

# Онлайн образование

otus.ru



Проверить, идет ли запись

# Меня хорошо видно && слышно?



Тема вебинара

# Мониторинг кластера Kafka



**Заигрин Вадим**

Ведущий эксперт по технологиям, Сбербанк

[vzaigrin@yandex.ru](mailto: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

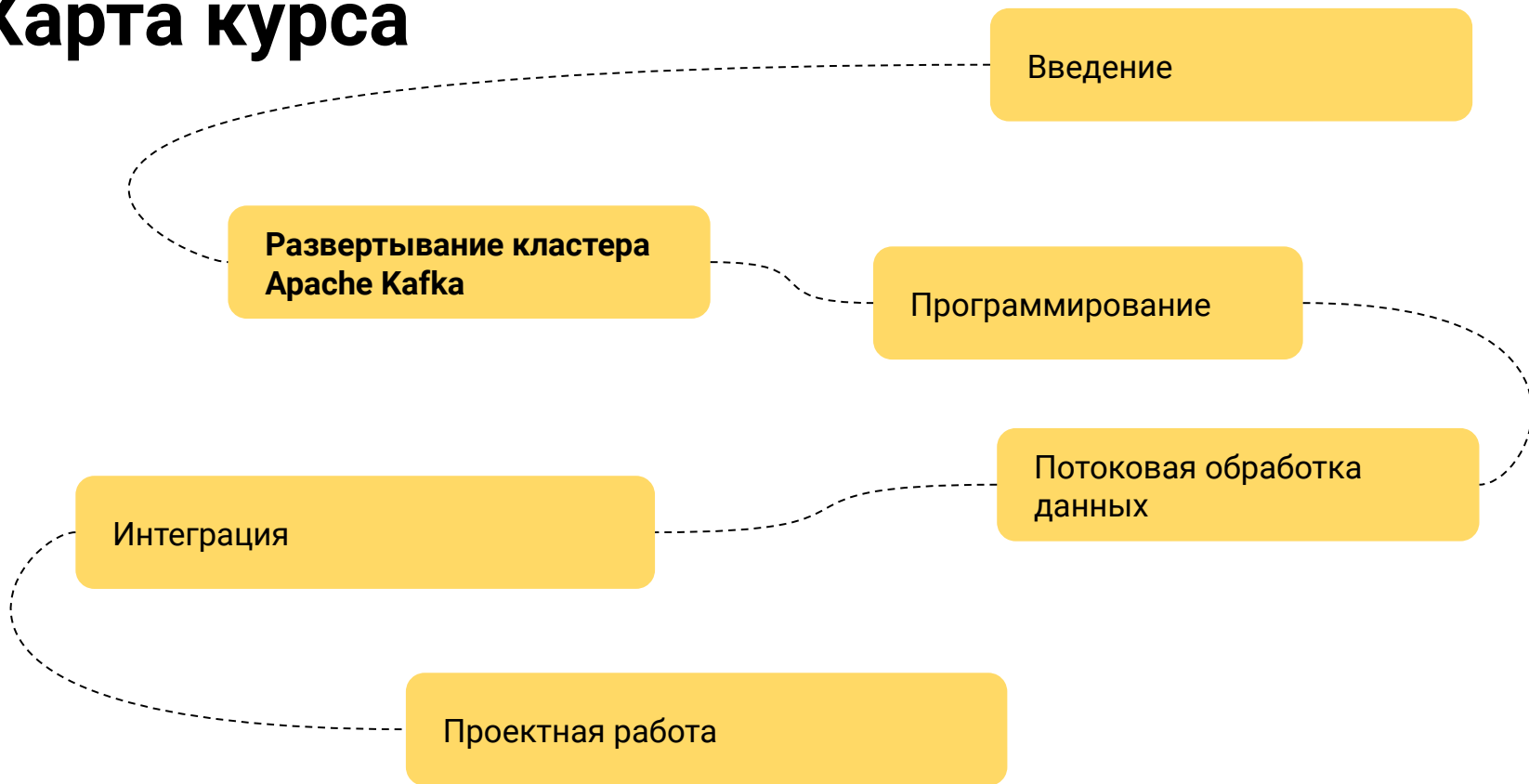


Документ



Ответьте себе или  
задайте вопрос

# Карта курса



# Маршрут вебинара



Мониторинг

Мониторинг Kafka

Промышленный мониторинг

Специализированные системы  
мониторинга

Рефлексия



# Цели вебинара

- |    |   |
|----|---|
| 1. | Узнаем зачем нужен мониторинг               |
| 2. | Узнаем что Kafka предлагает для мониторинга |
| 3. | Узнаем как настроить мониторинг Kafka       |



# Смысл

- |    |                                |
|----|--------------------------------|
| 1. | Сможем настраивать мониторинг  |
| 2. | Сможем использовать мониторинг |



