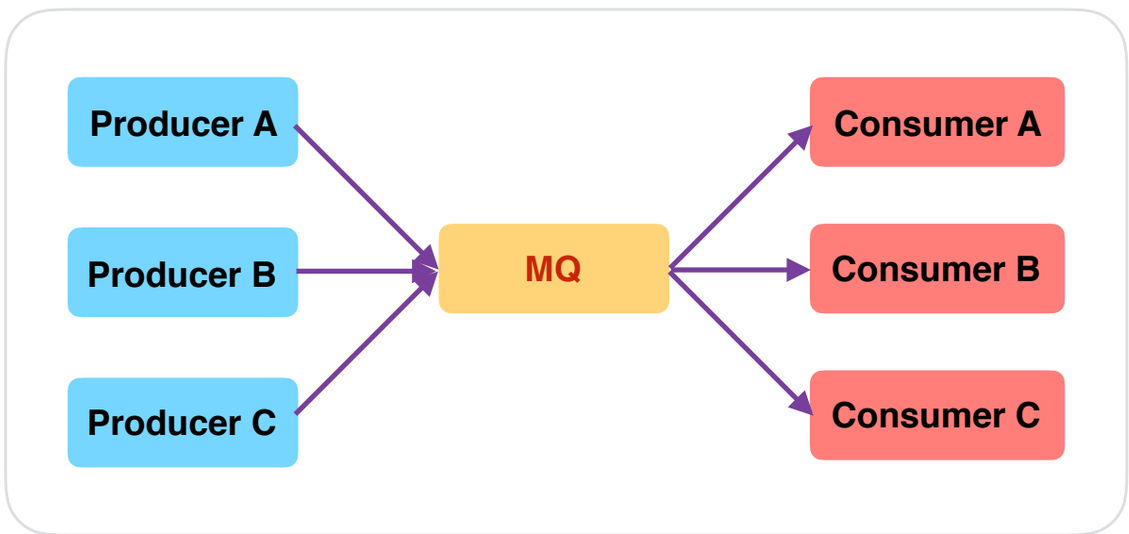
Consumer

Consumer

Consumer

- **问题4**：msg 之间的顺序保证
  - 推荐系统中，时间序列挖掘
  - 严格限制，msg 之间的先后顺序

- **解决办法**： partition 内部有序
  - 将有顺序要求的 msg，送入同一个 partition
  - 将时间戳，放入 msg 内



Producer

Producer

Producer

partition A

6 5 4 3 2 1 0

partition B

4 3 2 1 0

... ...

