Apache Kafka

Međuprocesna komunikacija i komunikacija u realnom vremenu

Apache Kafka

- *Distributed Streaming Platform*
- Počiva na PUB-SUB mehanizmu razmene poruka
- Skladišti poruke na robustan način
- Omogućava
  - Skaliranje
  - Brz protok poruka
  - Replikaciju poruka

Apache Kafka - osnovni koncepti

- Broker - posrednik u komunickaciji; Kafka klaster se sastoji iz više brokera
- Producer - klijentska aplikacija koja šalje poruke
- Consumer - klijentska aplikacija koja prima i obrađuje poruke
- Topic - tema na koju se šalju poruke
- Partition - topici podeljeni na particije što omogućava paralelizaciju obrade
- Consumer Group - grupa obrađivača; poruka dolazi tačno jednom članu
- Leader & Replica
  - Leader - broker na koji se šalju poruke i koji je zadužen za održavanje replikacija
- Offset - za svakog *consumer*-a topika postoji *offset* tj. pamti se do kog rednog broja su poruke pročitane

Instalacija

- Preduslov - JRE
- Download
  - `wget https://dlcdn.apache.org/kafka/4.1.1/kafka_2.13-4.1.1.tgz`
  - `tar -xzf kafka_2.13-4.1.1.tgz`
  - `rm kafka_2.13-4.1.1.tgz`
  - `cd kafka_2.13-4.1.1/`

Pokretanje Kafka klastera

- Pokretanje Kafka brokera:
  - Generisanje Cluster ID-ja
    - KAFKA_CLUSTER_ID="$(bin/kafka-storage.sh random-uuid)"
  - Formatiranje skladišta podataka
    - bin/kafka-storage.sh format -t $KAFKA_CLUSTER_ID -c config/kraft/server.properties --standalone
  - `bin/kafka-server-start.sh config/server.properties`
  - Sluša na portu 9092

Kreiranje topika

- Kreiranje *test* topika sa jednom particijom i faktorom replikacije 1
  - `bin/kafka-topics.sh --create --bootstrap-server localhost:9092 --replication-factor 1 --partitions 1 --topic test`
- Prikaz svih topika
  - `bin/kafka-topics.sh --list --bootstrap-server localhost:9092`

Razmena poruka

- Slanje poruka na *test* topik
  - `bin/kafka-console-producer.sh --bootstrap-server localhost:9092 --topic test --property parse.key=true --property key.separator=":"`
- Prijem poruka sa *test* topika
  - `bin/kafka-console-consumer.sh --bootstrap-server localhost:9092 --topic test --from-beginning  --property print.timestamp=true --property print.key=true --property print.value=true --property print.offset=true --property key.separator=”:”`
  - Moguće je specificirati još
    - Offset
    - Consumer group id
    - Partition
    - Key-deserializer
    - Value-deserializer
    - ...

Kafka multi-broker - konfiguracija

- Kreirati dodatna dva Kafka server.properties fajla
  - `cp config/server.properties config/server-1.properties`
  - `cp config/server.properties config/server-2.properties`
  - `cp config/server.properties config/server-3.properties`
- Promeniti sledeća podešavanja
  - `node.id` (1, 2, 3),
  - `Listeners i advertised.listeners:`
    - `PLAINTEXT://:9094,CONTROLLER://:9095`
    - `PLAINTEXT://:9096,CONTROLLER://:9097`
    - `PLAINTEXT://:9098,CONTROLLER://:9099`
  - `controller.quorum.bootstrap.servers` - 9095, 9097, 9099
  - `advertised.listeners` redom 9094, 9095 i 9096, 9097,
  - `log.dirs` (Redom `/tmp/kafka-1,/tmp/kafka-2 i /tmp/kafka-3`)