# Мониторинг

# Мониторинг

**Мониторинг** — это система постоянного наблюдения за явлениями и процессами, проходящими в окружающей среде и обществе, результаты которого служат для обоснования управленческих решений



Упреждение  
аварий



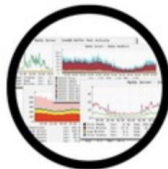
Уведомление  
об аварии



Быстрая  
диагностика



Точные  
данные



Полезные  
графики



Актуальные  
алерты

# Мониторинг в ИТ

- Системный мониторинг
- Мониторинг событий
- Мониторинг сети
- Мониторинг производительности приложений
- Мониторинг бизнес-транзакций
- Мониторинг активности пользователей
- Мониторинг веб-сайта

# Задачи мониторинга

- Сбор информации о текущем состоянии
- Предупреждение о возможных сбоях и проблемах
- Отражение состояния системы / сервера / сервиса / компонента сервиса
- Сбор статистики и визуализация
- Отчёты
- Дашборды

# Уровни мониторинга

- Мониторинг оборудования:
  - CPU
  - RAM
  - HDD
  - HBA
- Мониторинг приложений:
  - Кол-во запросов в единицу времени
  - Кол-во активных пользователей
  - Кол-во записей
- Мониторинг бизнес-метрик

# Типы мониторинга

- «Whitebox» — мониторинг по метрикам приложения / сервиса:
  - API
  - SNMP
  - интерфейсы
  - логи
- «Blackbox» — мониторинг по внешнему состоянию системы:
  - ping
  - HTTP request
  - Открытый порт
  - Наличие процессов

# Исторические данные

- Как их собирать?
- Где их хранить?
- Сколько их хранить?
- Как с ними работать?