Topic A

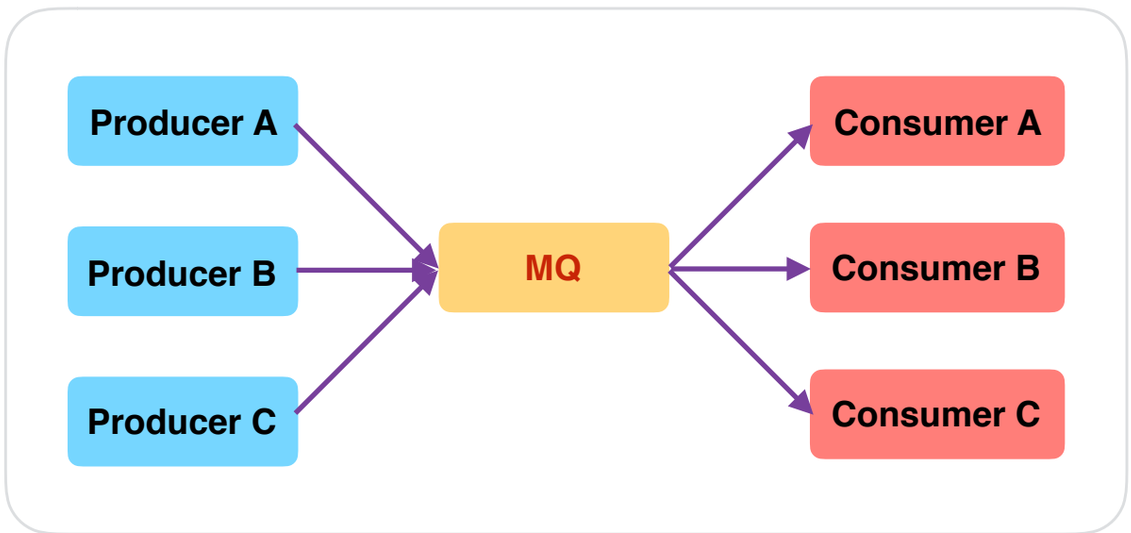
Consumer

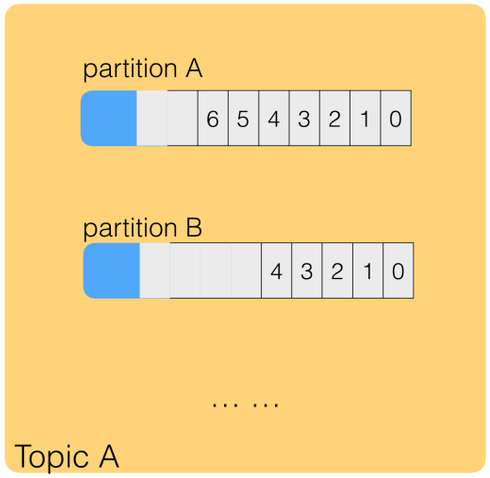
Consumer

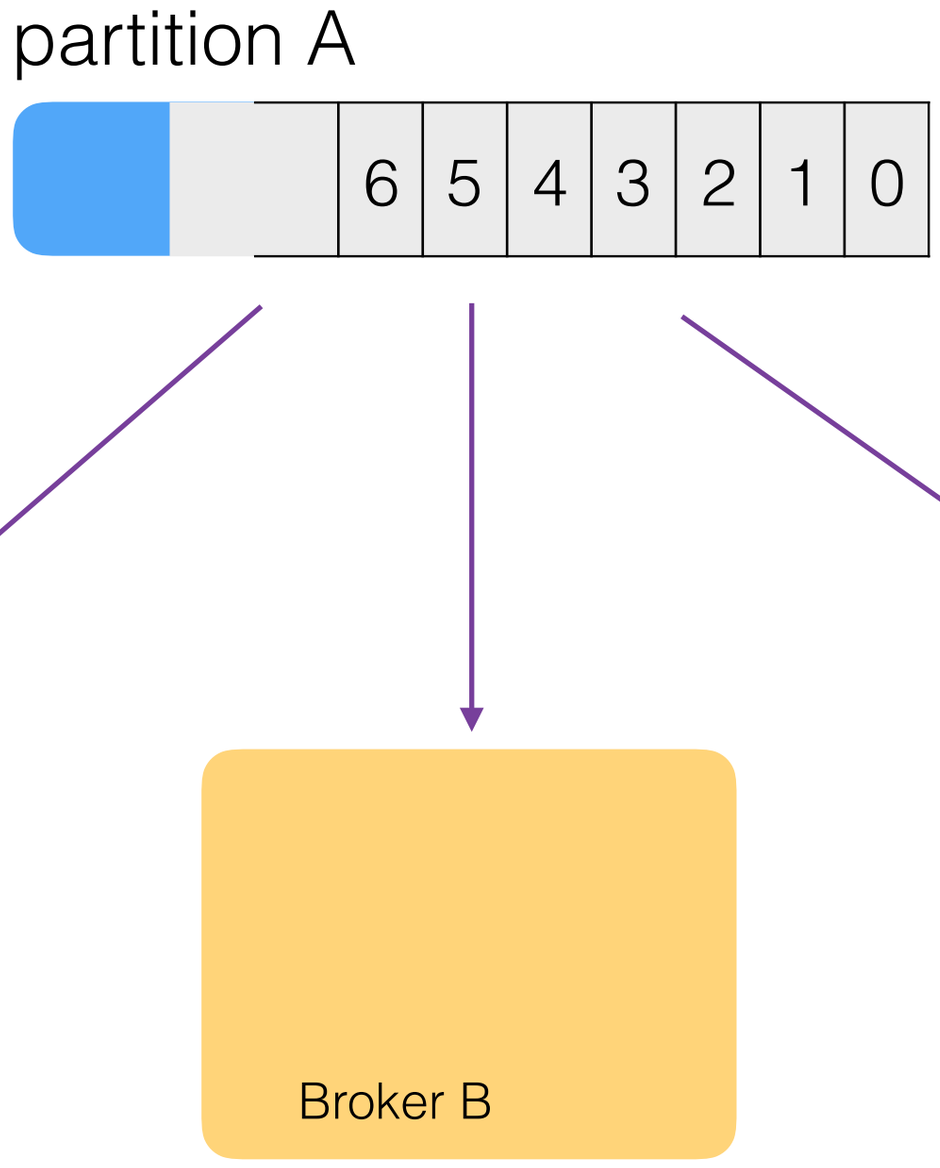
Consumer

- **问题5**：Kafka 的可靠性
  - Kafka 集群，由broker组成
  - 某个 broker 宕掉，数据是否会丢失?

- **解决办法**： replica
  - 每个 partion 都存储多份，分布在不同 broker 上
  - replica 的角色，分为 leader、follower

