И. А. Михневич

ПРОЕКТИРОВАНИЕ ВЫСОКОНАГРУЖЕННЫХ ПРИЛОЖЕНИЙ С APACHE KAFKA

Рассматриваются ключевые механизмы масштабирования, обеспечения отказоустойчивости и организации потоковой обработки данных в event-driven архитектурах.

Введение

Для решения задачи обмена больших данных в высоконагруженных распределённых системах предлагается использовать брокер сообщений Apache Kafka — распределённой платформы потоковой передачи данных, ориентированной на высокую пропускную способность и отказоустойчивость, работа которой основана на асинхронном взаимодействии системных компонентов, обеспечивая гарантии доставки, производительность системы, лёгкость масштабирования и надёжность хранения данных. В работе рассматриваются подходы к проектированию высоконагруженных приложений с использованием Kafka как центральной шины событий.

I. Брокер сообщений Apache Kafka

Apache Kafka представляет собой распределённый журнал сообщений, организованный в виде топиков, разделённых на партиции. Главными компонентами Apache Kafka являются consumer, producer, message и broker. Kafka построена по принципу "Глупая шина, умные консьюмеры"[1], обеспечивая бизнес логику(трансформация, маршрутизация, фильтрация) на стороне консьюмеров или продьюсеров, в то время как сама центральная шина отвечает за надёжное хранение и доставку сообщений. Использование Kafka как шины событий позволяет отделить источники данных от потребителей.

II. Обеспечение высокой производительности

Производительность систем, построенных на Kafka, определяется несколькими факторами: количеством партиций, конфигурацией брокеров, характеристиками сети и эффективностью потребителей. Партицирование позволяет позволяет повысить пропускную способность обработки сообщений более чем в 3 раза. Однако чрезмерное увеличение числа партиций приводит к росту накладных расходов на координацию и может снижать общую производительность.

III. Масштабирование систем на Kafka

Масштабирование является ключевым требованием для высоконагруженных приложений. Kafka обеспечивает горизонтальное масштабирование за счёт распределения данных по партициям и брокерам. Увеличение числа партиций позволяет повысить параллелизм обработки. Добавление брокеров распределяет общий объём хранимых данных. Масштабирование групп потребителей позволяет обрабатывать данные быстрее без изменения продюсеров.

IV. Обеспечение отказоустойчивости

Отказоустойчивость является критически важным свойством высоконагруженных систем. Kafka обеспечивает устойчивость к сбоям за счёт репликации партиций, позволяющее просто автоматически перераспределить лидера при сбоях, и распределённого консенсуса. Параметры ack позволяют определить уровенрь доставки. Также при выходе из строя Брокера-контроллера, система автоматически производит перевыбор нового, исходя из скорости ответа от других брокеров системы, что позволяет определить более производительный из них.

V. Хранение данных в Apache Kafka

Kafka рассматривает топики, хранящиеся на брокерах, как распределённые журналы, в которых сообщения сохраняются на диске. Каждый топик разбивается на сегменты фиксированного размера, что упрощает управление файлами и ускоряет операции удаления. Данные могут храниться по времени (retention.ms) или по объёму (retention.bytes), что позволяет адаптировать систему под требования конкретного приложения. Kafka использует механизм передачи данных напрямую из файловой системы в сеть, что снижает нагрузку на CPU и повышает пропускную способность.

VI. Выводы

Использование Apache Kafka позволяет обеспечить масштабируемость, отказоустойчивость и гибкость архитектуры.

Список литературы

1. Apache Kafka: системный подход [Электронный ресурс] / Systems Education. – Режим доступа: https://systems.education/kafka. – Дата доступа: 29.03.2026.

Михневич Иван Александрович, студент 3 курса ФИТиУ БГУИР, ivanmichnevih2@gmail.com.

Научный руководитель: Радченко Анастасия Дмитриевна, ассистент кафедры ВМиП БГУИР, a.radchenko@bsuir