



МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО
ОБРАЗОВАНИЯ «ЛИПЕЦКИЙ ГОСУДАРСТВЕННЫЙ
ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Институт компьютерных наук
Кафедра автоматизированных систем управления

Лабораторная работа №3
по дисциплине «Теория обработки больших массивов данных»

Студент М-РИТ-25

Станиславчук С. М.

(подпись, дата)

Руководитель

Доцент, к.т.н.

Тюрин А. С.

(подпись, дата)

Липецк 2025

Содержание

1. Задание кафедры.....	3
2. Ход работы.....	3
3. Вывод.....	6
Приложение 1. docker-compose.yml.....	7
Приложение 2. Скрипт consumer.py.....	7
Приложение 3. Скрипт producer.py.....	9

1. Задание кафедры

Развернуть Kafka и Kafdrop, проверить работоспособность конфигурации, проверить доступность веб-интерфейса Kafdrop. Используя скрипт, создать консьюмера и продюсера, создать топик и отправить в него сообщения, после чего прочитать его.

Цель: изучить технологию создания и использования очередей сообщения, чтения и записи сообщений в очереди.

2. Ход работы

**** В ходе выполнения работы использовалось ядро GNU/Linux v6.15.4, Docker v28.3.0, Kafka v7.0.1, Zookeeper v7.0.1, Kafdrop v4.2.0, Python v3.12.9 ****

docker-compose.yml (добавлен слушатель для работы внутри kafka: 29092, хост все еще на 9092):

```
services:
  zookeeper:
    image: confluentinc/cp-zookeeper:7.0.1
    environment:
      ZOOKEEPER_CLIENT_PORT: 2181
      ZOOKEEPER_TICK_TIME: 2000
    ports:
      - "2181:2181"

  kafka:
    image: confluentinc/cp-kafka:7.0.1
    depends_on:
      - zookeeper
    ports:
      - "9092:9092"
      - "29092:29092"
    environment:
      KAFKA_BROKER_ID: 1
      KAFKA_ZOOKEEPER_CONNECT: zookeeper:2181
      KAFKA_ADVERTISED_LISTENERS: PLAINTEXT://kafka:29092,PLAINTEXT_HOST://localhost:9092
      KAFKA_LISTENER_SECURITY_PROTOCOL_MAP: PLAINTEXT:PLAINTEXT,PLAINTEXT_HOST:PLAINTEXT
      KAFKA_INTER_BROKER_LISTENER_NAME: PLAINTEXT
      KAFKA_OFFSETS_TOPIC_REPLICATION_FACTOR: 1

  kafdrop:
    image: obsidiandynamics/kafdrop:latest
    depends_on:
      - kafka
    ports:
      - "9000:9000"
```

```
environment:
  KAFKA_BROKER_CONNECT: kafka:29092
  JVM_OPTS: "-Xms32M -Xmx64M"
  SERVER_SERVLET_CONTEXTPATH: "/"
```

Все контейнеры были запущены


```
[+] Running 4/4
✓ Network src_default          Created
✓ Container src-zookeeper-1    Started
✓ Container src-kafka-1        Started
✓ Container src-kafdrop-1      Started
```

и успешно функционируют

```
stanik@archlinux: /home/stanik/programmer/Programmer/5_1/BIG_DATA/lab/3/src $ sudo docker ps
CONTAINER ID   IMAGE                                COMMAND                  CREATED        STATUS        PORTS                               NAMES
87e225137e17   obsidiandynamics/kafdrop:latest    "/kafdrop.sh"          About a minute ago    Up About a minute    0.0.0.0:9000->9000/tcp, [::]:9000->9000/tcp    src-kafdrop-1
6cfb6800b07f   confluentinc/cp-kafka:7.0.1        "/etc/confluent/docker" About a minute ago    Up About a minute    0.0.0.0:9092->9092/tcp, [::]:9092->9092/tcp    src-kafka-1
34a05d82d808   confluentinc/cp-zookeeper:7.0.1    "/etc/confluent/docker" About a minute ago    Up About a minute    2888/tcp, 0.0.0.0:2181->2181/tcp, [::]:2181->2181/tcp, 3888/tcp    src-zookeeper-1
```

Следующий шаг: запустить скрипты consumer.py и producer.py и посмотреть отправку и получение сообщений.