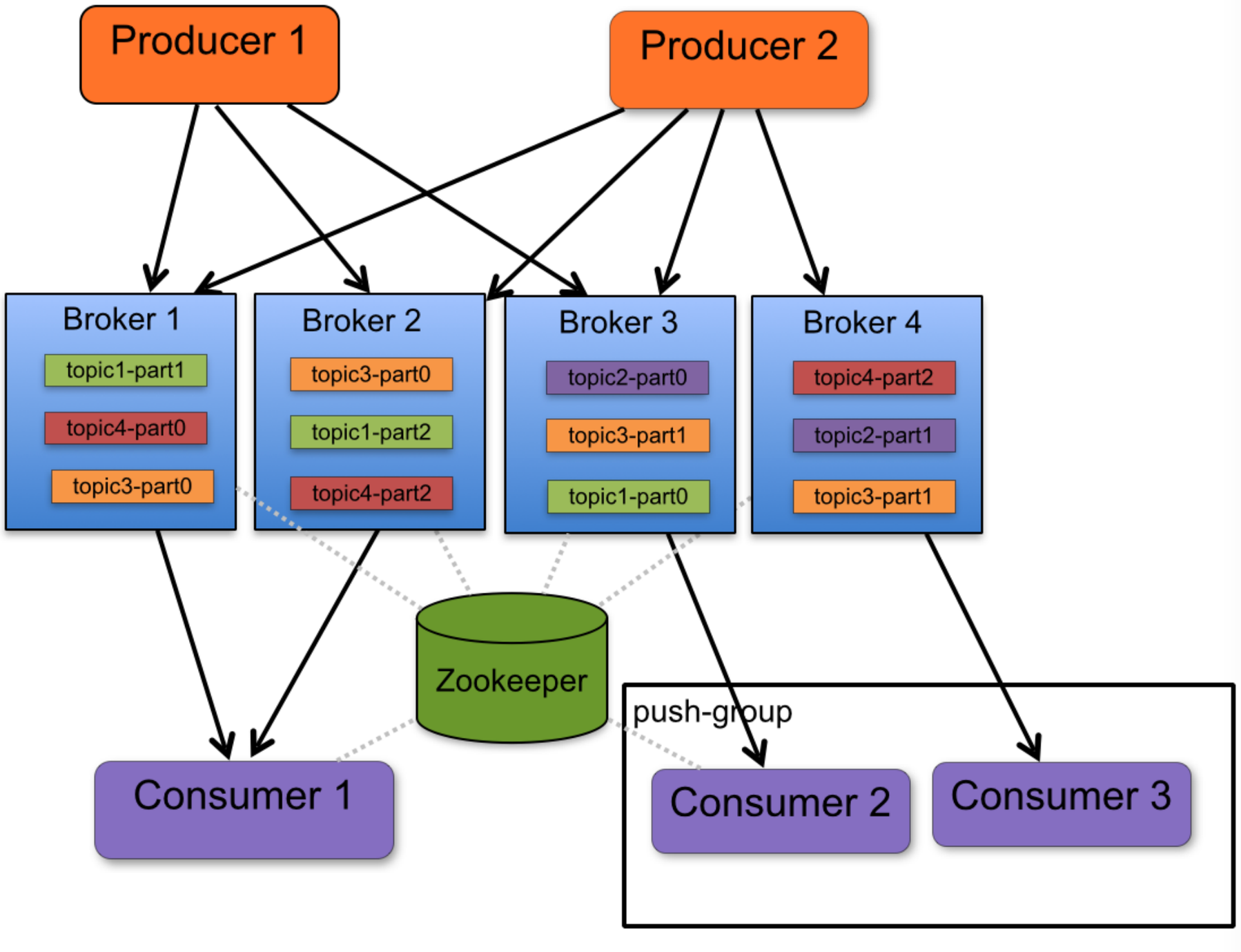



partition A
6 5 4 3 2 1 0
partition B
4 3 2 1 0
… …
Topic A


partition A
6 5 4 3 2 1 0
Broker A
Broker B
Broker C

术语整理

- **整体结构**:
  - producer: 向kafka指定topic发送数据的程序称作producer
  - consumer: 从kafka指定topic消费数据的程序称作consumer
  - broker: kafka集群中每个实例称作 个broker，由唯 id标识
- **关键术语**:
  - topic: kafka维护的消息种类,每一类消息由一个topic标识
  - consumer group：解决单播、多播问题
  - partition: 每个topic可以分成多个区，分布在不同的broker上
  - replica: 每个topic可以设置副本数，所有的副本称作replica
    - leader: 所有的副本中只有leader处理读写，其他的follower从leader同步
    - isr: replica中能够跟上leader的实例称作isr