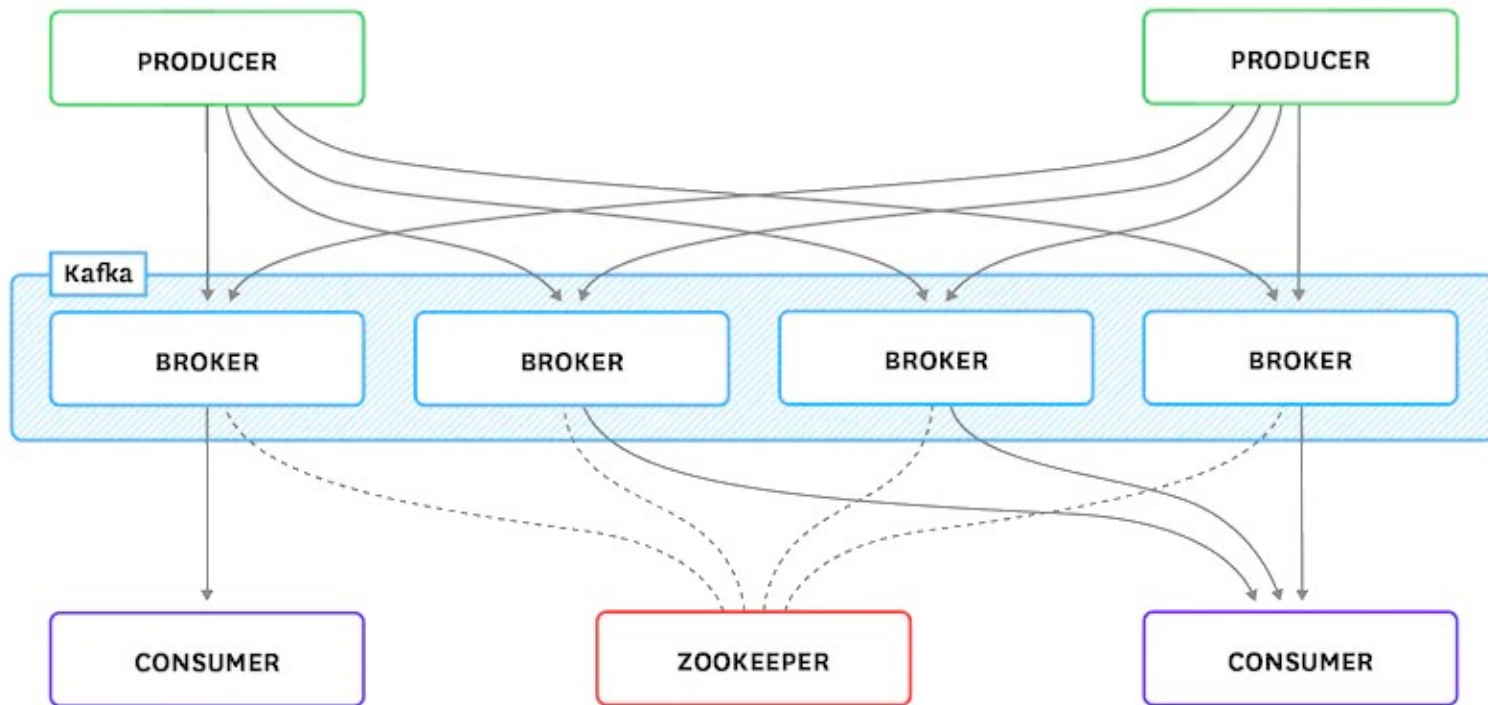


# Инструменты мониторинга

- Zabbix — свои агент, веб, база
- Nagios / Icinga — свои агент, веб, база
- Graphite — carbon (агент), whisper (tsdb), graphite-web
- Cacti, Ganglia, Collectd, MRTG, Munin — база rrdtool
- Netdata — агент и веб
- Prometheus — агент, база, веб, alertmanager
- InfluxData — telegraph (агент), influxdb, chronograph (web), kapacitor (alerts)
- StatsD — набор инструментов для сбора метрик
- Grafana — интерфейс для дашбордов и алертинга

# Мониторинг Kafka

# Архитектура Kafka



# Мониторинг Kafka

- Broker
  - Метрики Kafka
  - Метрики сервера
  - Метрики JVM garbage collection
- Producer
- Consumer
- Zookeeper

# Метрики брокера для просмотра

- Метрики для просмотра:
  - UnderReplicatedPartitions
  - IsrShrinksPerSec / IsrExpandsPerSec
  - TotalTimeMs
  - BytesInPerSec / BytesOutPerSec
  - RequestsPerSec
- Метрики для оповещений:
  - OfflinePartitionsCount (controller only)
  - UncleanLeaderElectionsPerSec

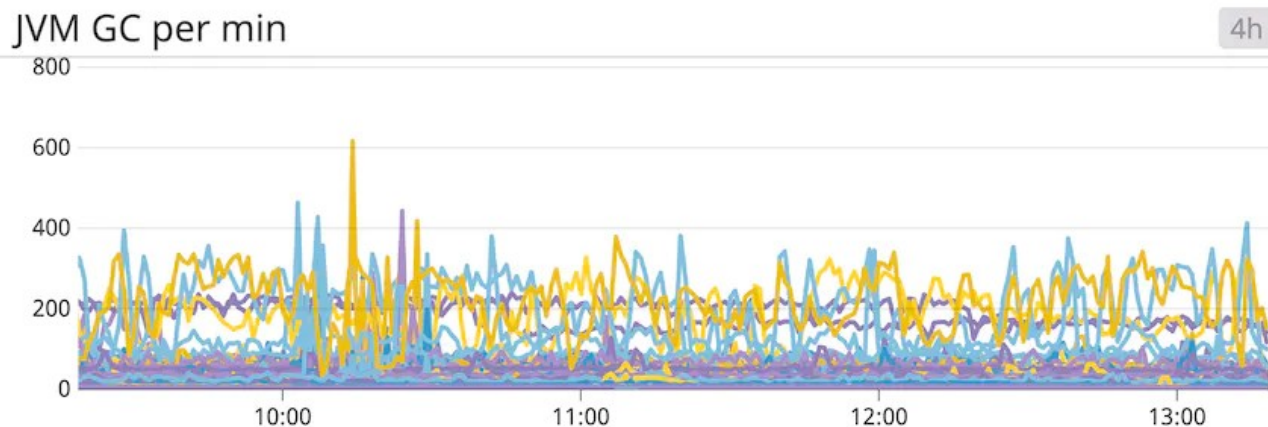
# Метрики сервера

- Page cache read ratio
- Disk usage
- CPU usage
- Network bytes sent/received



# Метрики JVM garbage collection

- Young generation garbage collection time
- Old generation garbage collection count/time



# Метрики Producer

JMX attribute	MBean name	Description	Metric type
compression-rate-avg	kafka.producer:type=producer-metrics,client-id=([-\.w]+)	Average compression rate of batches sent	Work: Other
response-rate	kafka.producer:type=producer-metrics,client-id=([-\.w]+)	Average number of responses received per second	Work: Throughput
request-rate	kafka.producer:type=producer-metrics,client-id=([-\.w]+)	Average number of requests sent per second	Work: Throughput
request-latency-avg	kafka.producer:type=producer-metrics,client-id=([-\.w]+)	Average request latency (in ms)	Work: Throughput
outgoing-byte-rate	kafka.producer:type=producer-metrics,client-id=([-\.w]+)	Average number of outgoing/incoming bytes per second	Work: Throughput
io-wait-time-ns-avg	kafka.producer:type=producer-metrics,client-id=([-\.w]+)	Average length of time the I/O thread spent waiting for a socket (in ns)	Work: Throughput
batch-size-avg	kafka.producer:type=producer-metrics,client-id=([-\.w]+)	The average number of bytes sent per partition per request	Work: Throughput





