| | |
|---|---|
| 생성일 | @2021년 3월 5일 오전 10:08 |
| 태그 | |


하이퍼레저 패브릭에서 오더러는 방대한 양의 트랜잭션을 검증하고 이들을 모아서 블록을 생성한다.

이때 트랜잭션이 많아지면 오버헤드가 걸리게 되고, 이를 위해 **MQ Solution(message queue)**인 **solo**와 **kafka**방식을 사용한다.

이중에서 **kafka**방식에 대해 알아본다.

(solo방식은 보통 테스트용으로 orderer 하나가 정렬 및 블록 생성의 모든 과정을 담당하는 방식이다.)

# Kafka란?

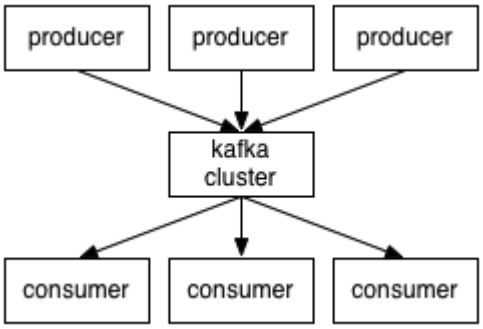
Apache Kafka는 LinkedIn에서 개발된 분산 메시징 시스템으로써 2011년에 오픈소스로 공개되었다.대용량의 실시간 로그 처리에 특화된 설계를 통하여 기존 메시징 시스템보다 우수한 TPS(Transaction Per Second)를 보여준다.

## Kafka의 기본 구성 요소의 동작

Kafka는

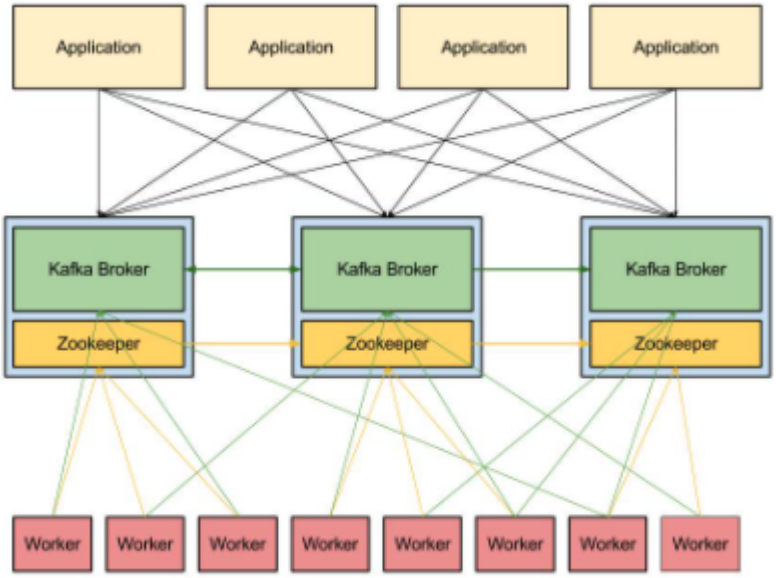
**발행-구독(publish-subscribe)**

모델을 기반으로 동작하며 크게 producer, consumer, broker로 구성된다.

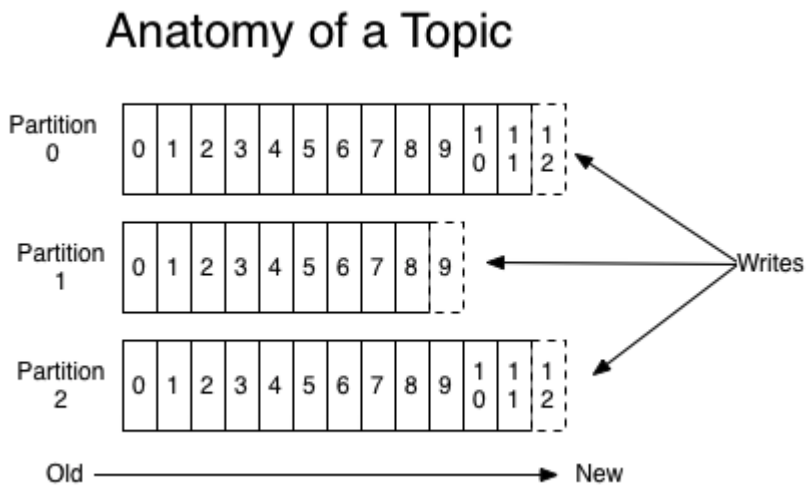



kafka의 브로커는 파티션의 집합인 topic을 기준으로 메시지를 관리한다.프로듀서는 특정 topic의 메시지를 생성한 뒤 해당 메시지를 브로커에 발행(publish)한다. 이 때 메시지를 전달할 대상을 명시하지는 않으며 브로커는 전달받은 메시지를 topic 별로 분류하여 쌓아둔다.그러면 해당 topic을 구독하는 컨슈머들이 메시지를 가져가서(pull) 처리하게 된다.

kafka는 수평적인 확장과 고가용성을 위하여 broker들이 클러스터로 구성되어 동작하도록 설계되어 있다. 클러스터 내의 broker에 대한 분산 처리는 Apache Zookeeper가 담당한다.




