СБЕР УНИВЕРСИТЕТ

Школа Java Middle Developer

Kafka

Внутреннее устройство Kafka

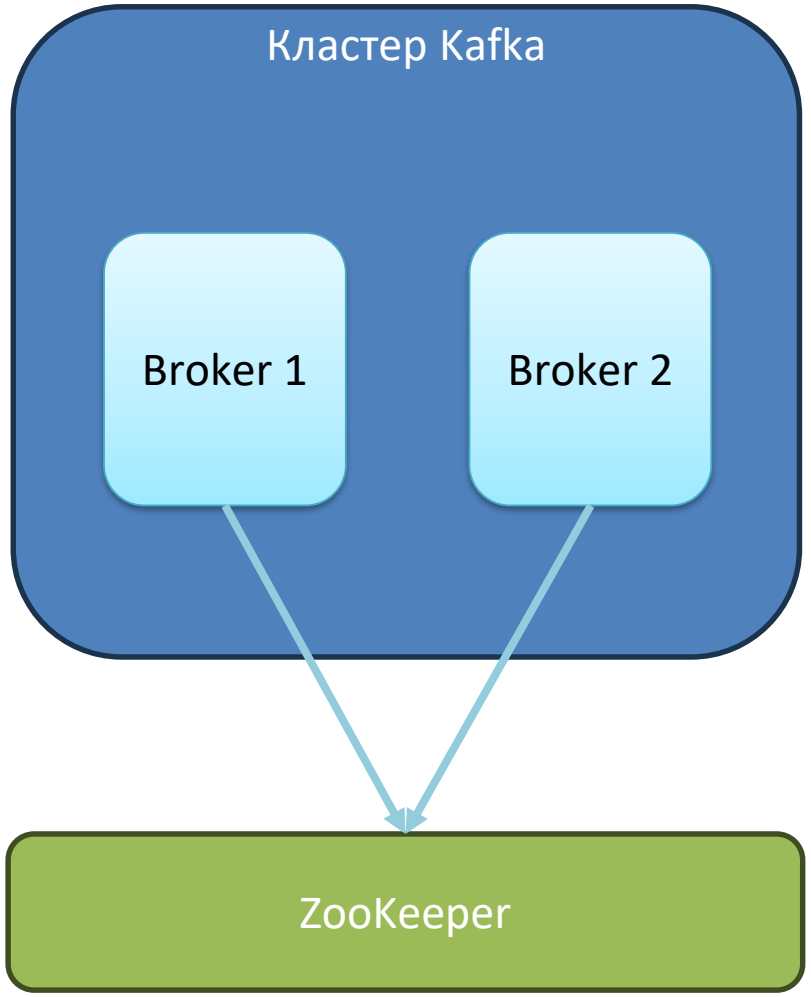
Содержание

1. Функционирование репликации Kafka;
2. Обработка Kafka запросов от производителей и потребителей;
3. Хранение данных в Kafka: форматы файлов и индексы.

Зачем нам знать как устроена Kafka внутри?