Kafka multi-broker - konfiguracija

- Pokrenuti kreiranje random ID-jeva
  - ```
    KAFKA_CLUSTER_ID="$(bin/kafka-storage.sh random-uuid)" &&
    DIR_ID_1=$(bin/kafka-storage.sh random-uuid) &&
    DIR_ID_2=$(bin/kafka-storage.sh random-uuid) &&
    DIR_ID_3=$(bin/kafka-storage.sh random-uuid)
    ```
- Pokrenuti formatiranje direktorijuma (za sva tri properties fajla)
  - ```
    bin/kafka-storage.sh format -t $KAFKA_CLUSTER_ID -c
    config/server-n.properties --initial-controllers
    "1@localhost:9095:$DIR_ID_1,2@localhost:9097:$DIR_ID_2,3@localhost:9
    099:$DIR_ID_3"
    ```

Kafka multi-broker

- Pokrenuti brokere
  - `bin/kafka-server-start.sh config/server-1.properties`
  - `bin/kafka-server-start.sh config/server-2.properties`
  - `bin/kafka-server-start.sh config/server-3.properties`

Rad sa repliciranim topikom

- Ponovo pokrenuti “ubijenog” brokera
- Kreiranje repliciranog topika
  - `bin/kafka-topics.sh --create --bootstrap-server localhost:9094 --replication-factor 3 --partitions 1 --topic my-replicated-topic`
- Prikaz strukture repliciranog topika
  - `bin/kafka-topics.sh --describe --bootstrap-server localhost:9094 --topic my-replicated-topic`
- Slanje poruka
  - `bin/kafka-console-producer.sh --bootstrap-server localhost:9094 --topic my-replicated-topic`
- Prijem poruka
  - `bin/kafka-console-consumer.sh --bootstrap-server localhost:9094 --from-beginning --topic my-replicated-topic`

Simulacija otkaza

- Namerno stopiranje brokera 2
  - `ps aux | grep server-n.properties (n zavisi od trenutnol leader-a)`
  - `kill -9 PID`
- Prikaz strukture repliciranog topika
  - `bin/kafka-topics.sh --describe --bootstrap-server localhost:n --topic my-replicated-topic (n port nekog od dostupnih)`
- Poruke je i dalje moguće pročitati iako je *leader* broker stopiran
  - `bin/kafka-console-consumer.sh --bootstrap-server localhost:n --from-beginning --topic my-replicated-topic`

Consumer grupe

- Kreiranje topika sa 3 particije i replikacijom
  - `bin/kafka-topics.sh --create --bootstrap-server localhost:9094 --replication-factor 3 --partitions 3 --topic my-topic`
- Slanje poruka
  - `bin/kafka-console-producer.sh --bootstrap-server localhost:9094 --topic my-topic`
- Pokretanje više consumer-a s istom grupom
  - `bin/kafka-console-consumer.sh --bootstrap-server localhost:9094 --topic my-topic --consumer-property group.id=g1`
    - `bin/kafka-topics.sh --describe --bootstrap-server localhost:9094 --topic my-topic`
    - `bin/kafka-consumer-groups.sh --bootstrap-server localhost:9094 --list`
    - `bin/kafka-consumer-groups.sh --bootstrap-server localhost:9094 --describe --group g1`