Web-UI Kafdrop после создания топика:

Star

4.2.0 [2025-07-31T09:48:58.619Z]

Kafka Cluster Overview

Bootstrap servers	kafka:29092
Total topics	2
Total partitions	51
Total preferred partition leader	100%
Total under-replicated partitions	0

Brokers

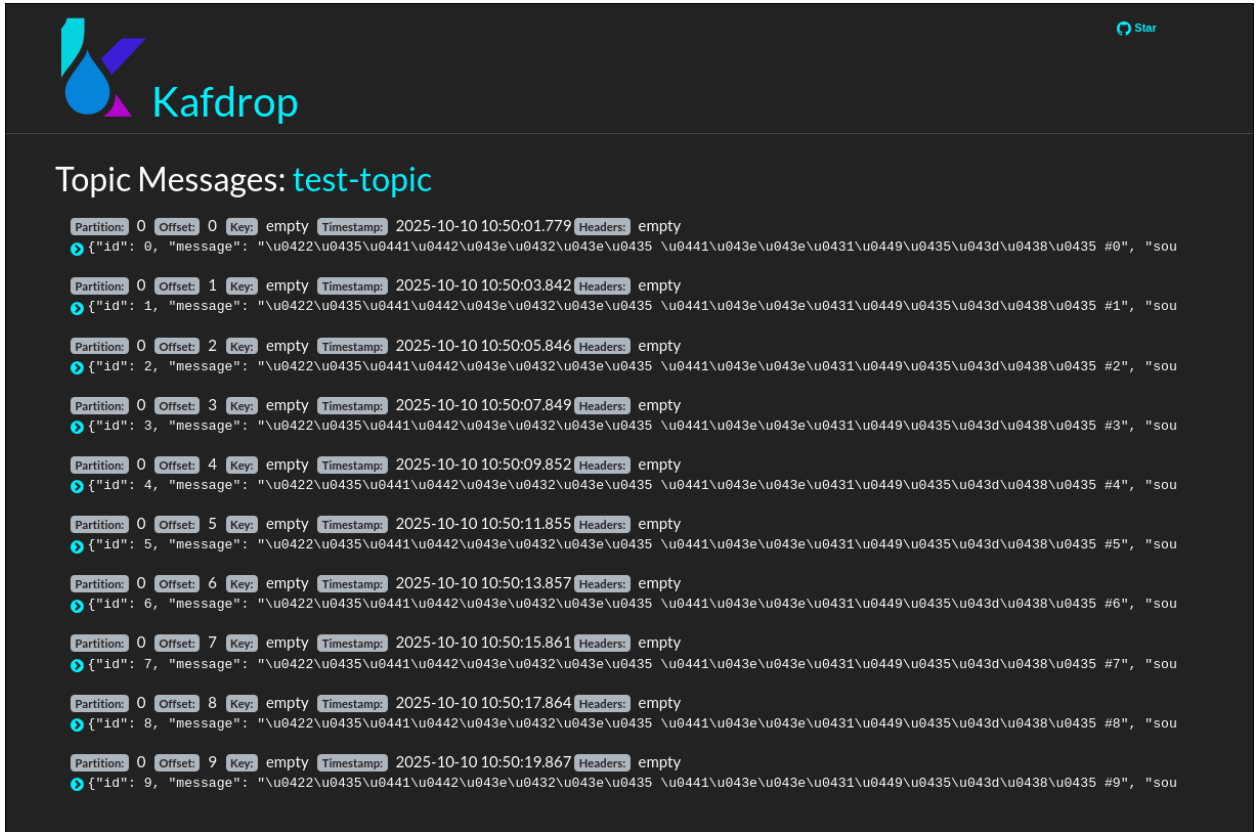
ID	Host	Port	Rack	Controller	Number of partitions (% of total)
1	kafka	29092	-	Yes	51 (100%)