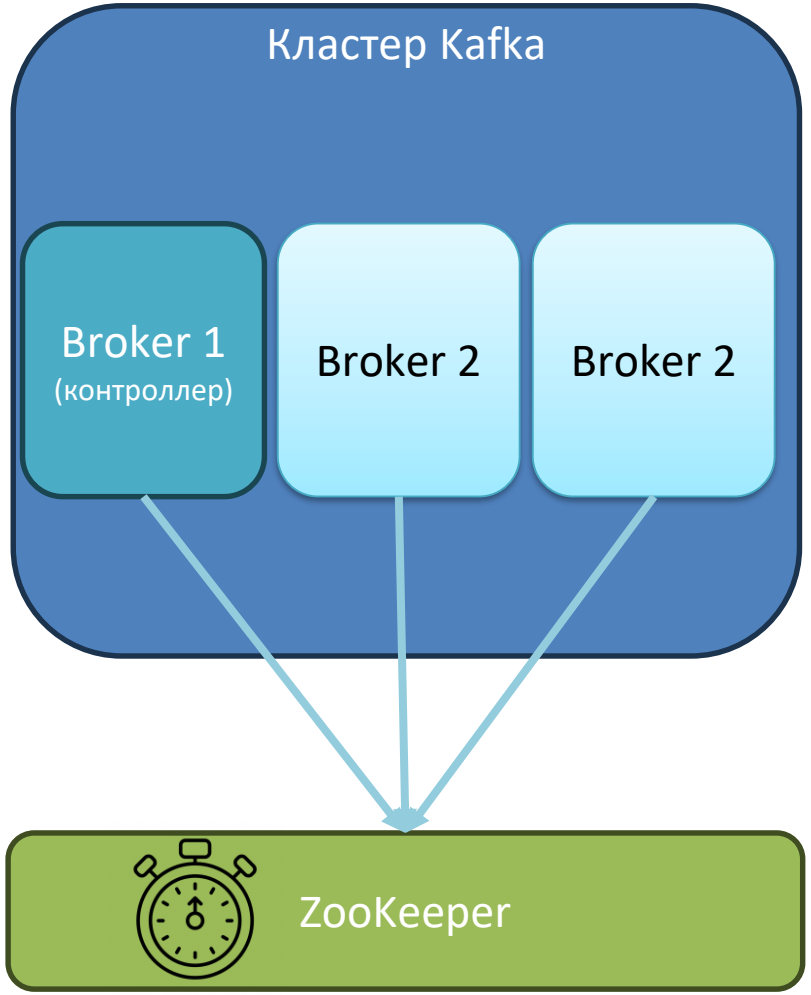
Устройство кластера Kafka

- ✓ У каждого брокера есть уникальный идентификатор (задаётся в конфигурации или генерируется автоматически);
- ✓ При каждом запуске процесса брокер регистрируется с своим ID в ZooKeeper;
- ✓ Если попробовать запустить второй брокер с тем же ID, будет возвращена ошибка;
- ✓ При потере связи брокера с ZooKeeper созданный при запуске брокера временный узел будет автоматически удален из ZooKeeper.



Контроллер

- Контроллер - это брокер Kafka, который помимо выполнения обычных своих функций отвечает за выбор ведущих реплик для партиций;
- Первый запущенный в кластере брокер становится контроллером;
- В случае останова брокера-контроллера или разрыва его соединения с ZooKeeper другие брокеры из кластера будут оповещены об этом посредством таймеров ZooKeeper и попытаются сами создать узел-контроллер.