# Метрики Consumer

JMX attribute	MBean name	Description	Metric type
records-lag	kafka.consumer:type=consumer-fetch-manager-metrics,client-id=([-w]+),topic=([-w]+),partition=([-w]+)	Number of messages consumer is behind producer on this partition	Work: Performance
records-lag-max	kafka.consumer:type=consumer-fetch-manager-metrics,client-id=([-w]+),topic=([-w]+),partition=([-w]+) kafka.consumer:type=consumer-fetch-manager-metrics,client-id=([-w]+)	Maximum number of messages consumer is behind producer, either for a specific partition or across all partitions on this client	Work: Performance
bytes-consumed-rate	kafka.consumer:type=consumer-fetch-manager-metrics,client-id=([-w]+),topic=([-w]+) kafka.consumer:type=consumer-fetch-manager-metrics,client-id=([-w]+)	Average number of bytes consumed per second for a specific topic or across all topics.	Work: Throughput
records-consumed-rate	kafka.consumer:type=consumer-fetch-manager-metrics,client-id=([-w]+),topic=([-w]+) kafka.consumer:type=consumer-fetch-manager-metrics,client-id=([-w]+)	Average number of records consumed per second for a specific topic or across all topics	Work: Throughput
fetch-rate	kafka.consumer:type=consumer-fetch-manager-metrics,client_id=([-w]+)	Number of fetch requests per second from the consumer	Work: Throughput

# Метрики ZooKeeper

Name	Description	Metric type	Availability
outstanding_requests	Number of requests queued	Resource: Saturation	Four-letter words, AdminServer, JMX
avg_latency	Amount of time it takes to respond to a client request (in ms)	Work: Throughput	Four-letter words, AdminServer, JMX
num_alive_connections	Number of clients connected to ZooKeeper	Resource: Availability	Four-letter words, AdminServer, JMX
followers	Number of active followers	Resource: Availability	Four-letter words, AdminServer
pending_syncs	Number of pending syncs from followers	Other	Four-letter words, AdminServer, JMX
open_file_descriptor_count	Number of file descriptors in use	Resource: Utilization	Four-letter words, AdminServer

# Включаем мониторинг

# Отдельный сервер

- `bin/zookeeper-server-start.sh -daemon config/zookeeper.properties`
- `JMX_PORT=7071 bin/kafka-server-start.sh -daemon config/server.properties`

# Docker c ZooKeeper

```
docker run -d \  
--name=zookeeper-jmx \  
--net=host \  
-e ZOOKEEPER_TICK_TIME=2000 \  
-e ZOOKEEPER_CLIENT_PORT=32181 \  
-e KAFKA_JMX_PORT=9101 \  
confluentinc/cp-zookeeper:7.4.0
```

```
docker run -d \  
--name=kafka-jmx \  
--net=host \  
-e KAFKA_BROKER_ID=1 \  
-e KAFKA_ZOOKEEPER_CONNECT=localhost:32181/jmx \  
-e KAFKA_ADVERTISED_LISTENERS=PLAINTEXT://localhost:39092 \  
-e KAFKA_JMX_PORT=9101 \  
-e KAFKA_OFFSETS_TOPIC_REPLICATION_FACTOR=1 \  
confluentinc/cp-kafka:7.4.0
```



# Docker c KRaft

```
docker run -d \  
--name=kafka-jmx \  
-h kafka-jmx \  
-p 9101:9101 \  
-p 9092:9092 \  
-e KAFKA_NODE_ID=1 \  
-e KAFKA_LISTENER_SECURITY_PROTOCOL_MAP='CONTROLLER:PLAINTEXT,PLAINTEXT:PLAINTEXT,PLAINTEXT_HOST:PLAINTEXT' \  
-e KAFKA_ADVERTISED_LISTENERS='PLAINTEXT://kafka-jmx:29092,PLAINTEXT_HOST://localhost:9092' \  
-e KAFKA_JMX_PORT=9101 \  
-e KAFKA_JMX_HOSTNAME=localhost \  
-e KAFKA_PROCESS_ROLES='broker,controller' \  
-e KAFKA_OFFSETS_TOPIC_REPLICATION_FACTOR=1 \  
-e KAFKA_CONTROLLER_QUORUM_VOTERS='1@kafka-jmx:29093' \  
-e KAFKA_LISTENERS='PLAINTEXT://kafka-jmx:29092,CONTROLLER://kafka-jmx:29093,PLAINTEXT_HOST://0.0.0.0:9092' \  
-e KAFKA_INTER_BROKER_LISTENER_NAME='PLAINTEXT' \  
-e KAFKA_CONTROLLER_LISTENER_NAMES='CONTROLLER' \  
-e CLUSTER_ID='MkU30EVBNTcwNTJENDM2Qk' \  
confluentinc/cp-kafka:7.4.0
```