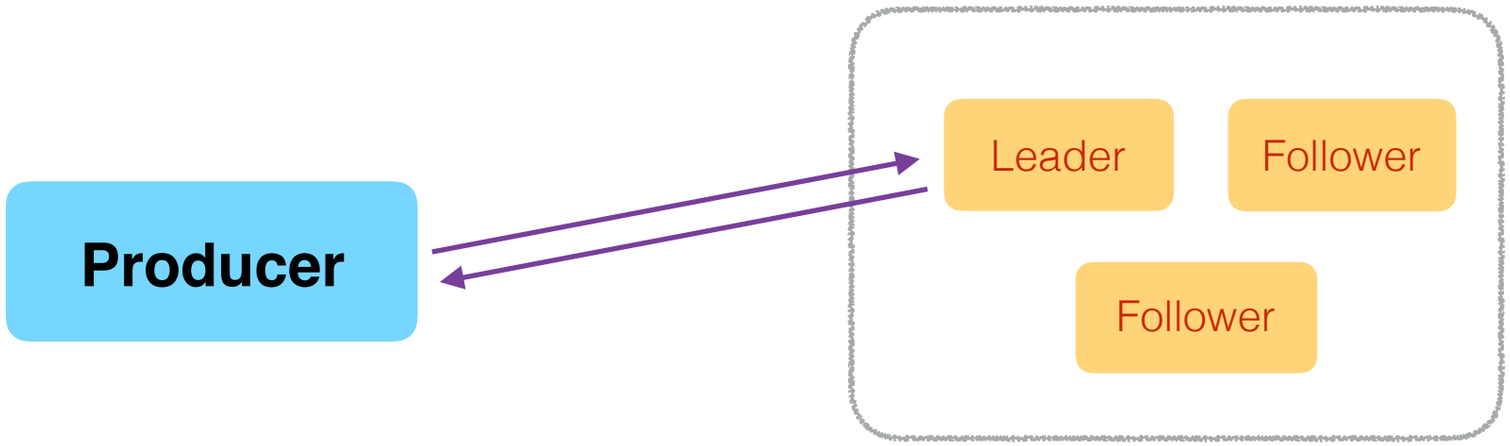

Producer 1
Producer 2
Broker 1
topic1-part1
topic4-part0
topic3-part0
Broker 2
topic3-part0
topic1-part2
topic4-part2
Broker 3
topic2-part0
topic3-part1
topic1-part0
Broker 4
topic4-part2
topic2-part1
topic3-part1
Zookeeper
push-group
Consumer 1
Consumer 2
Consumer 3

Kafka 典型特点

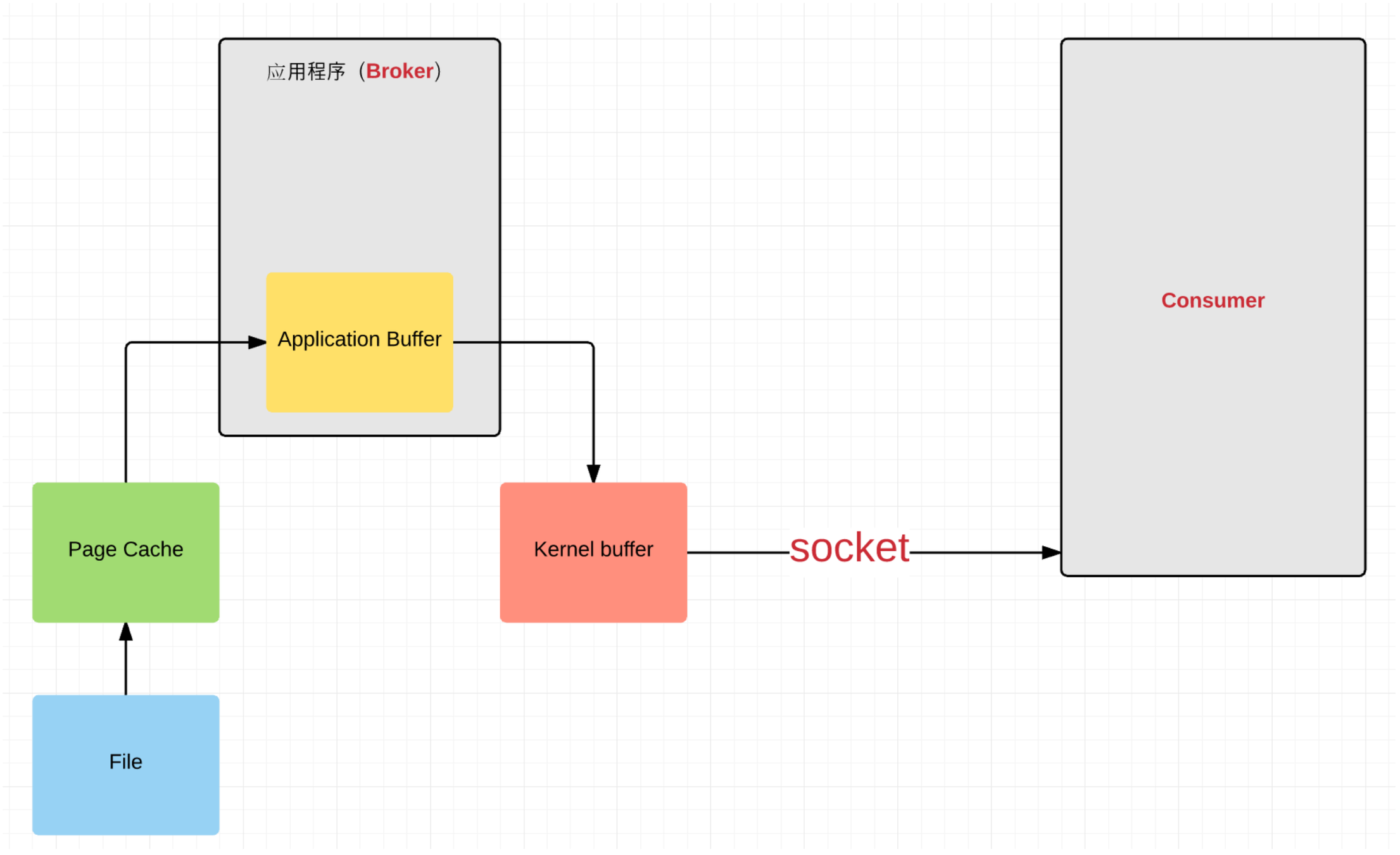
Kafka 相对其他 MQ 的典型特点?