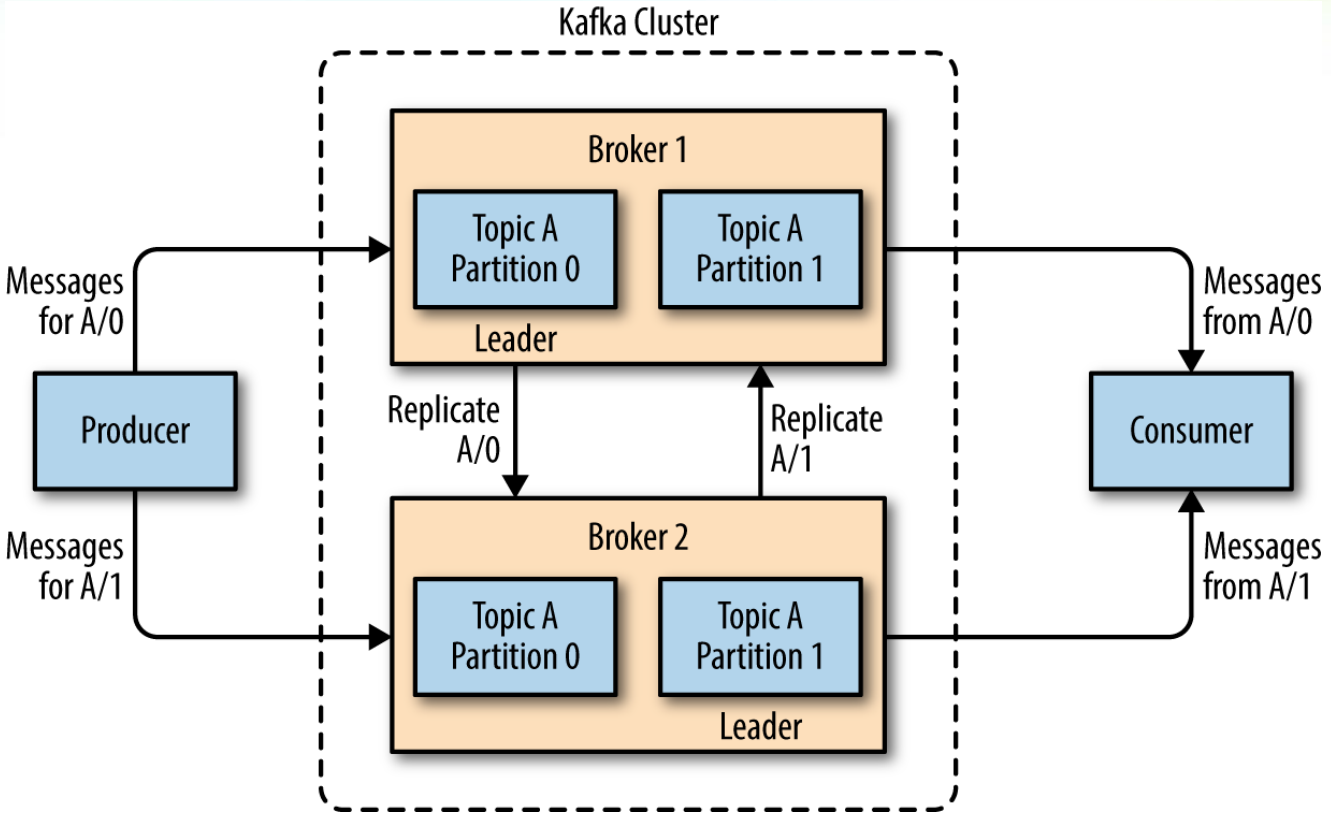


Контроллер

- Если контроллер обнаруживает, что брокер вышел из состава кластера, то он определяет новую ведущую реплику;
- Новая ведущая реплика начинает обслуживать запросы на генерацию и потребление от клиентов, а ведомые приступают к репликации сообщений от новой ведущей;
- Контроллер, получив информацию о присоединении брокера к кластеру, задействует идентификатор брокера для выяснения того, есть ли на этом брокере реплики. Если да, контроллер уведомляет как новый, так и уже существующие брокеры об изменении, а реплики на новом брокере приступают к репликации сообщений от имеющихся ведущих реплик.

Репликация

С помощью репликации Kafka обеспечивает доступность и сохраняемость данных при неизбежных сбоях отдельных узлов.