# Kafka с ZooKeeper как сервис

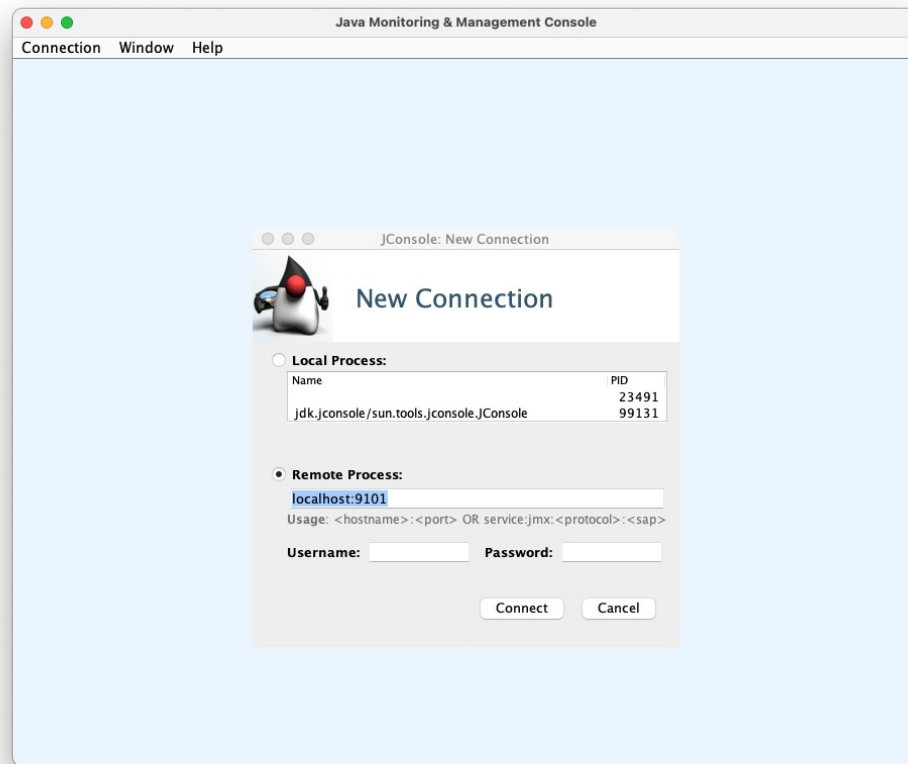
- *zookeeper.service*

```
Environment="KAFKA_JMX_OPTS=-Dcom.sun.management.jmxremote -Dcom.sun.management.jmxremote.port=10020 -  
Dcom.sun.management.jmxremote.local.only=false -Dcom.sun.management.jmxremote.authenticate=false"
```

- *kafka.service*

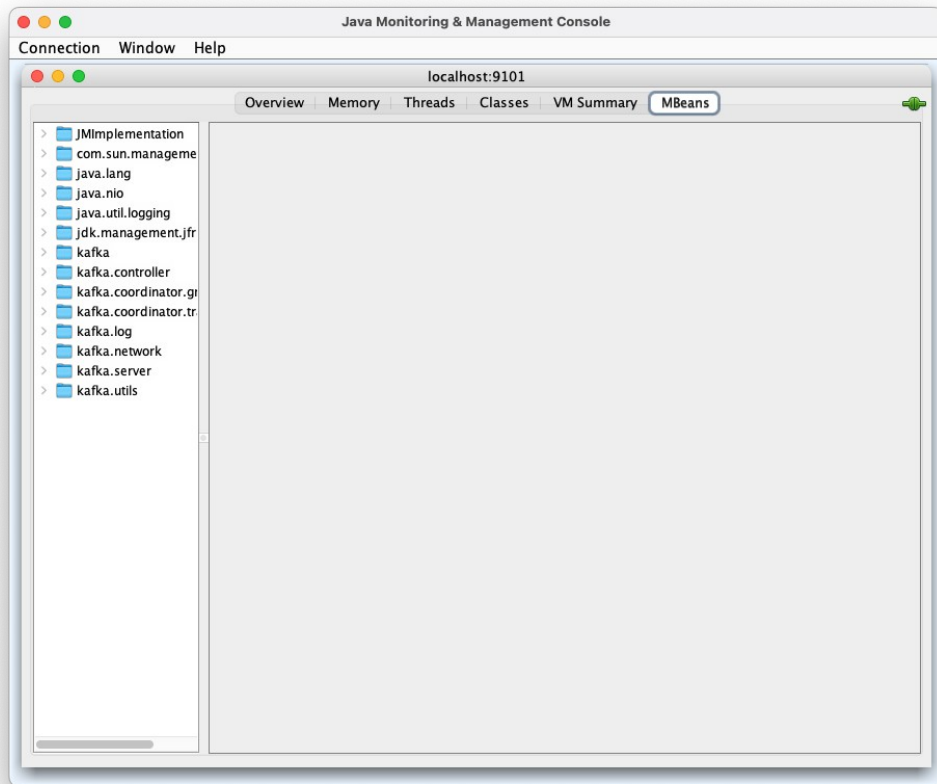
```
Environment="KAFKA_JMX_OPTS=-Dcom.sun.management.jmxremote -Dcom.sun.management.jmxremote.port=10030 -  
Dcom.sun.management.jmxremote.local.only=false -Dcom.sun.management.jmxremote.authenticate=false"
```