- Producer:
  - 支持同步复制和异步复制
  - 批量发送：Nagle 策略，利用缓冲区、超时时间，合并小的数据包
    - 减少网络IO 次数
    - 增加有效 payload 比例，提升有效吞吐量
- Broker:
  - 利用sendfile系统调用，zero-copy，批量传输数据
  - 消息磁盘持久化，不在内存中 cache，充分利用磁盘顺序读写优势
    - broker 没有 cache 压力，因此，更适合支持 pull 模式
- Consumer:
  - pull 模式，broker 无状态，consumer 自己保存 offset
    - 配置过期时长（broker）
    - 多次回放数据

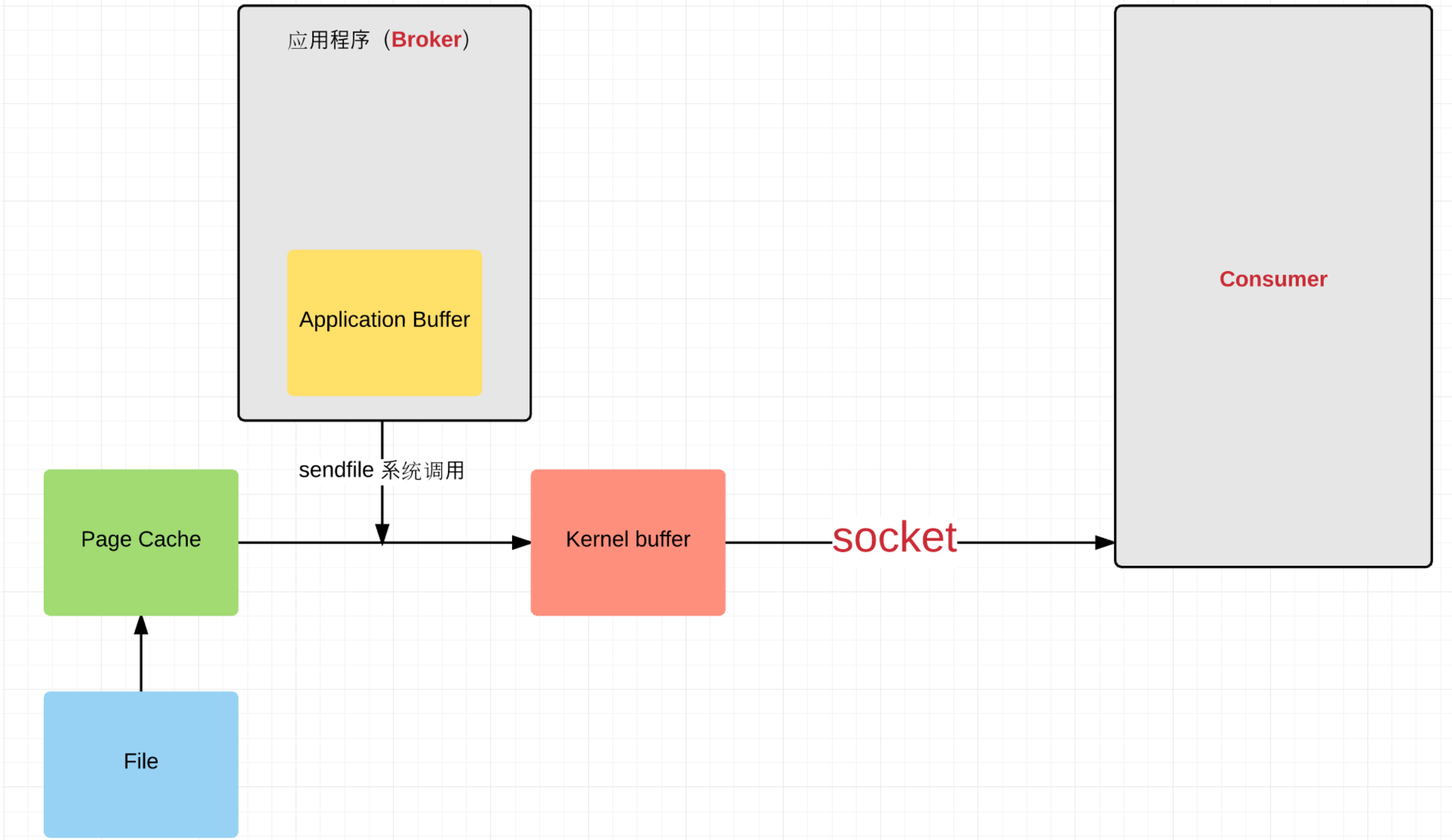
Kafka 典型特点— 同步复制、异步复制



Kafka 典型特点— sendfile 系统调用

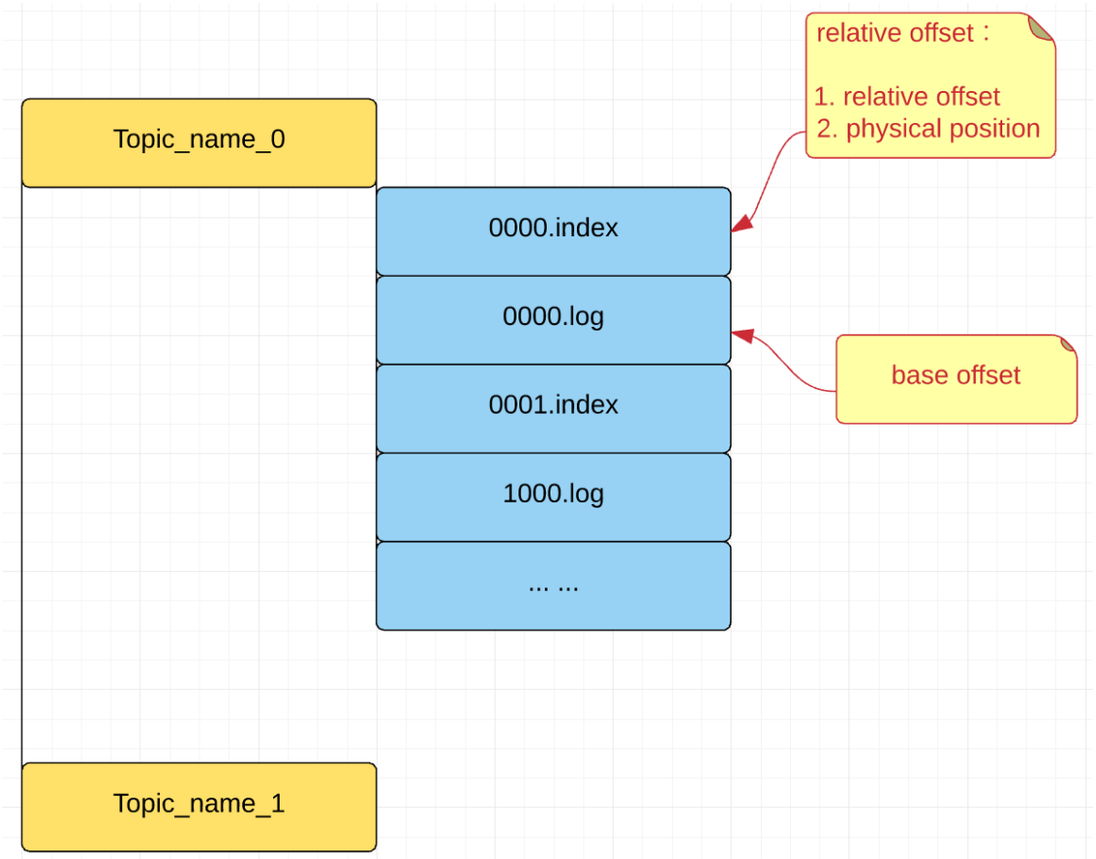


Kafka 典型特点— sendfile 系统调用