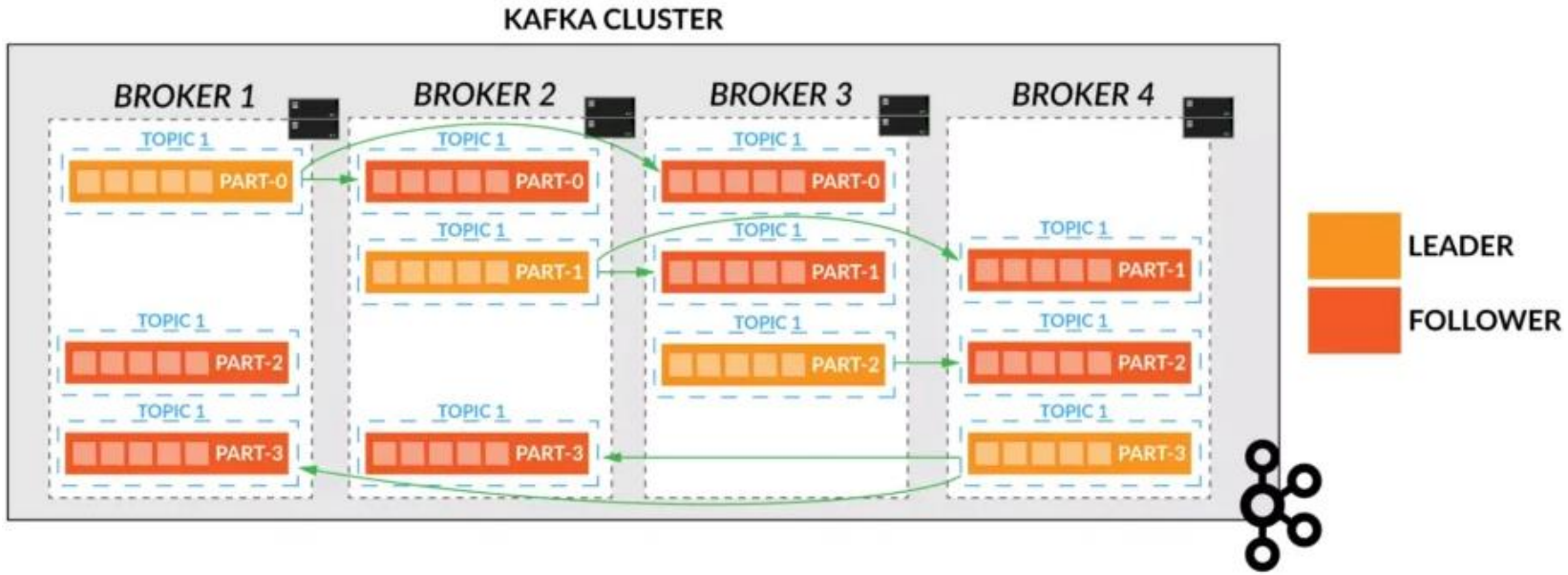


Существуют два вида реплик:

- Ведущие (leader) – обслуживает все запросы от производителей и потребителей;
- Ведомые (followers) - реплицируют сообщения от ведущей реплики и поддерживают актуальное по сравнению с ней состояние.

Репликация

- Ведущая реплика знает, какие ведомые реплики актуальны (по сравнению с ней), а какие нет;
- Ведомые реплики могут отставать вследствие множества причин;
- Чтобы не отстать от ведущей, ведомые посылают ей запросы Fetch;
- Ведущая реплика может определить, насколько отстают ведомые;
- Если реплика не запрашивала сообщений более 10 секунд или запрашивала, но отстает более чем на 10 секунд, то она считается рассогласованной (out of sync)



Репликация

- Стабильно запрашивающие новые сообщения реплики называются согласованными (in-sync);
- Только согласованная реплика может быть избрана ведущей репликой партиции;
- Параметр настройки replica.lag.time.max.ms задаёт промежуток времени, по истечении которого бездействующая или отстающая ведомая реплика будет сочтена рассогласованной;
- В каждой партиции есть предпочтительная ведущая реплика (preferred leader) - та, которая была ведущей в момент создания топика;
- Параметр auto.leader.rebalance.enable=true говорит о том, что Kafka будет проверять, является ли предпочтительная реплика ведущей и согласована ли она, инициируя в этом случае выбор ведущей реплики, чтобы сделать предпочтительную ведущую реплику действующей.

Как найти предпочтительную реплику?

- Подробные данные о разделах и репликах можно найти в выводимой утилитой kafka-topics.sh информации;
- Предпочтительная ведущая реплика всегда стоит первой в списке;
- При перераспределении реплик вручную важно помнить, что их нужно распределять по разным брокерам, чтобы не перегружать одни брокеры ведущими, оставляя другие без законной доли нагрузки.

Обработка запросов

Каждый запрос имеет стандартный заголовок, включающий:

