Queue Messaging Systems (QMS)

28th October, 2016

Arjun Singhvi
Overview

• Motivation
• Initial Use Case
• QMS Architecture Choices
• Deep dive into Kafka
Motivation

• Data Explosion in the last 10 years due to emergence of IoT
Motivation

• Sheer volume of these datasets lead to the emergence of Big Data systems
Motivation

• Need an efficient way to deal with this heterogeneous data coming in from different sources

• Need to process the same data in various ways
  – Real-time analytics
  – Batch analytics
Motivation - Pre-QMS Era

- Custom data pipeline for each unique source-destination pair
- Does not scale well
Motivation

• Efficiently aggregate all types of data and provide at –
  – High throughput
  – Low latency
  – Real time

• Lead to emergence of various QMS
QMS Architecture Choices

• Message Queues
  – ActiveMQ
  – RabbitMQ

• Publish-Subscribe Systems
  – Kafka
  – Kestrel
Initial Use Case

• Mainly used in the data processing pipelines for data ingestion or aggregation
• Envisioned mainly to be used at the beginning or end of a data processing pipeline
• Example –
  – Incoming data from various sensors
  – Ingest this data into a streaming system for real-time analytics or a distributed file system for batch analytics
Kafka: Introduction

- Producers – Publish data streams to Kafka cluster
- Consumers – Subscribe to one or more data streams
- Kafka Cluster – Distributed log of data over serves known as brokers
Kafka: Introduction

Broker-1  Broker-2  Broker-n

Kafka Cluster
Kafka: Topics

- Category to which the messages are published
- For each topic, the Kafka cluster maintains a partitioned log
Kafka: Partitions

Kafka Cluster

Broker-1

Broker-2

Broker-n

Leader

Followers
Kafka: Partitions

- Ordered, immutable sequence of records that is continually appended to
- Each record is associated with a sequential id number called as offset
- Partitions are distributed over the servers in Kafka
- Each partition is replicated for fault tolerance
- Partition and replicas follow the leader-followers pattern
Kafka: Producer

• Publishes data to topics of their choice
• In fact also responsible for choosing which record to assign to which partition within the topic
• Think of publishers as data sources
Kafka: Consumer

- Consumer Group maps to a logical subscriber
- Each group consists of consumer instances for scalability and fault tolerance
- Advantages of both queuing as well as publish-subscribe
KaJa: ZooKeeper

- ZooKeeper is a distributed, open-source coordination service for distributed applications
- Kafka uses it to coordinate between the producers, consumers and brokers
- ZooKeeper stores metadata
  - List of brokers
  - List of consumers and their offsets
  - List of producers
- ZooKeeper runs several algorithms
  - Consumer registration algorithm
  - Consumer rebalancing algorithm
Kafka: Design Choices

• Push vs. Pull model for Consumers
  – Push model
    • Challenging for the broker to deal with diverse consumers as it controls the rate at which data is transferred
    • Need to decide whether to send a message immediately for accumulate more data and send
  – Pull model
    • In case broker has no data, consumer may end up busy-waiting for data to arrive
Kafka: Ordering Guarantees

• Messages sent by a producer to a particular topic partition will be appended in the order they are sent

• Consumer instance sees records in the order they are stored in the log

• Provides a total order over records within a partition, not between different partitions in a topic. Per-partition ordering combined with the ability to partition data by key is sufficient for most applications.
Kafka: Fault Tolerance

- Replicates partitions for fault tolerance
- Kafka makes a message available for consumption only after all the replicas acknowledge to the leader replica a successful write
- Implies that a message may not be immediately available for consumption
Kafka: Producer Batching
Kafka: Limitations

- Kafka follows the pattern of active-backup with the notion of “leader” partition replica and “follower” partition replicas
- Kafka stores a partition on a single disk

*DistributedLog* from Twitter claims to solve these issues
Kafka: In Real World

• 50+ companies are using Kafka as their primary infrastructure to handle data and make it available in real-time
KaJa: In Real World

- Netflix uses Kafka for data collection and buffering so that it can be used by downstream systems
Kafka: In Real World

- Uber uses Kafka for real-time business driven decisions (For example – Surge)
Kafka: Only for data ingestion?

- Samza is a distributed stream processing framework
- It uses Kafka for data management layer for the streaming system
- Kafka being used even within a data processing pipeline
A Samza job consists of
- Kafka consumer, an event loop that calls application code to process incoming messages
- Kafka producer that sends output messages back to Kafka
Summary – QMS Era

• QMS are an essential part of the entire big data processing pipeline
• No longer just used for data ingestion and aggregation