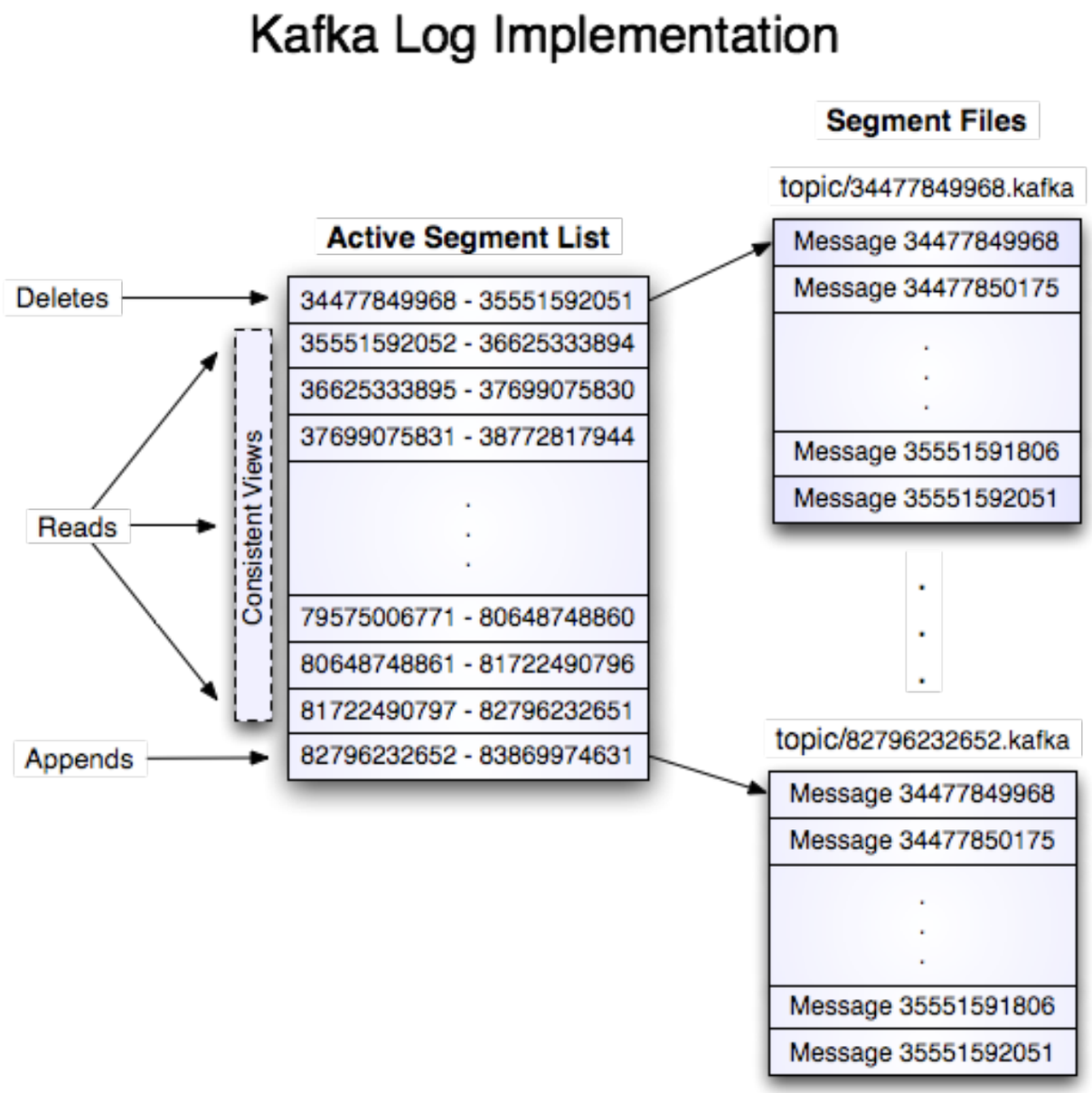
Kafka 典型特点— 磁盘持久化

- Kafka - topic - partition 的存储关系：
  - 一个 partition 对应一个文件夹
  - 一个 partition 分为多个 segment
    - segment 命名：offset.log
    - segment 对应一个 index 文件



- 核心问题：如何快速 seek 到任意 offset

Kafka 典型特点— 磁盘持久化



小结

- Kafka 设计原理?
  - topic
  - partition
  - consumer group
  - replica
- Kafka 典型特点?
  - 支持同步复制和异步复制
  - sendFile 系统调用
  - 磁盘持久化
  - pull模式
  - broker 无状态

Kafka 使用实践

- 使用实践：
  - API 简介：吞吐量 *vs.* 数据实时性
  - 并发效率：数据倾斜
- 数据丢失，典型问题：
  - 数据丢失 1：producer 侧