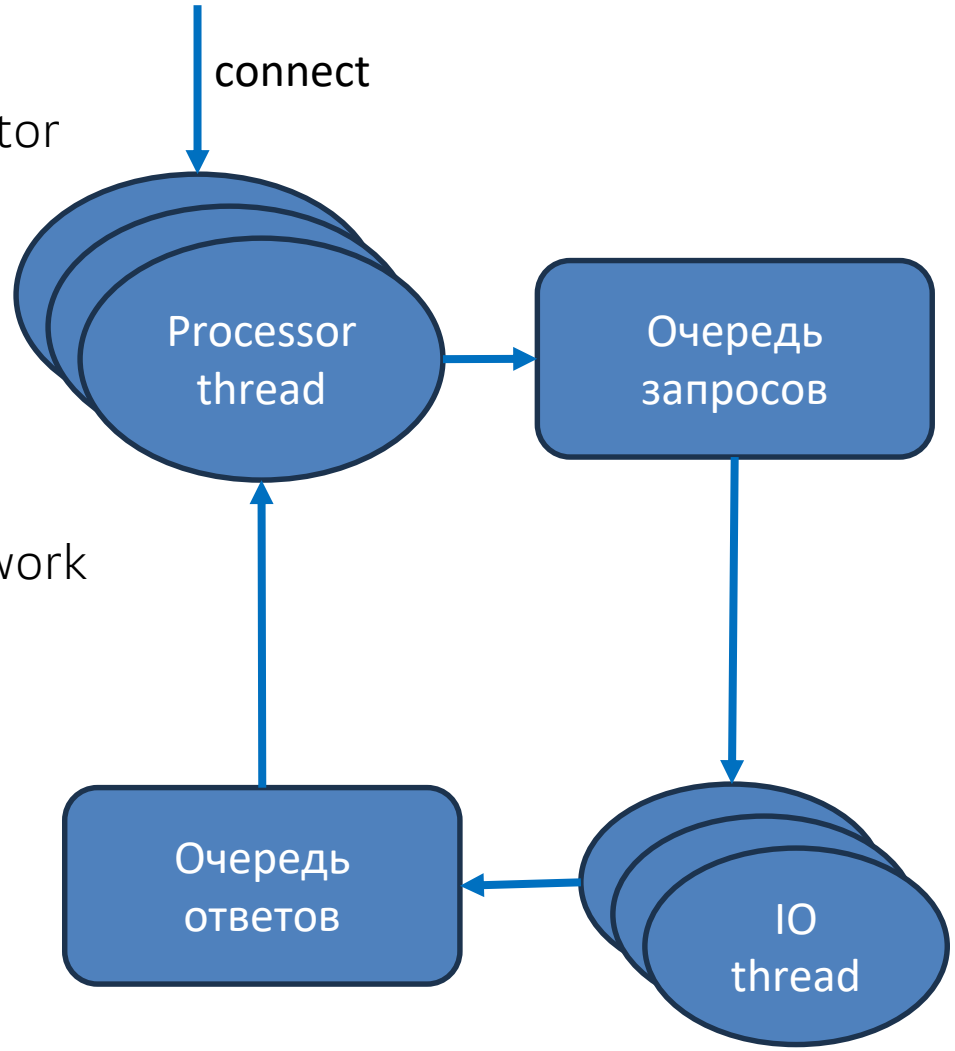
- тип запроса;
- версию запроса;
- идентификатор корреляции - число, уникально идентифицирующее запрос и включаемое также в ответ и журналы ошибок;
- идентификатор клиента - используется для идентификации отправившего запрос приложения.

Протокол, используемый в Kafka, изложен в документации Kafka и находится в свободном доступе (https://kafka.apache.org/protocol.html)

Обработка запросов

- Для каждого порта, на котором брокер выполняет прослушивание, запускается принимающий поток (acceptor thread);
- Принимающий поток создаёт соединение и передаёт контроль над ним обрабатывающему потоку (processor thread);
- Обрабатывающие потоки также называют сетевыми (network threads), их количество можно задать в конфигурации;
- После помещения запросов в очередь их обработку начинают потоки ввода/вывода (I0 threads).

  Распространенные типы запросов:
  - Запросы от производителей;
  - Запросы на извлечение.



Откуда клиенты знают, куда им отправлять запросы?