# JConsole





# Просмотр параметров в JConsole



# ZooKeeper AdminServer

- *config/zookeeper.properties*

admin.enableServer=true

admin.serverPort=8080

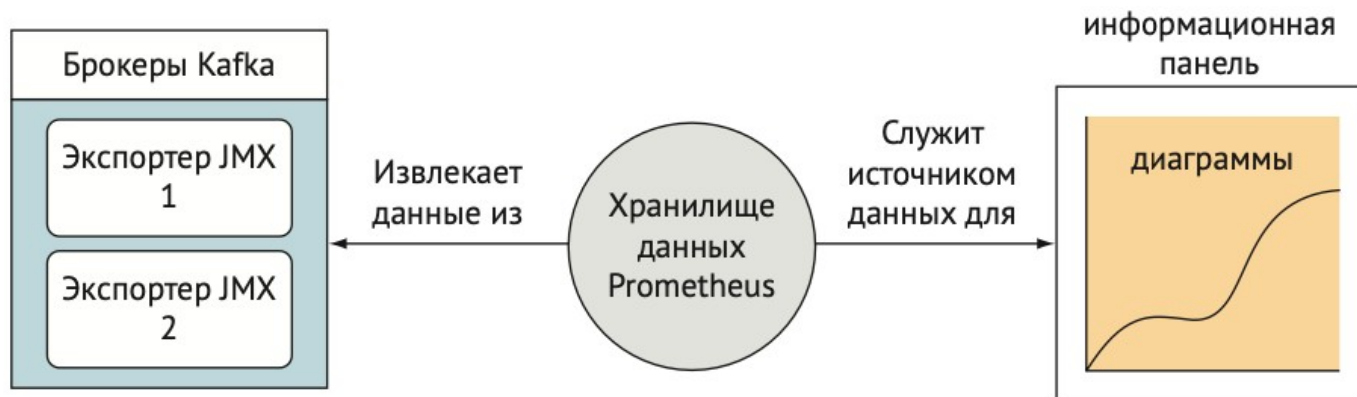
- `curl http://localhost:8080/commands`
- `curl http://localhost:8080/commands/monitor`
- `curl http://localhost:8080/commands/stats`

# LIVE

# Промышленный мониторинг

# Промышленный мониторинг

- Prometheus — хранение метрик
- Kafka exporter — извлечение метрик
- JMX exporter — извлечение метрик
- Grafana — визуализация метрик
- Пример дашборда для Kafka: <https://grafana.com/grafana/dashboards/721-kafka/>



# Установка и настройка Kafka с JMX exporter

- Устанавливаем и настраиваем Kafka
- Устанавливаем Prometheus JMX exporter:
  - `cd libs`
  - `wget https://repo1.maven.org/maven2/io/prometheus/jmx/jmx_prometheus_javaagent/0.18.0/jmx_prometheus_javaagent-0.18.0.jar`
  - `wget https://github.com/prometheus/jmx_exporter/raw/main/example_configs/kafka-0-8-2.yml`
- Запускаем ZooKeeper:
  - `bin/zookeeper-server-start.sh -daemon config/zookeeper.properties`
- Запускаем Kafka с JMX exporter:
  - `KAFKA_OPTS="$KAFKA_OPTS -javaagent:$PWD/libs/jmx_prometheus_javaagent-0.18.0.jar=7071:$PWD/libs/kafka-0-8-2.yml" bin/kafka-server-start.sh -daemon config/server.properties`

# Prometheus

- Устанавливаем и настраиваем Prometheus:
  - `cd opt`
  - `sudo wget https://github.com/prometheus/prometheus/releases/download/v2.44.0/prometheus-2.44.0.Linux-amd64.tar.gz`
  - `sudo tar xvzf prometheus-2.44.0.Linux-amd64.tar.gz`
  - `cd prometheus-2.44.0.Linux-amd64`
  - `sudo vi prometheus.yml`

global:

scrape\_interval: 10s

evaluation\_interval: 10s

scrape\_configs:

- job\_name: 'kafka'

static\_configs:

- targets:

- kafka1:7071

- kafka2:7071

- kafka3:7071

# Grafana

- Устанавливаем и настраиваем Grafana:
  - `cd opt`
  - `sudo wget https://dl.grafana.com/oss/release/grafana-9.5.2.linux-amd64.tar.gz`
  - `sudo tar xzvf grafana-9.5.2.linux-amd64.tar.gz`
- Запускаем Grafana:
  - `cd grafana-9.5.2`
  - `sudo ./bin/grafana-server web`
- Добавляем Prometheus data source:
  - Click on the Grafana logo to open the sidebar menu.
  - Look for "Data Source" in the sidebar.
  - Click on "Add New".
  - Select "Prometheus" as a type.
  - Select the Prometheus server URL(`http://localhost:9090/`).
  - Click Add to save the new DataSource



# Grafana



# LIVE

# Специализированные системы

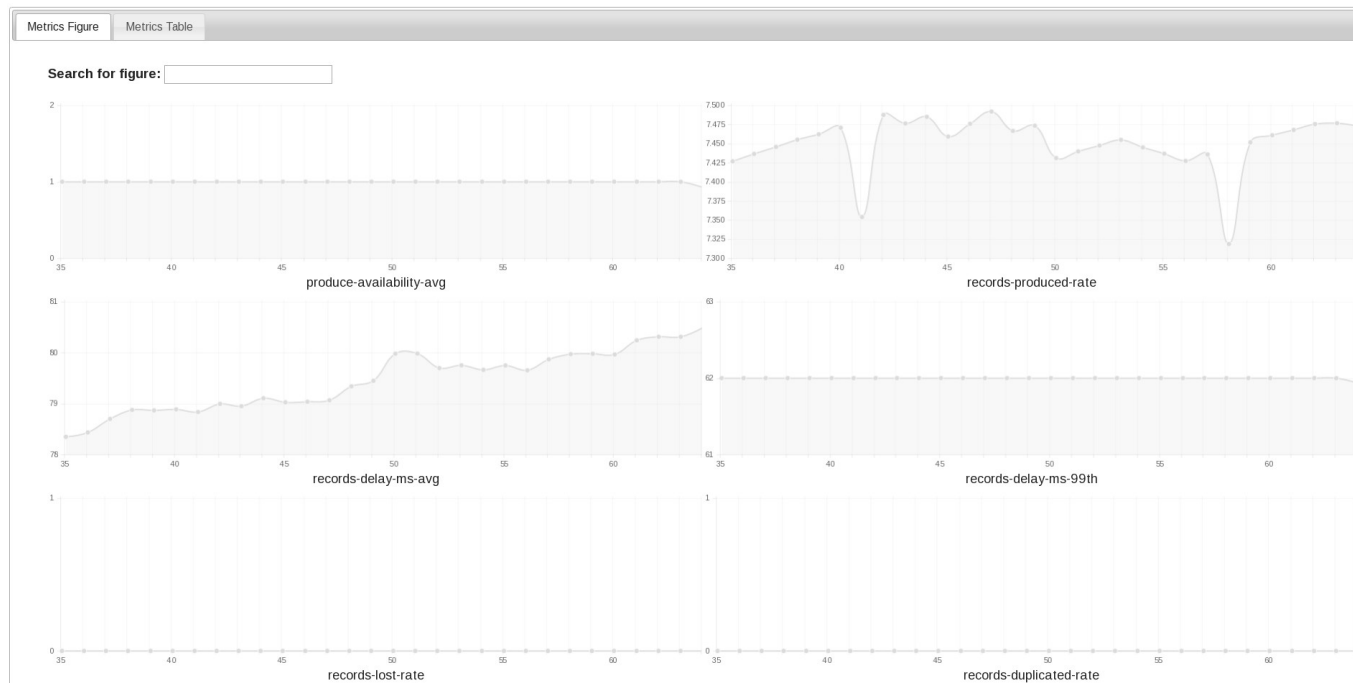
# Специализированные системы

	AKHQ	Kowl	Kafdrop	UI for Apache Kafka	Lenses	CMAK	Confluent CC	Conduktor
Multi-Cluster Management	Yes	No	No	Yes	Yes	Yes	Yes	No
Message Browsing	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes
Protobuf Support	Partial	Yes	Yes	Yes	No	No	Yes	Yes
Avro Support	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes
Live Tailing	Yes	Yes	No	Roadmap	No	No	Yes	Yes
Dynamic Topic Configuration	No	No	No	Yes	No	Yes	Yes	Yes
Authentication	Yes	Paid	No	Yes	No	No	Yes	No
Authorization	Yes	Paid	No	Yes	No	No	Yes	No
Partition Increase	No	No	No	Yes	No	Yes	Yes	Yes
Replica Change	No	No	No	Yes	No	Yes	Yes	Yes
Amazon MSK IAM support	No	Yes	No	Yes	No	No	No	Yes
Kafka Connect management	Yes	No	No	Yes	Separate Service	No	Yes	Yes
UI Quality	Bad	Great	Average	Good	Great	Good	Good	Good
Schema Registry	Yes	No	No	Yes	No	No	Yes	Yes
KSQL Integration	No	No	No	Yes	No	No	Yes	Yes
Kafka Streams Topologies	No	No	No	Roadmap	No	No	Yes	No
Read-Only Mode	Yes	No	No	Yes	No	No	Yes	Yes
JMX Metrics Visualization and charts	No	No	No	Roadmap	No	No	Yes	Yes