  - 数据丢失 2：consumer 侧
- 其他：
  - 多 kafka 集群，共享同一个 zookeeper 集群

Kafka API 简介

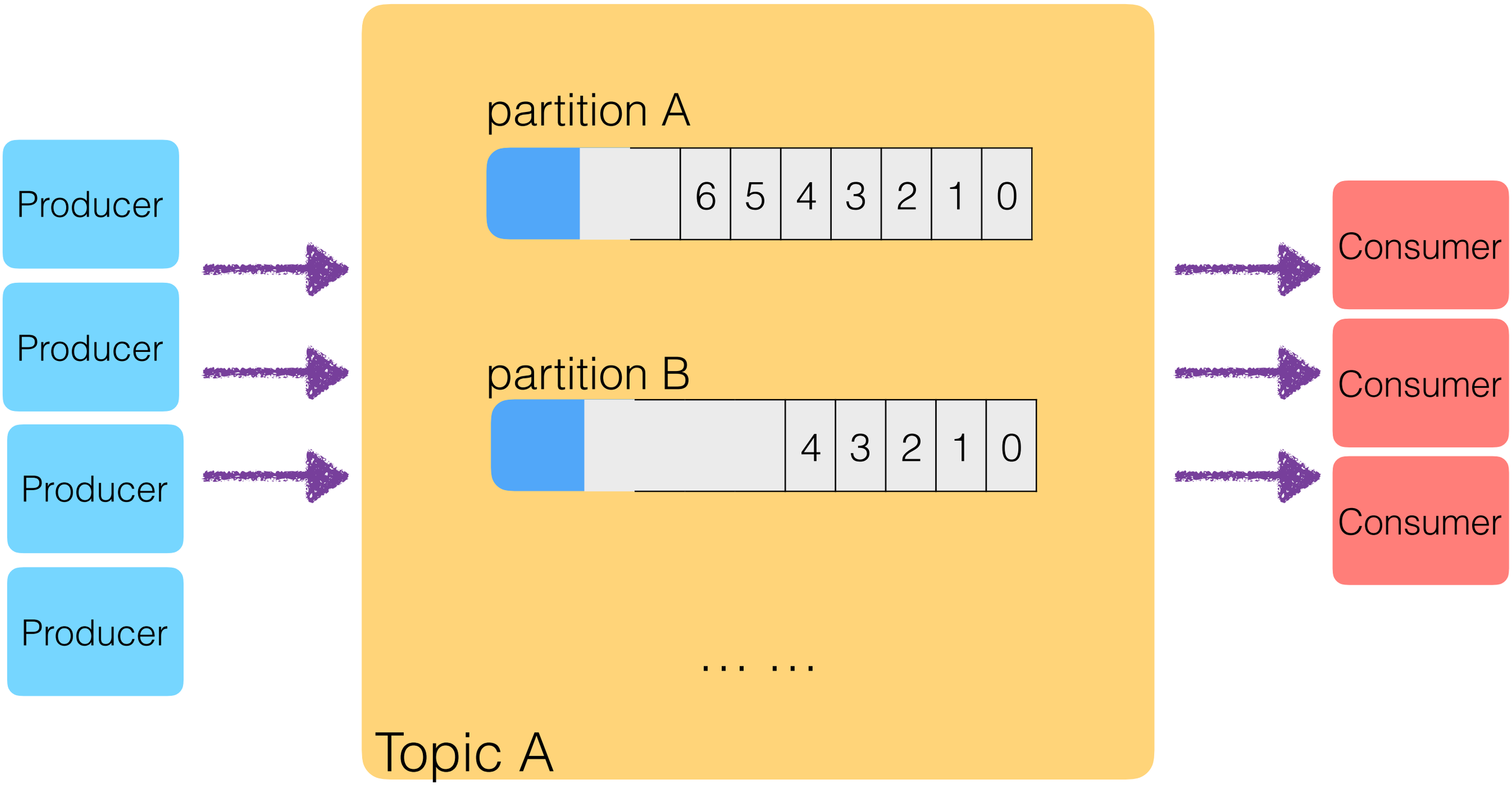
- Producer API:

```
props.put("bootstrap.servers", "localhost:4242");
props.put("acks", "all");
props.put("retries", 0);
props.put("batch.size", 16384);
props.put("linger.ms", 1);
props.put("buffer.memory", 33554432);
props.put("key.serializer", "org.apache.kafka.common.serialization.StringSerializer");
props.put("value.serializer", "org.apache.kafka.common.serialization.StringSerializer");
```

