OTUS

Онлайн образование

otus.ru

Проверить, идет ли запись

Меня хорошо видно && слышно?

Тема вебинара

Безопасность Kafka

Заигрин Вадим

Ведущий эксперт по технологиям, Сбербанк

vzaigrin@yandex.ru
https://t.me/vzaigrin

Преподаватель

Вадим Заигрин

Более 30 лет в ИТ:
- Big Data
  - Data Engineer
  - Data Science
- Разработка
  - Scala, Java, Python, C, Lisp
- IT Infrastructure
  - Администрирование
  - Сопровождение
  - Архитектура

Big Data проекты в банках, телекоме и в рознице.

Правила вебинара

Активно
участвуем

Off-topic обсуждаем
в Telegram

Задаем вопрос
в чат или голосом

Вопросы вижу в чате,
могу ответить не сразу

Условные обозначения

Индивидуально

Время, необходимое
на активность

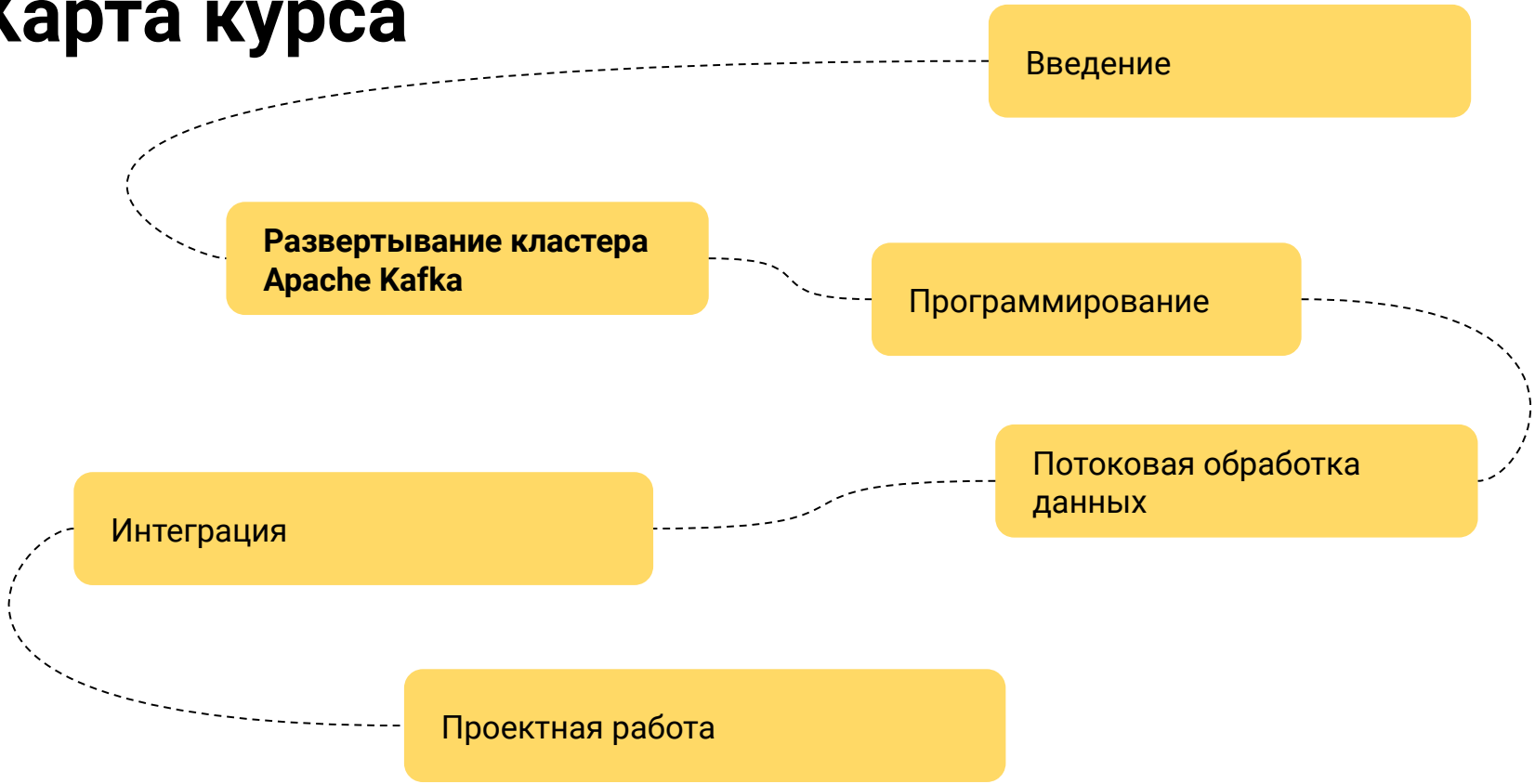
Пишем в чат

Говорим голосом