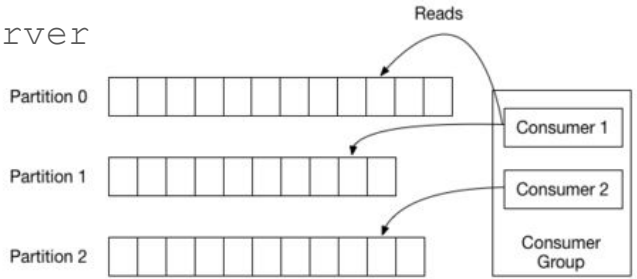
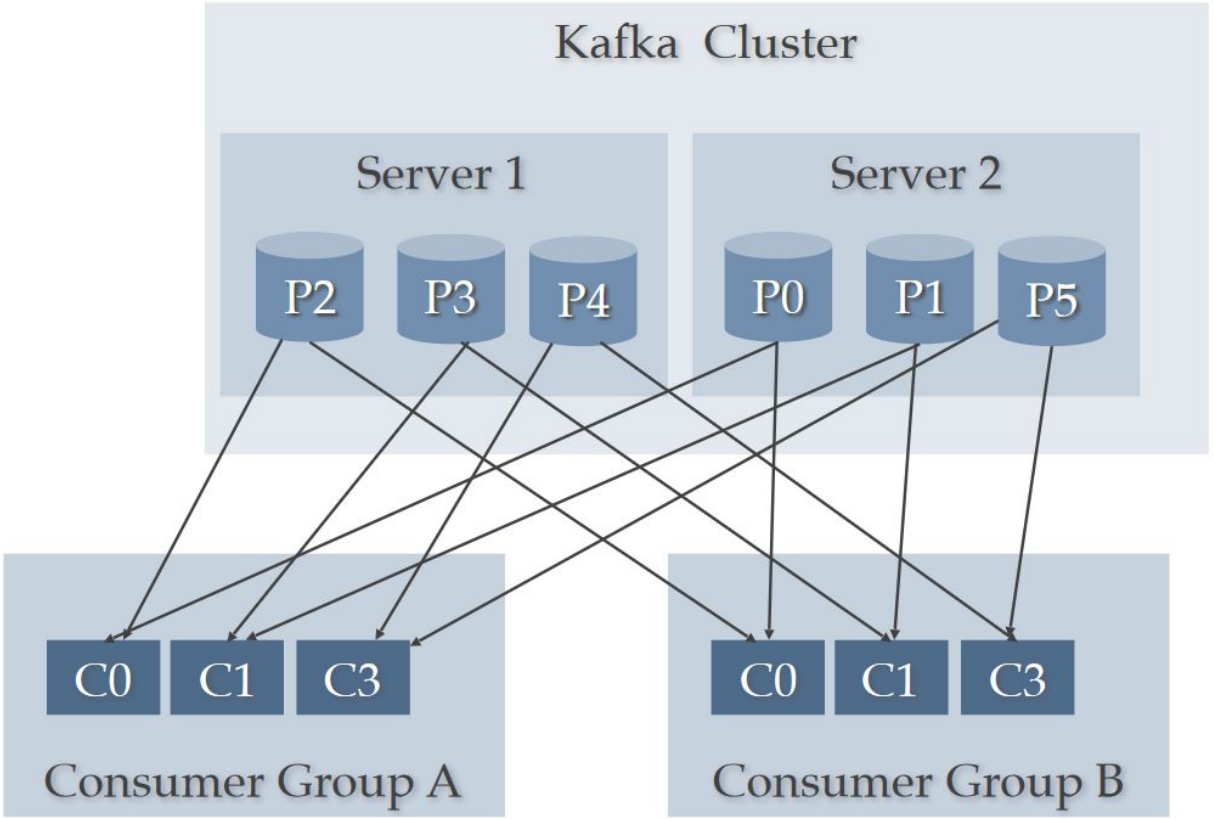