Topics [ACLs](#)

Name	Partitions	% Preferred	# Under-replicated	Custom Config
_consumer_offsets	50	100%	0	Yes
test-topic	1	100%	0	No

[+ New](#)

Web-UI Kafdrop сообщения созданного топика (получены корректно):



The screenshot displays the Kafdrop Web-UI interface. At the top left is the Kafdrop logo, and at the top right is a 'Star' icon. The main heading is 'Topic Messages: test-topic'. Below this, a list of 10 messages is shown, each with its partition, offset, key, timestamp, and headers. The messages are displayed in a dark theme with light text. Each message is preceded by a small blue icon with a white 'x'.

Topic Messages: test-topic

Partition: 0 Offset: 0 Key: empty Timestamp: 2025-10-10 10:50:01.779 Headers: empty
{"id": 0, "message": "\u0042\u0035\u0041\u0042\u003e\u0032\u003e\u00435 \u0041\u003e\u00431\u0049\u0035\u003d\u0038\u00435 #0", "sou

Partition: 0 Offset: 1 Key: empty Timestamp: 2025-10-10 10:50:03.842 Headers: empty
{"id": 1, "message": "\u0042\u0035\u0041\u0042\u003e\u0032\u003e\u00435 \u0041\u003e\u00431\u0049\u0035\u003d\u0038\u00435 #1", "sou

Partition: 0 Offset: 2 Key: empty Timestamp: 2025-10-10 10:50:05.846 Headers: empty
{"id": 2, "message": "\u0042\u0035\u0041\u0042\u003e\u0032\u003e\u00435 \u0041\u003e\u00431\u0049\u0035\u003d\u0038\u00435 #2", "sou

Partition: 0 Offset: 3 Key: empty Timestamp: 2025-10-10 10:50:07.849 Headers: empty
{"id": 3, "message": "\u0042\u0035\u0041\u0042\u003e\u0032\u003e\u00435 \u0041\u003e\u00431\u0049\u0035\u003d\u0038\u00435 #3", "sou

Partition: 0 Offset: 4 Key: empty Timestamp: 2025-10-10 10:50:09.852 Headers: empty
{"id": 4, "message": "\u0042\u0035\u0041\u0042\u003e\u0032\u003e\u00435 \u0041\u003e\u00431\u0049\u0035\u003d\u0038\u00435 #4", "sou

Partition: 0 Offset: 5 Key: empty Timestamp: 2025-10-10 10:50:11.855 Headers: empty
{"id": 5, "message": "\u0042\u0035\u0041\u0042\u003e\u0032\u003e\u00435 \u0041\u003e\u00431\u0049\u0035\u003d\u0038\u00435 #5", "sou

Partition: 0 Offset: 6 Key: empty Timestamp: 2025-10-10 10:50:13.857 Headers: empty
{"id": 6, "message": "\u0042\u0035\u0041\u0042\u003e\u0032\u003e\u00435 \u0041\u003e\u00431\u0049\u0035\u003d\u0038\u00435 #6", "sou

Partition: 0 Offset: 7 Key: empty Timestamp: 2025-10-10 10:50:15.861 Headers: empty
{"id": 7, "message": "\u0042\u0035\u0041\u0042\u003e\u0032\u003e\u00435 \u0041\u003e\u00431\u0049\u0035\u003d\u0038\u00435 #7", "sou

Partition: 0 Offset: 8 Key: empty Timestamp: 2025-10-10 10:50:17.864 Headers: empty
{"id": 8, "message": "\u0042\u0035\u0041\u0042\u003e\u0032\u003e\u00435 \u0041\u003e\u00431\u0049\u0035\u003d\u0038\u00435 #8", "sou