Документ

Ответьте себе или
задайте вопрос

Карта курса


Введение
Развертывание кластера Apache Kafka
Программирование
Потоковая обработка данных
Интеграция
Проектная работа

Маршрут вебинара

- Безопасность
- Аутентификация
- Авторизация
- Аудит
- Рефлексия

Цели вебинара

1.	Узнаем зачем нужена безопасность
2.	Узнаем что Kafka предлагает для безопасности
3.	Узнаем как настроить безопасность Kafka

Смысл

1.	Сможем настраивать безопасность
2.	Сможем работать с Kafka безопасно

Безопасность

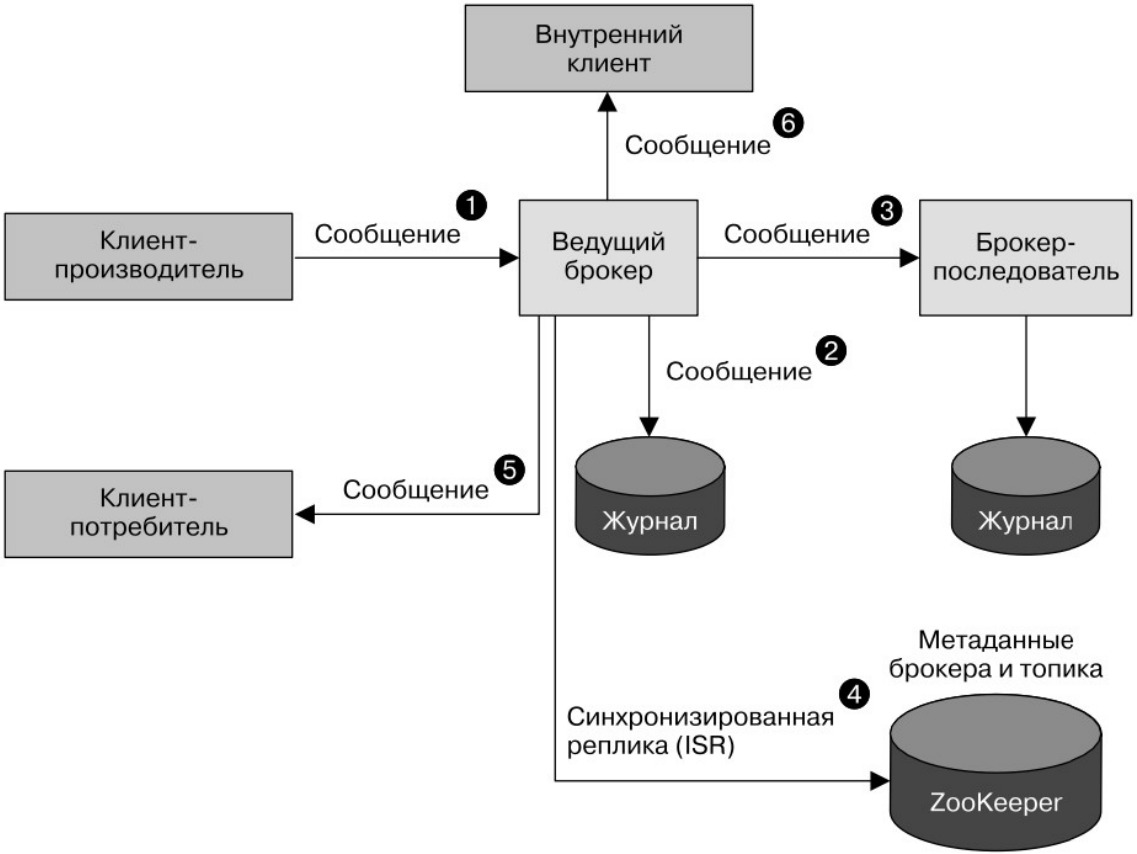
Информационная безопасность

Информационная безопасность (*Information Security*, *InfoSec*) — практика предотвращения несанкционированного доступа, использования, раскрытия, искажения, изменения, исследования, записи или уничтожения информации.