## Topic과 Partition




kafka의 topic은 partition이라는 단위로 쪼개져서 클러스터의 각 서버들에 분산되어 저장되고, 고가용성을 위하여 복제(replication)설정을 할 경우 파티션 단위로 각 서버들에 분산되어 복제된다.만약 장애가 발생하면 파티션 단위로 fail over가 수행된다.

기본적으로 프로듀서는 발행한 메시지가 어떤 파티션에 저장되는지 관여하지 않는다. (메시지 키와 파티셔너를 이용하여 특정 파티션으로 메시지를 전송할 수 있도록 가능은 하다.)

각 파티션은 카프카 클러스터의 브로커 중 하나가 컨트롤러가 되어 메시지를 고루 나눠가지게 된다.

특정 파티션으로 전달된 메시지에는 offset이라고 하는 숫자가 할당되어, 해당 파티션내에서 메시지를 식별하는 ID로 사용된다.오프셋값은 파티션마다 별도로 관리되므로 topic내에서 메시지를 식별할 때는 partition 번호와 offset 값을 함께 사용한다.

## Partition 분산

프로듀서가 메시지를 실제로 어떤 partiton에 전송할지는 사용자가 구현한 **partition 분배 알고리즘**에 의해 결정된다.

예를 들어 라운드-로빈 방식의 partition 분배 알고리즘을 구현하여 각 partition에 메시지를 균등하게 분배하도록 하거나, 메시지 키를 활용하여 분배할 수도 있다.

## Partition 복제

kafka는 고가용성을 제공하기 위해 파티션 데이터의 복사본을 유지할 수 있다. 몇 개의 복사본을 저장할 것인지는 replication factor로 정할 수 있다.

만약 replication factor를 N으로 설정할 경우 N개의 replica는 1개의 leader와 N-1개의 follower로 구성된다.

각 partition에 대한 읽기와 쓰기는 모두 leader에서 이루어지고, follower는 단순히 leader를 복제하기만 한다.

만약 leader에 장애가 발생하면 follower 중 하나가 새로운 leader가 된다.

## Producer, Consumer, Consumer Group

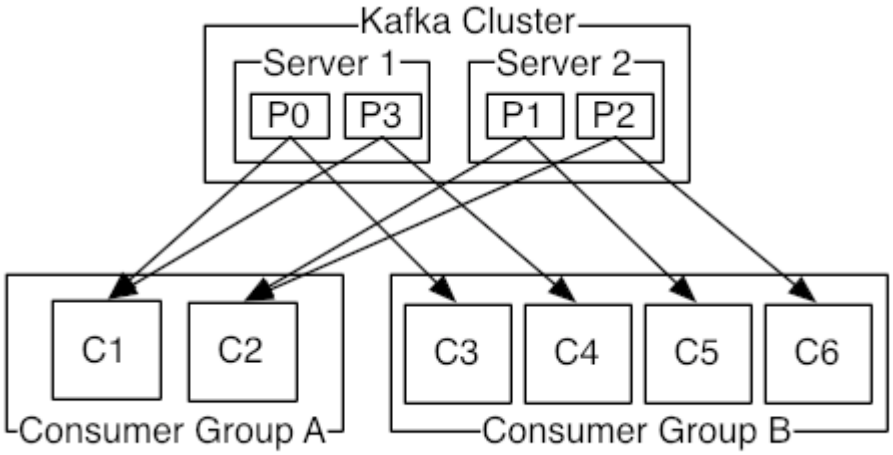
kafka의 클라이언트는 기본적으로 **프로듀서**와 **컨슈머**라는 두 가지 분류가 존재한다.

**프로듀서**는 메시지를 생성하여 카프카에 전달하는 클라이언트를 의미한다.

**컨슈머**는 메시지를 카프카로부터 읽어가는 클라이언트이다.

**컨슈머 그룹**은 컨슈머들로 형성되어 진다.카프카의 토픽은 컨슈머 그룹 단위로 구독되어진다.토픽의 파티션은 그룹 당 하나의 컨슈머만이 소비할 수 있는데, 다시 말해 **같은 컨슈머 그룹에 속한 컨슈머들이 동시에 동일한 파티션에서 메시지를 읽어갈 수 없다.**

Consumer group을 구성하는 consumer의 수가 partition의 수보다 작으면 하나의 consumer가 여러 개의 partition을 소유하게 되고, 반대로 consumer의 수가 partition의 수보다 많으면 여분의 consumer는 메시지를 처리하지 않게되므로 **partition 개수와 consumer 수의 적절한 설정이 필요**하다.




## 파일 시스템을 활용한 고성능 디자인

Kafka는 기존 메시징 시스템과 달리 메시지를 메모리대신 **파일 시스템***에 쌓아두고 관리하여 **영속성**을 얻는다. 즉 나중에 다시 특정 메시지를 소비하고 싶을 때 파일 시스템에 저장된 메시지를 읽어서 컨슈머에게 전송할 수 있다.