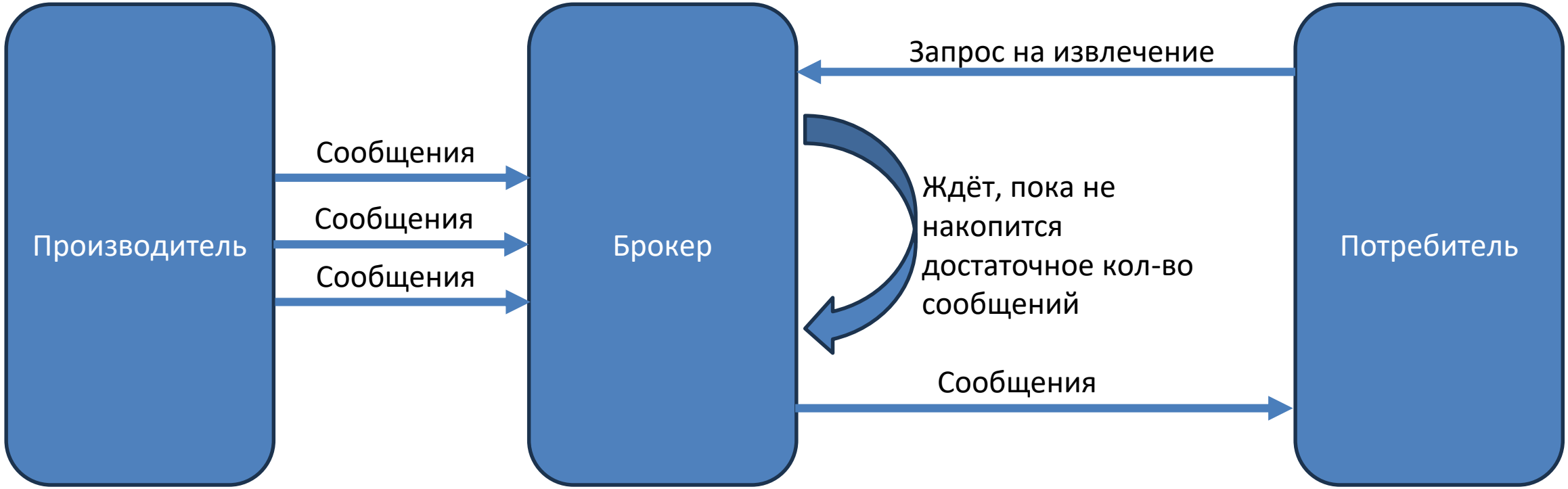
- Клиенты Kafka применяют запрос метаданных (metadata request), включающий список топиков, интересующих клиента;
- Запросы метаданных можно отправлять любому брокеру;
- Клиенты обычно кэшируют эту информацию и используют её для направления запросов производителей и запросов на извлечение нужному брокеру для каждой из партиций;
- Интервал обновления этой информации задается параметром конфигурации metadata.max.age.ms.

Запросы от производителей

Брокер, на котором находится ведущая реплика партиции, при получении запроса к ней от производителя начинает с нескольких проверок:

- Есть ли у отправляющего данные пользователя права на запись в этот топик?
- Допустимо ли указанное в запросе значение параметра acks?
- Если параметр acks установлен в значение all, достаточно ли согласованных реплик для безопасной записи сообщения?

Запросы на извлечение



Другие запросы

- Контроллер, извещая о новой ведущей реплике партиции, отправляет запрос LeaderAndIsr новой ведущей реплике, чтобы она начала принимать запросы клиентов, и ведомым репликам, чтобы они ориентировались на новую ведущую реплику;
- Протокол Kafka включает >20 типов запросов и постоянно совершенствуется;
- Запрос ApiVersionRequest позволяет клиентам запрашивать у брокера поддерживаемые версии запросов и применять соответствующую версию.

Физическое хранилище

- Основная единица хранения Kafka - реплика партиции;
- Размер партиции ограничивается доступной памятью на текущем физическом сервере;
- С помощью параметра log.dirs можно задать список каталогов для хранения партиций.

Распределение партиций

Основные задачи распределения следующие:

- Равномерно распределить реплики по брокерам;
- Гарантировать, что все реплики для каждой из партиций находятся на разных брокерах;
- Если у брокеров имеется информация о размещении в стойках, то желательно по возможности разместить реплики для каждого из партиций на различных стойках;

Управление файлами

- Kafka не хранит данные вечно и не ждёт, когда все потребители прочтут сообщение, перед тем как его удалить;
- Партиции разбиваются на сегменты (по умолчанию 1 Гбайт/1 неделя);
- Сегмент, в который в настоящий момент производится запись, называется активным (active segment);
- Активный сегмент никогда не удаляется.

Формат файлов

- Каждый сегмент хранится в отдельном файле данных;
- Формат файла на диске идентичен формату сообщений, отправляемых от производителя брокеру, а затем от брокера потребителям;
- Брокеры Kafka идут в комплекте с утилитой DumpLogSegment, позволяющей просматривать сегменты партиций в файловой системе и исследовать их содержимое.

```
bin/kafka-run-class.sh kafka.tools.DumpLogSegments
```

Индексы

- Kafka даёт потребителям возможность извлекать сообщения, начиная с любого смещения;
- Kafka поддерживает индексы для всех партиций;
- Индекс задаёт соответствие смещения файлу сегмента и месту в этом файле;
- Индексы также разбиты на сегменты.

Сжатие

Kafka поддерживает две возможные стратегии сохранения для топика:

- delete (удалять), при которой события, чей возраст превышает интервал хранения, удаляются;
- compact (сжимать), при которой сохраняется только последнее значение для каждого из ключей топика.

Спасибо за внимание