Основы безопасности

- **Аутентификация** — это процесс доказательства, что пользователь или приложение действительно является тем, за кого себя выдаёт.
- **Авторизация** определяет какие действия разрешено выполнять.
- **Шифрование** — это сокрытие исходного вида информации.
- **Аудит** — отслеживание действий.
- **Квоты** — ограничение использования ресурсов.

Безопасность в Kafka



Аутентификация

Протоколы безопасности

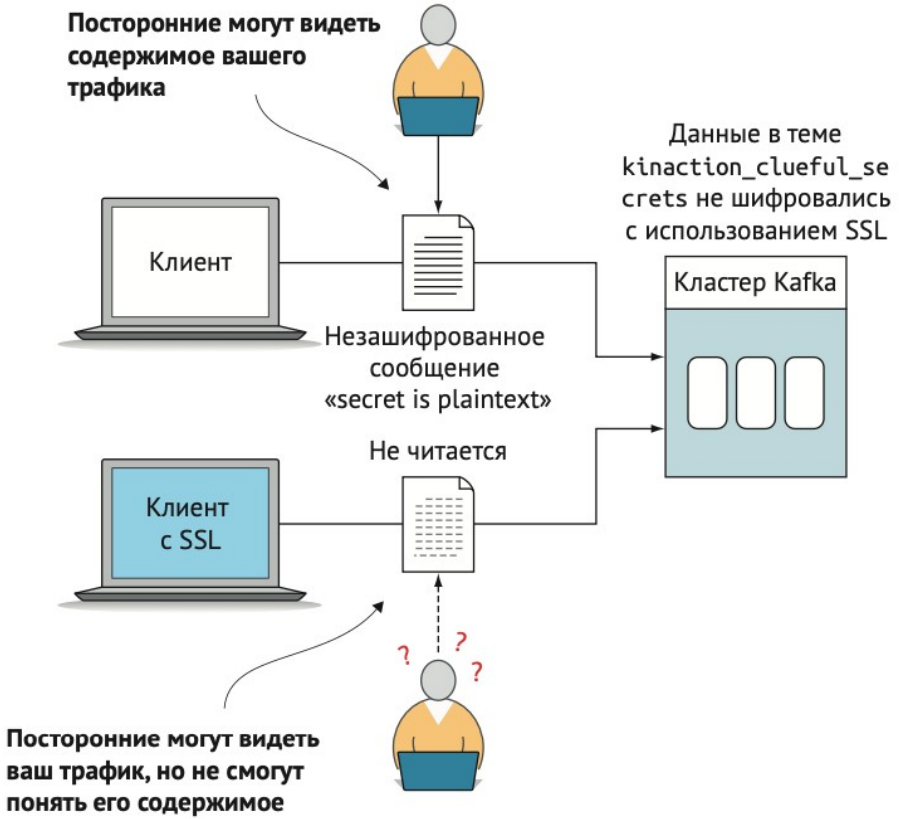
- *PLAINTEXT* — транспортный уровень с открытым тестом без аутентификации
- *SSL* — транспортный уровень SSL с аутентификацией клиента SSL
- *SASL_PLAINTEXT* — транспортный уровень с открытым тестом с аутентификацией SASL
- *SASL_SSL* — транспортный уровень SSL с аутентификацией клиента SASL

SSL

SSL (Secure Sockets Layer)

`ssl.client.auth=`

- none — не требуется
- `required` — требуется
- `requested` — запрашивается



Настройка SSL

1) Создать SSL ключ и сертификат для каждого брокера
2) Создать собственный центр авторизации
3) Подписать сертификат
4) Настроить брокеры
5) Настроить клиенты