Partition: 0 Offset: 9 Key: empty Timestamp: 2025-10-10 10:50:19.867 Headers: empty
{"id": 9, "message": "\u0042\u0035\u0041\u0042\u003e\u0032\u003e\u00435 \u0041\u003e\u00431\u0049\u0035\u003d\u0038\u00435 #9", "sou

stdout скриптов (вверху консьюмер, внизу продюсер):

```
-----
Получено сообщение: {'id': 27, 'message': 'Тестовое сообщение #27', 'source': 'python-producer', 'timestamp': '2025-10-10T13:50:55.924128'}
Топик: test-topic, Партия: 0
Offset: 27, Timestamp: 1760093455924
-----
Получено сообщение: {'id': 28, 'message': 'Тестовое сообщение #28', 'source': 'python-producer', 'timestamp': '2025-10-10T13:50:57.928500'}
Топик: test-topic, Партия: 0
Offset: 28, Timestamp: 1760093457928
-----
Получено сообщение: {'id': 29, 'message': 'Тестовое сообщение #29', 'source': 'python-producer', 'timestamp': '2025-10-10T13:50:59.932107'}
Топик: test-topic, Партия: 0
Offset: 29, Timestamp: 1760093459932
-----
Получено сообщение: {'id': 30, 'message': 'Тестовое сообщение #30', 'source': 'python-producer', 'timestamp': '2025-10-10T13:51:01.934889'}
Топик: test-topic, Партия: 0
Offset: 30, Timestamp: 1760093461935
-----
Получено сообщение: {'id': 31, 'message': 'Тестовое сообщение #31', 'source': 'python-producer', 'timestamp': '2025-10-10T13:51:03.938933'}
Топик: test-topic, Партия: 0
Offset: 31, Timestamp: 1760093463939
-----
Получено сообщение: {'id': 32, 'message': 'Тестовое сообщение #32', 'source': 'python-producer', 'timestamp': '2025-10-10T13:51:05.941625'}
Топик: test-topic, Партия: 0
Offset: 32, Timestamp: 1760093465941
-----
Получено сообщение: {'id': 33, 'message': 'Тестовое сообщение #33', 'source': 'python-producer', 'timestamp': '2025-10-10T13:51:07.944875'}
Топик: test-topic, Партия: 0
Offset: 33, Timestamp: 1760093467945
-----
Получено сообщение: {'id': 34, 'message': 'Тестовое сообщение #34', 'source': 'python-producer', 'timestamp': '2025-10-10T13:51:09.947744'}
Топик: test-topic, Партия: 0
Offset: 34, Timestamp: 1760093469947
-----
*Остановка консьюмера..
stanik@archlinux: /home/stanik/programmer/++Programmer/5_1/BIG_DATA/lab/3/src >

Отправлено сообщение: {'id': 3, 'message': 'Тестовое сообщение #3', 'source': 'python-producer', 'timestamp': '2025-10-10T13:50:07.849534'}
Отправлено сообщение: {'id': 4, 'message': 'Тестовое сообщение #4', 'source': 'python-producer', 'timestamp': '2025-10-10T13:50:09.852430'}
Отправлено сообщение: {'id': 5, 'message': 'Тестовое сообщение #5', 'source': 'python-producer', 'timestamp': '2025-10-10T13:50:11.855217'}
Отправлено сообщение: {'id': 6, 'message': 'Тестовое сообщение #6', 'source': 'python-producer', 'timestamp': '2025-10-10T13:50:13.857712'}
Отправлено сообщение: {'id': 7, 'message': 'Тестовое сообщение #7', 'source': 'python-producer', 'timestamp': '2025-10-10T13:50:15.861397'}
Отправлено сообщение: {'id': 8, 'message': 'Тестовое сообщение #8', 'source': 'python-producer', 'timestamp': '2025-10-10T13:50:17.864306'}
Отправлено сообщение: {'id': 9, 'message': 'Тестовое сообщение #9', 'source': 'python-producer', 'timestamp': '2025-10-10T13:50:19.867135'}
Отправлено сообщение: {'id': 10, 'message': 'Тестовое сообщение #10', 'source': 'python-producer', 'timestamp': '2025-10-10T13:50:21.870445'}
Отправлено сообщение: {'id': 11, 'message': 'Тестовое сообщение #11', 'source': 'python-producer', 'timestamp': '2025-10-10T13:50:23.874036'}
