Kafka

Kafka: ????????

Сообщение

- Ключ - метаданные для управления записью сообщения в разделы
- Схема - способ разобрать сообщение. Может быть JSON, XML, Apache Avro
- Смещение - точка считывания, добавляется Kafka

Тема (топик) - раздел - сообщение. Упорядочиваются в пределах раздела.

Производители

Потребители

Могут объединяться в группы. Чтение каждого раздела только одним членом группы. Принадлежность - какой раздел какому потребителю.

Брокер

Отдельный сервер Kafka - брокер. Объединяются в кластер, один из брокеров - контроллер. Если раздел нескольким брокерам, то происходит репликация. Основной брокер - ведущий. Механизмы репликации только в пределах одного кластера. MirrorMaker - репликация между кластерами.

broker.id	Целочисленный идентификатор, с 0, уникальный.
listeners	протокол://имя_хоста:порт Перечисляются через запятую. Имя хоста: конкретный ip - соответствующий интерфейс, 0.0.0.0 - все интерфейсы, не указан - интерфейс по умолчанию. Протокол: PLAINTEXT, SSL Если порт менее 1024 то Kafka от имени root,
log.dirs	Директории размещения логов.

Python: kafka-python

?????????

Kafka с web интерфейсом:

```
services:
  kafka:
    image: apache/kafka:4.2.0
    container_name: kafka
    hostname: kafka
    restart: unless-stopped

    ports:
      - "9092:9092"   # INTERNAL (docker network)
      - "29092:29092" # EXTERNAL (LAN access)

    environment:
      # =========================
      # KRaft
      # =========================
      KAFKA_NODE_ID: 1
      KAFKA_PROCESS_ROLES: broker,controller
      KAFKA_CONTROLLER_QUORUM_VOTERS: 1@kafka:9093

      # =========================
      # Listeners
      # =========================
      KAFKA_LISTENERS: INTERNAL://:9092,EXTERNAL://:29092,CONTROLLER://:9093
      KAFKA_ADVERTISED_LISTENERS: INTERNAL://kafka:9092,EXTERNAL://192.168.1.195:29092
      KAFKA_LISTENER_SECURITY_PROTOCOL_MAP:
INTERNAL:PLAINTEXT,EXTERNAL:PLAINTEXT,CONTROLLER:PLAINTEXT
      KAFKA_INTER_BROKER_LISTENER_NAME: INTERNAL
      KAFKA_CONTROLLER_LISTENER_NAMES: CONTROLLER

      # =========================
      # Single-node safety
      # =========================
      KAFKA_OFFSETS_TOPIC_REPLICATION_FACTOR: 1
      KAFKA_TRANSACTION_STATE_LOG_REPLICATION_FACTOR: 1
```

```
      KAFKA_TRANSACTION_STATE_LOG_MIN_ISR: 1
      KAFKA_GROUP_INITIAL_REBALANCE_DELAY_MS: 0

      # Optional tuning
      KAFKA_AUTO_CREATE_TOPICS_ENABLE: "true"

    volumes:
      - ./kafka_data:/var/lib/kafka/data

  kafka-ui:
    image: provectuslabs/kafka-ui:latest
    container_name: kafka-ui
    restart: unless-stopped

    ports:
      - "8080:8080"

    environment:
      KAFKA_CLUSTERS_0_NAME: local
      KAFKA_CLUSTERS_0_BOOTSTRAPSERVERS: kafka:9092

    depends_on:
      - kafka
```

На https://kafka.apache.org/quickstart/ есть информация по запуску без docker

Проверка внутренними средствами

Создание темы

```
docker exec -it kafka /opt/kafka/bin/kafka-topics.sh \
  --create \
  --topic demo \
  --bootstrap-server kafka:9092 \
  --partitions 3 \
  --replication-factor 1
```

Запуск продюсера (сервер на 1.120)

```
docker exec -it kafka /opt/kafka/bin/kafka-console-producer.sh \
  --topic demo \
```

```
  --bootstrap-server 192.168.1.120:29092
```

Запуск консъюмера (сервер на 1.120)

```
docker exec -it kafka /opt/kafka/bin/kafka-console-consumer.sh \
  --topic demo \
  --from-beginning \
  --bootstrap-server 192.168.1.120:29092
```

.