Создать SSL ключ и сертификат для брокера

```
keytool -genkey \
-keyalg RSA \
-keystore server.keystore.jks \
-keypass password \
-alias localhost \
-validity 365 \
-storetype pkcs12 \
-storepass password \
-dname "CN=localhost,OU=Kafka,O=Otus,L=Moscow,ST=Moscow,C=RU"
```

Создать собственный центр авторизации (CA)

- `openssl req -new -x509 -keyout ca-key -out ca-cert -days 365`
- `keytool -keystore client.truststore.jks -alias CARoot -importcert -file ca-cert`
- `keytool -keystore server.truststore.jks -alias CARoot -importcert -file ca-cert`

Подписать сертификат

- `keytool -keystore server.keystore.jks -alias localhost -certreq -file cert-file`
- `openssl x509 -req -CA ca-cert -CAkey ca-key -in cert-file -out cert-signed -days 365 -CAcreateserial -passin pass:password`
- `keytool -keystore server.keystore.jks -alias CARoot -importcert -file ca-cert`
- `keytool -keystore server.keystore.jks -alias localhost -importcert -file cert-signed`

Настроить брокер

- `listeners=SSL://:9093`
- `ssl.keystore.location=/opt/kafka/private/server.keystore.jks`
- `ssl.keystore.password=password`
- `ssl.key.password=password`
- `ssl.truststore.location=/opt/kafka/private/server.truststore.jks`
- `ssl.truststore.password=password`
- `security.inter.broker.protocol=SSL`
- `ssl.client.auth=requested`
- `ssl.endpoint.identification.algorithm=`

Настроить клиента

client-ssl.properties

- `security.protocol=SSL`
- `ssl.truststore.location=/var/private/ssl/client.truststore.jks`
- `ssl.truststore.password=password`

Проверяем

- `bin/zookeeper-server-start.sh -daemon config/zookeeper.properties`
- `bin/kafka-server-start.sh -daemon config/server-security.properties`
- `kafka-topics.sh --list --bootstrap-server localhost:9093 --command-config client-ssl.properties`
- `kafka-console-producer.sh --bootstrap-server localhost:9093 --topic test --producer.config client-ssl.properties`
- `kafka-console-consumer.sh --bootstrap-server localhost:9093 --topic test --consumer.config client-ssl.properties -from-beginning`

LIVE

SASL

SASL (Simple Autentication and Security Layer)

SASL — это фреймворк для предоставления аутентификации и защиты данных в протоколах на основе соединений.

- *GSSAPI* — аутентификация Kerberos
- *PLAIN* — аутентификацией по имени пользователя/паролю, проверка пароля из внешнего хранилища паролей
- *SCRAM-SHA-256* и SCRAM-SHA-512 — аутентификацией по имени пользователя/паролю (не требует внешнего хранилища паролей)
- *OAUTHBEARER* — аутентификация с помощью токенов OAuth

SASL/GSSAPI

Брокер

```
sasl.enabled.mechanisms=GSSAPI
listener.name.external.gssapi.sasl.jaas.config=\  ❶
    com.sun.security.auth.module.Krb5LoginModule required \
        useKeyTab=true storeKey=true     \
        keyTab="/path/to/broker1.keytab" \  ❷
        principal="kafka/broker1.example.com@EXAMPLE.COM";  ❸
```

Клиент

```
sasl.mechanism=GSSAPI
sasl.kerberos.service.name=kafka ❶
sasl.jaas.config=com.sun.security.auth.module.Krb5LoginModule required \
    useKeyTab=true storeKey=true   \
    keyTab="/path/to/alice.keytab" \
    principal="Alice@EXAMPLE.COM"; ❷
```

SASL/PLAIN

Брокер

```
sasl.enabled.mechanisms=PLAIN
sasl.mechanism.inter.broker.protocol=PLAIN
listener.name.external.plain.sasl.jaas.config=\
    org.apache.kafka.common.security.plain.PlainLoginModule required \
        username="kafka" password="kafka-password" \ ❶
        user_kafka="kafka-password" \
        user_Alice="Alice-password"; ❷
```