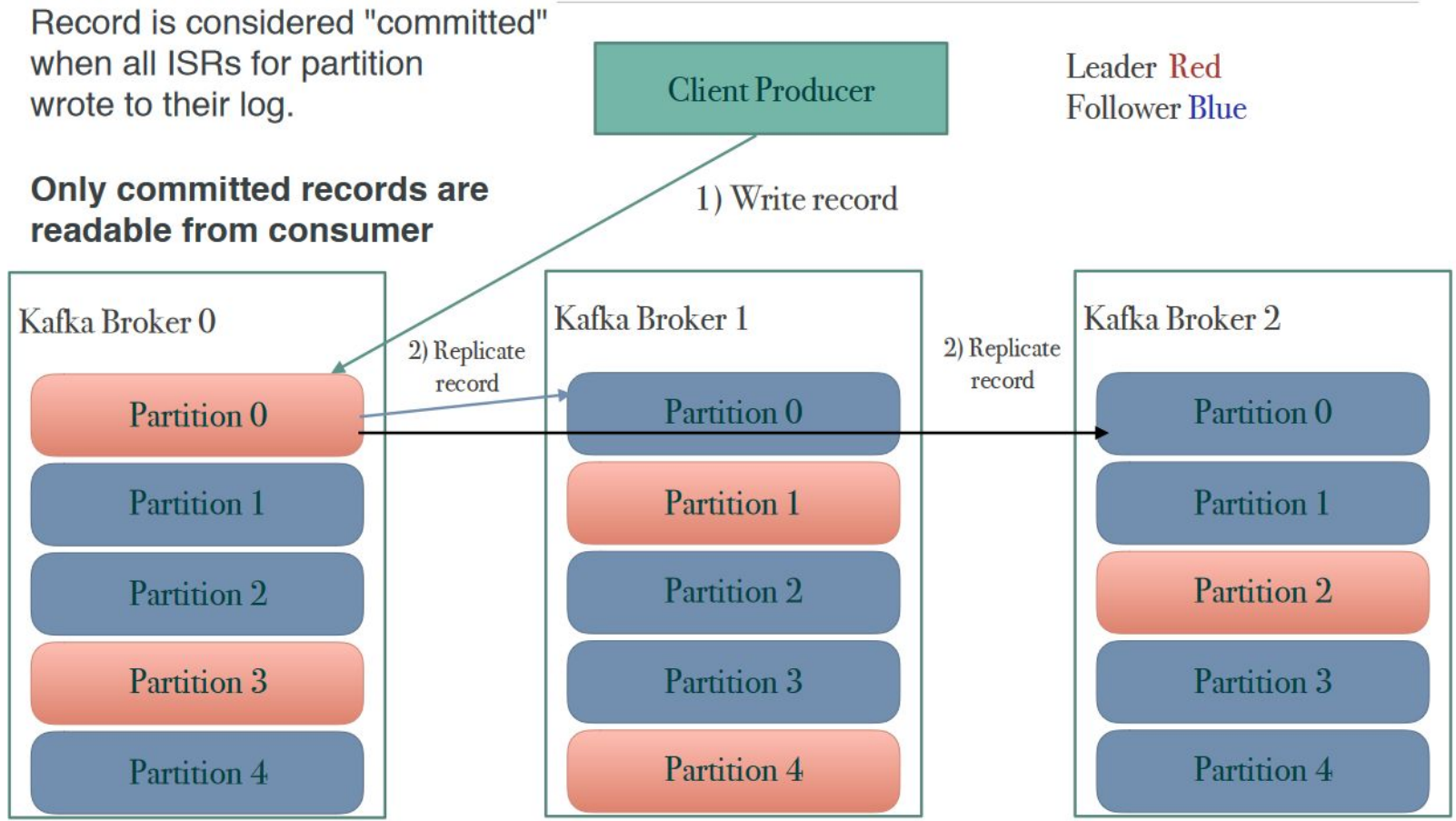