# Xinfra Monitor

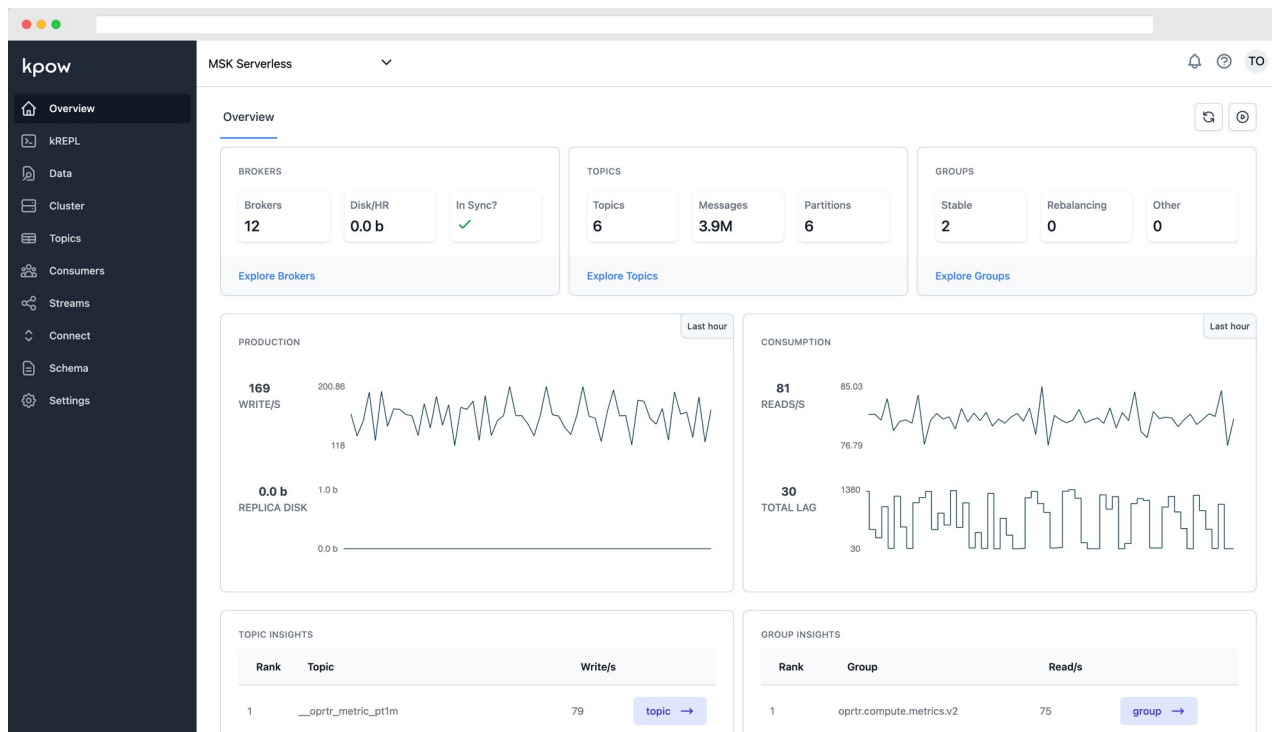
<https://github.com/linkedin/kafka-monitor>

## Kafka Monitor GUI



# Kpow

- <https://kpow.io>
- <https://github.com/factorhouse/kpow-local>

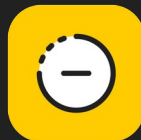


# LIVE

# Вопросы?



Ставим “+”,  
если вопросы есть



Ставим “-”,  
если вопросов нет





# Литература

# Список материалов для изучения

1. Kafka Monitoring
2. Monitoring Kafka performance metrics
3. Collecting Kafka performance metrics
4. Monitoring Kafka with JMX
5. Prometheus Exporters and integrations
6. Kafka exporter for Prometheus
7. JMX Exporter
8. Kafka Dashboard
9. Xinfra Monitor
10. Kpow

# Рефлексия

# Рефлексия



С какими впечатлениями уходите с вебинара?



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

# Следующий вебинар



## Безопасность



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](mailto:vzaigrin@yandex.ru)

<https://t.me/vzaigrin>