Клиент

```
sasl.mechanism=PLAIN
sasl.jaas.config=org.apache.kafka.common.security.plain.PlainLoginModule \
    required username="Alice" password="Alice-password";
```

SASL/SCRAM

Создаём пользователей до запуска брокеров

```
$ bin/kafka-configs.sh --zookeeper localhost:2181 --alter --add-config \
    'SCRAM-SHA-512=[iterations=8192,password=Alice-password]'          \
    --entity-type users --entity-name Alice
```

Брокер

```
sasl.enabled.mechanisms=SCRAM-SHA-512
sasl.mechanism.inter.broker.protocol=SCRAM-SHA-512
listener.name.external.scram-sha-512.sasl.jaas.config=\
    org.apache.kafka.common.security.scram.ScramLoginModule required \
        username="kafka" password="kafka-password"; ❶
```

Клиент

```
sasl.mechanism=SCRAM-SHA-512
sasl.jaas.config=org.apache.kafka.common.security.scram.ScramLoginModule \
    required username="Alice" password="Alice-password";
```

SASL/OAUTHBEARER

Брокер

```
sasl.enabled.mechanisms=OAUTHBEARER
sasl.mechanism.inter.broker.protocol=OAUTHBEARER
listener.name.external.oauthbearer.sasl.jaas.config=\
    org.apache.kafka.common.security.oauthbearer.OAuthBearerLoginModule \
        required unsecuredLoginStringClaim_sub="kafka"; ❶
```

Клиент

```
sasl.mechanism=OAUTHBEARER
sasl.jaas.config=\
    org.apache.kafka.common.security.oauthbearer.OAuthBearerLoginModule \
        required unsecuredLoginStringClaim_sub="Alice"; ❶
```

SASL/PLAIN Пример

Настроить и запустить ZooKeeper

- *zookeeper-sasl.properties*
  - `authProvider.sasl=org.apache.zookeeper.server.auth.SASLAuthenticationProvider`
- *zookeeper_jaas.conf*

```
Server {
       org.apache.zookeeper.server.auth.DigestLoginModule required
       user_super="admin-secret"
       user_kafka="kafka-secret";
};
```

- `KAFKA_OPTS="-Djava.security.auth.login.config=/opt/kafka/private/zookeeper_jaas.conf" /opt/kafka/bin/zookeeper-server-start.sh -daemon /opt/kafka/config/zookeeper-sasl.properties`

Настроить брокер

- `listeners=SSL://:9093,SASL_SSL://:9094`
- `security.inter.broker.protocol=SSL`
- `ssl.client.auth=required`
- `ssl.keystore.location=/opt/kafka/private/server.keystore.jks`
- `ssl.keystore.password=password`
- `ssl.key.password=password`
- `ssl.truststore.location=/opt/kafka/private/server.truststore.jks`
- `ssl.truststore.password=password`
- `ssl.endpoint.identification.algorithm=`
- `sasl.enabled.mechanisms=PLAIN`

Настроить брокер

kafka_server_jaas.conf

```
KafkaServer {
   org.apache.kafka.common.security.plain.PlainLoginModule required
   username="kafkabroker"
   password="kafkabroker-secret"
   user_kafkabroker="kafkabroker-secret"
   user_kafka-broker-metric-reporter="kafkabroker-metric-reporter-secret"
   user_client="client-secret";
};

Client {
   org.apache.zookeeper.server.auth.DigestLoginModule required
   username="kafka"
   password="kafka-secret";
};
```

Запустить брокер

- `KAFKA_OPTS="-Djava.security.auth.login.config=/opt/kafka/private/kafka_server_jaas.conf" /opt/kafka/bin/kafka-server-start.sh -daemon /opt/kafka/config/server-sasl.properties`

Настроить клиента

client-sasl.properties

- security.protocol=SASL_SSL
- ssl.truststore.location=/var/private/ssl/client.truststore.jks
- ssl.truststore.password=password
- sasl.mechanism=PLAIN
- sasl.jaas.config=org.apache.kafka.common.security.plain.PlainLoginModule required \
- username="client" \
- password="client-secret";