- Consumer API:
  - old High-level Consumer API
  - old Simple Consumer API
  - new Consumer API （0.9.0）

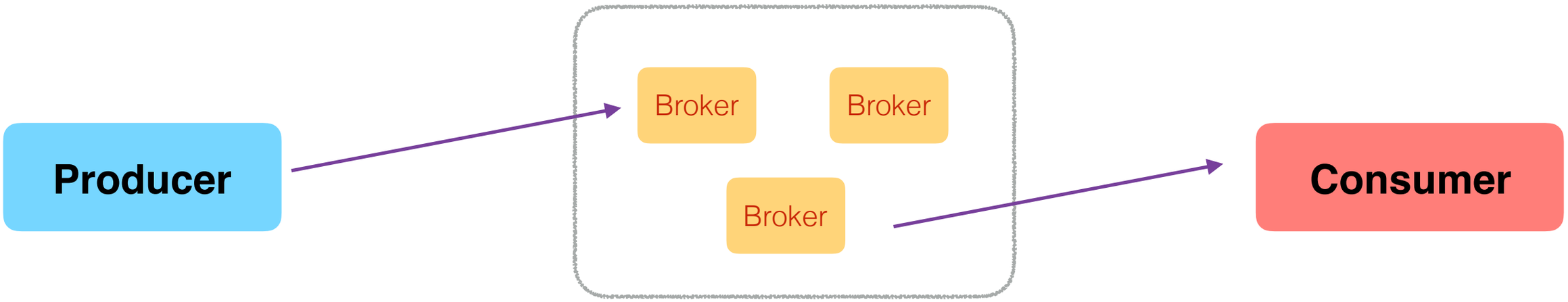
```
Properties props = new Properties();
props.put("zookeeper.connect", a_zookeeper);
props.put("group.id", a_groupId);
props.put("zookeeper.session.timeout.ms", "400");
props.put("zookeeper.sync.time.ms", "200");
props.put("auto.commit.interval.ms", "1000");
```

并发效率：数据倾斜


Producer
Producer
Producer
partition A
6 5 4 3 2 1 0
partition B
4 3 2 1 0
... ...
Topic A
Consumer
Consumer
Consumer

数据丢失典型场景

- 1. Producer 发送数据：
  - 一个 broker 收到数据后，返回 ack，但在向其余 broker 同步数据之前，当前节点宕机，数据丢失
- 2. Consumer 处理数据：
  - consumer 获取数据后，告知 broker 数据获取成功，此后，consumer 处理数据期间宕机，数据丢失


Producer
Broker
Broker
Broker
Consumer

小结

- MQ 的作用
- Kafka 的设计原理
  - Kafka 的基本原理
  - Kafka 的典型特点
- Kafka 实践

参考资料

- Top 10 Uses For A Message Queue
- Kafka: a Distributed Messaging System for Log Processing
- http://kafka.apache.org/documentation.html Kafka 0.9.0 文档
- http://kafka.apache.org/082/documentation.html Kafka 0.8.2 文档