Figure 1: Consumer Group

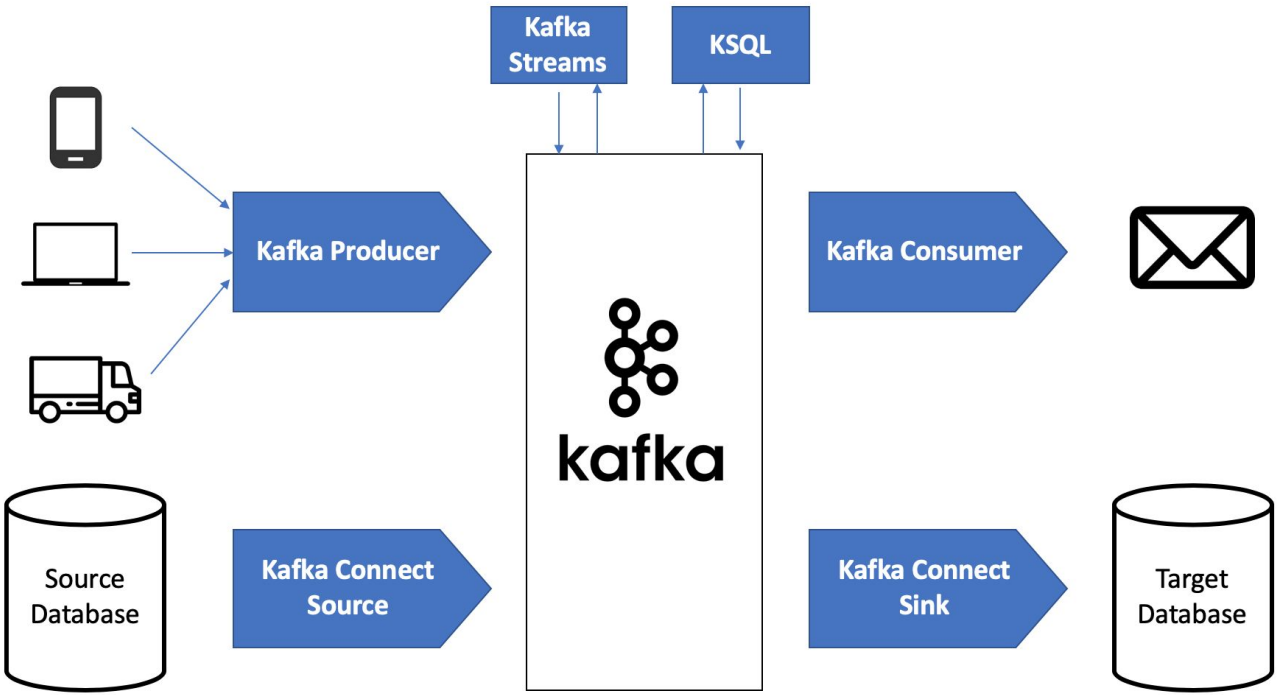
Consumer grupe



Replikacija i ditribucija topika



Kafka APIs


Kafka Streams
KSQL
Kafka Producer
kafka
Kafka Consumer
Source Database
Kafka Connect Source
Kafka Connect Sink
Target Database

Primer realne Kafka Consumer aplikacije

- Zasnovana na *confluent-kafka* biblioteci
  - Dokumentacija: https://docs.confluent.io/kafka-clients/python/current/overview.html
- Primer “dokerizovane” Kafke
- Za slanje poruka iskoristiti:

  ```
  ocker exec -it simple-kafka-consumer-consumer-1 bash -c "python produce_event.py"
  ```

  - Primer jednostavne producer aplikacije
- Za pregled poruka iskoristiti kafka-ui

Kafka klijenti

- Pored izvornog API-ja u Javi postoje i klijentske biblioteke za većinu programskih jezika opšte namene
  - C/C++, Python, golang, Node.js, Ruby, .NET, …
- Više o dostupnim implementacijama klijentskih Kafka biblioteka na sledećem linku
  - https://cwiki.apache.org/confluence/display/KAFKA/Clients

Testiranje klastera na opterećenje

- Locust load testing framework
  - Dokumentacija - https://docs.locust.io/en/stable/
  - Dostupan na: http://localhost:8091/
- `docker-compose up --scale locust-worker=4`
- `docker exec -it example_kafka_consumer_example-consumer_1 bash -c "/opt/kafka_2.11-1.1.1/bin/kafka-consumer-groups.sh -bootstrap-server kafka:9092 --describe --group example-consumer"`
- Isprobati različit broj consumer aplikacija u grupi, različit broj particija topika, te proučiti uticaj na mogućnost rada pod različitim nivoima opterećenja