Проверяем

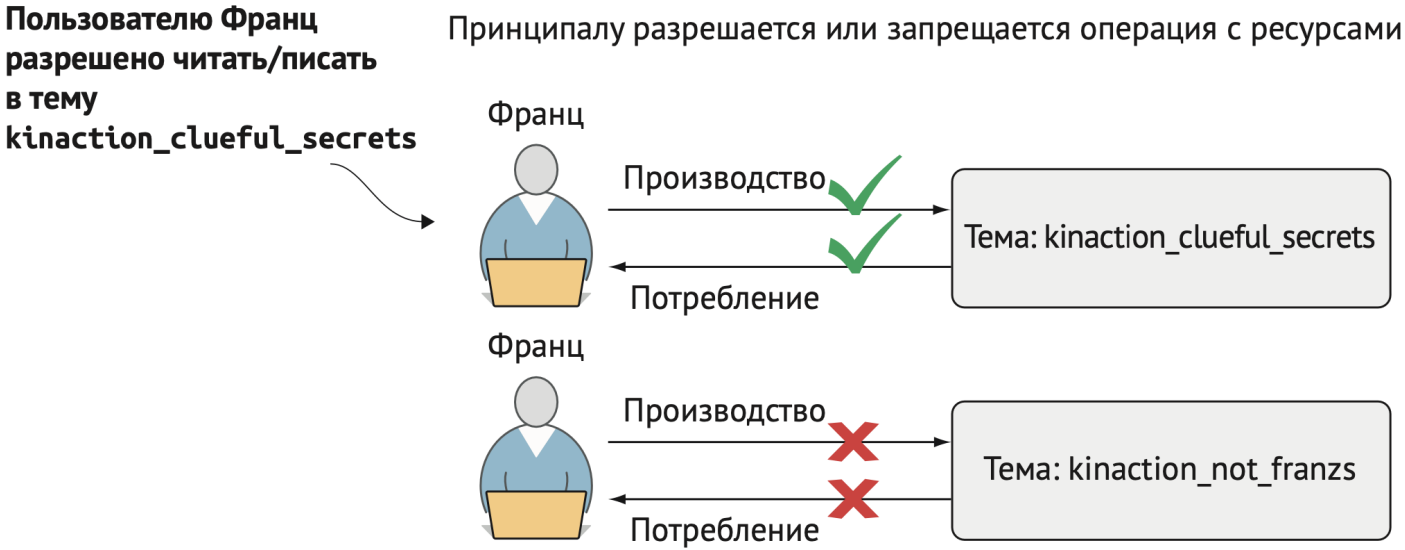
- `kafka-topics.sh --list --bootstrap-server localhost:9094 --command-config client-sasl.properties`
- `kafka-console-producer.sh --bootstrap-server localhost:9094 --topic test --producer.config client-sasl.properties`
- `kafka-console-consumer.sh --bootstrap-server localhost:9094 --topic test --consumer.config client-sasl.properties -from-beginning`

LIVE

Авторизация

Авторизация

Авторизация определяет какие действия разрешено выполнять над какими ресурсами



Включение авторизации

Брокеры управляют контролем доступа с помощью авторизатора

- Kafka с ZooKeeper
  - `authorizer.class.name=kafka.security.authorizer.AclAuthorizer`
- Kafka с KRaft
  - `authorizer.class.name=org.apache.kafka.metadata.authorizer.StandardAuthorizer`

Kafka ACL

`Principal {P} is [Allowed|Denied] Operation {O} From Host {H} on any Resource {R} matching ResourcePattern {RP}`

- Принципал: `<principalType>:<principalName>, User:*` - все пользователи
- Операция: `Describe|Create|Delete|Alter|Read|Write|DescribeConfigs|AlterConfigs`
- Хост: `ip-адрес, * - все`
- Ресурс: `Cluster|Topic|Group|TransactionalId|DelegationToken`
- Шаблон: `Literal|Prefixed`

Kafka ACL

- Если `Resource {R}` не соответствует никакому `ResourcePattern`, тогда у ресурса R нет ACL , и доступ есть только у суперпользователей

- `allow.everyone.if.no.acl.found=true` — разрешить доступ
- `super.users=User:Bob;User:Alice` — определить суперпользователей

kafka-acls.sh

`kafka-acls.sh` — утилита управления ACL

- `--add`
- `--remove`
- `--list`
- `--bootstrap-server`
- `--command-config`
- `--cluster`
- `--topic [topic-name]`
- `--group [group-name]`
- `--user-principal [user-principal]`
- `--resource-pattern-type [pattern-type]`

kafka-acls.sh

- `--allow-principal`
- `--deny-principal`
- `--principal`
- `--allow-host`
- `--deny-host`
- `--operation`
- `--producer`
- `--consumer`
- `--idempotent`
- `--force`

Примеры

- `kafka-acls.sh --bootstrap-server localhost:9094 --add --allow-principal User:Bob --operation Write --topic test --command-config client-sasl.properties`
- `kafka-acls.sh --bootstrap-server localhost:9094 --add --allow-principal User:Alice --operation Read --topic test --command-config client-sasl.properties`
- `kafka-acls.sh --bootstrap-server localhost:9094 --list --command-config client-sasl.properties`
-
- `kafka-topics.sh --list --bootstrap-server localhost:9094 --command-config client-sasl.properties`
- `kafka-topics.sh --list --bootstrap-server localhost:9094 --command-config client-bob.properties`
- `kafka-topics.sh --list --bootstrap-server localhost:9094 --command-config client-alice.properties`
-
- `kafka-console-producer.sh --bootstrap-server localhost:9094 --topic test --producer.config client-bob.properties`
- `kafka-console-consumer.sh --bootstrap-server localhost:9094 --topic test --consumer.config client-alice.properties -from-beginning`
- `kafka-console-consumer.sh --bootstrap-server localhost:9094 --topic test --consumer.config client-sasl.properties -from-beginning`

LIVE

Аудит

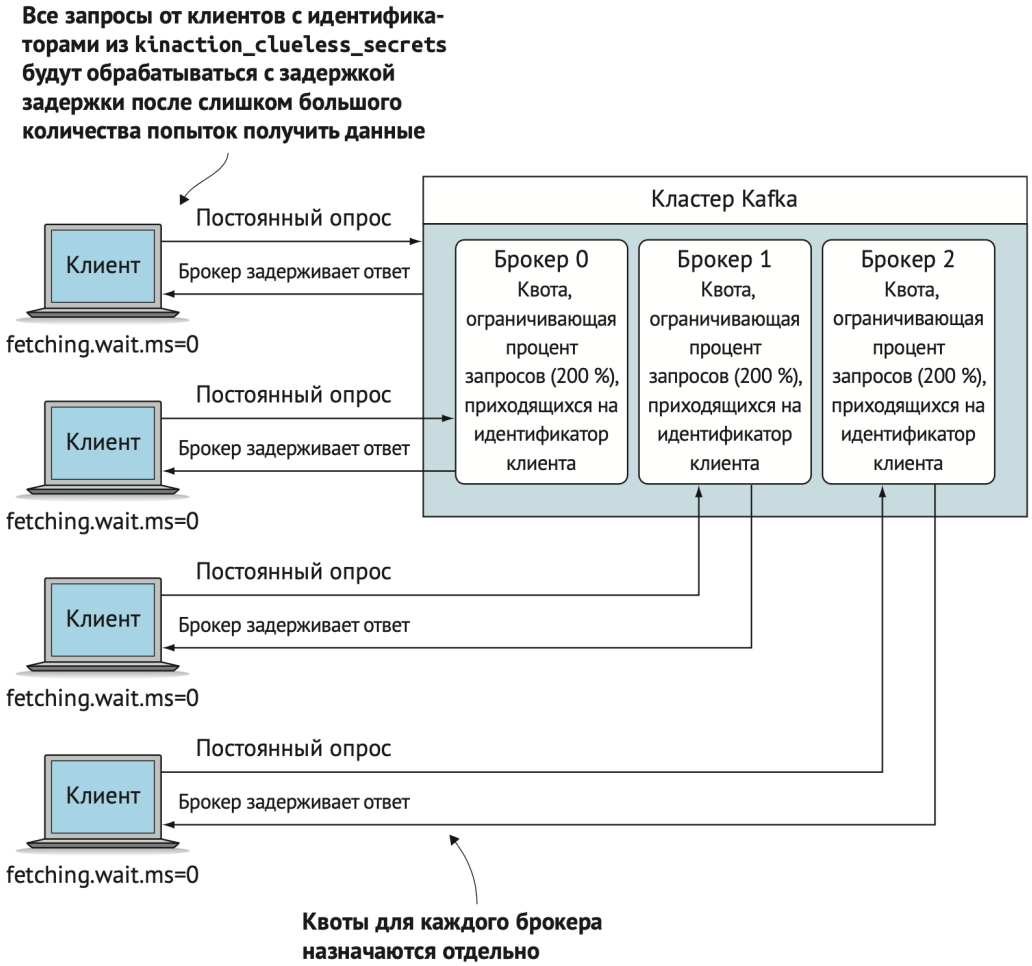
Аудит

Аудит — отслеживание действий.

- `log4j.properties` — настройка журналирования Kafka
  - `kafka.authorizer.logger` — журналирование авторизации
  - `kafka.request.logger` — журналирование запросов

Квоты

Квоты


Все запросы от клиентов с идентификаторами из kinaction_clueless_secrets будут обрабатываться с задержкой задержки после слишком большого количества попыток получить данные
Кластер Kafka
Брокер 0
Квота, ограничивающая процент запросов (200 %), приходящихся на идентификатор клиента
Брокер 1
Квота, ограничивающая процент запросов (200 %), приходящихся на идентификатор клиента
Брокер 2
Квота, ограничивающая процент запросов (200 %), приходящихся на идентификатор клиента
Клиент
Постоянный опрос
Брокер задерживает ответ
fetching.wait.ms=0
Клиент
Постоянный опрос
Брокер задерживает ответ
fetching.wait.ms=0
Клиент
Постоянный опрос
Брокер задерживает ответ
fetching.wait.ms=0
Клиент
Постоянный опрос
Брокер задерживает ответ
fetching.wait.ms=0
Квоты для каждого брокера назначаются отдельно

Ограничение пропускной способности сети

Определение квоты

```
bin/kafka-configs.sh --bootstrap-server localhost:9094 --alter \
  --add-config 'producer_byte_rate=1048576,
➥ consumer_byte_rate=5242880' \
  --entity-type clients --entity-name kinaction_clueful
```

Производителям разрешено передавать до 1 Мбайт/с, а потребителям читать до 5 Мбайт/с

Квота распространяется на клиентов с идентификатором `kinaction_clueful`

Вывод списка и удаление квот

```
bin/kafka-configs.sh --bootstrap-server localhost:9094 \
  --describe \
  --entity-type clients --entity-name kinaction_clueful
```

Перечисляет существующие настройки для указанного клиента

```
bin/kafka-configs.sh --bootstrap-server localhost:9094 --alter \
  --delete-config
➥ 'producer_byte_rate,consumer_byte_rate' \
  --entity-type clients --entity-name kinaction_clueful
```

Команда `delete-config` удалит только что добавленную квоту

Вопросы?

Ставим “+”,
если вопросы есть

Ставим “–”,
если вопросов нет

Литература

Список материалов для изучения

1. Kafka Security
2. Kafka Security Overview
3. Apache Kafka Security
4. OpenSSL
5. OpenSSL Binaries
6. TLS/SSL и сертификаты SSL (X.509)

Рефлексия

Рефлексия

С какими впечатлениями уходите с вебинара?

Как будете применять на практике то,
что узнали на вебинаре?

Следующий вебинар

Producer

- Ссылка на вебинар будет в ЛК за 15 минут
- Материалы к занятию в ЛК — можно изучать
- Обязательный материал обозначен красной лентой

Заполните, пожалуйста, опрос о занятии по ссылке в чате

Спасибо за внимание!

Приходите на следующие вебинары

Заигрин Вадим

Ведущий эксперт по технологиям, Сбербанк

vzaigrin@yandex.ru
https://t.me/vzaigrin