Отправлено сообщение: {'id': 12, 'message': 'Тестовое сообщение #12', 'source': 'python-producer', 'timestamp': '2025-10-10T13:50:25.877066'}
Отправлено сообщение: {'id': 13, 'message': 'Тестовое сообщение #13', 'source': 'python-producer', 'timestamp': '2025-10-10T13:50:27.879670'}
Отправлено сообщение: {'id': 14, 'message': 'Тестовое сообщение #14', 'source': 'python-producer', 'timestamp': '2025-10-10T13:50:29.882104'}
Отправлено сообщение: {'id': 15, 'message': 'Тестовое сообщение #15', 'source': 'python-producer', 'timestamp': '2025-10-10T13:50:31.884810'}
Отправлено сообщение: {'id': 16, 'message': 'Тестовое сообщение #16', 'source': 'python-producer', 'timestamp': '2025-10-10T13:50:33.888219'}
Отправлено сообщение: {'id': 17, 'message': 'Тестовое сообщение #17', 'source': 'python-producer', 'timestamp': '2025-10-10T13:50:35.892765'}
Отправлено сообщение: {'id': 18, 'message': 'Тестовое сообщение #18', 'source': 'python-producer', 'timestamp': '2025-10-10T13:50:37.896574'}
Отправлено сообщение: {'id': 19, 'message': 'Тестовое сообщение #19', 'source': 'python-producer', 'timestamp': '2025-10-10T13:50:39.899600'}
Отправлено сообщение: {'id': 20, 'message': 'Тестовое сообщение #20', 'source': 'python-producer', 'timestamp': '2025-10-10T13:50:41.902437'}
Отправлено сообщение: {'id': 21, 'message': 'Тестовое сообщение #21', 'source': 'python-producer', 'timestamp': '2025-10-10T13:50:43.904817'}
Отправлено сообщение: {'id': 22, 'message': 'Тестовое сообщение #22', 'source': 'python-producer', 'timestamp': '2025-10-10T13:50:45.908012'}
Отправлено сообщение: {'id': 23, 'message': 'Тестовое сообщение #23', 'source': 'python-producer', 'timestamp': '2025-10-10T13:50:47.912176'}
Отправлено сообщение: {'id': 24, 'message': 'Тестовое сообщение #24', 'source': 'python-producer', 'timestamp': '2025-10-10T13:50:49.915012'}
Отправлено сообщение: {'id': 25, 'message': 'Тестовое сообщение #25', 'source': 'python-producer', 'timestamp': '2025-10-10T13:50:51.917919'}
Отправлено сообщение: {'id': 26, 'message': 'Тестовое сообщение #26', 'source': 'python-producer', 'timestamp': '2025-10-10T13:50:53.920808'}
Отправлено сообщение: {'id': 27, 'message': 'Тестовое сообщение #27', 'source': 'python-producer', 'timestamp': '2025-10-10T13:50:55.924128'}
Отправлено сообщение: {'id': 28, 'message': 'Тестовое сообщение #28', 'source': 'python-producer', 'timestamp': '2025-10-10T13:50:57.928500'}
Отправлено сообщение: {'id': 29, 'message': 'Тестовое сообщение #29', 'source': 'python-producer', 'timestamp': '2025-10-10T13:50:59.932107'}
Отправлено сообщение: {'id': 30, 'message': 'Тестовое сообщение #30', 'source': 'python-producer', 'timestamp': '2025-10-10T13:51:01.934889'}
Отправлено сообщение: {'id': 31, 'message': 'Тестовое сообщение #31', 'source': 'python-producer', 'timestamp': '2025-10-10T13:51:03.938933'}
Отправлено сообщение: {'id': 32, 'message': 'Тестовое сообщение #32', 'source': 'python-producer', 'timestamp': '2025-10-10T13:51:05.941625'}
Отправлено сообщение: {'id': 33, 'message': 'Тестовое сообщение #33', 'source': 'python-producer', 'timestamp': '2025-10-10T13:51:07.944875'}
Отправлено сообщение: {'id': 34, 'message': 'Тестовое сообщение #34', 'source': 'python-producer', 'timestamp': '2025-10-10T13:51:09.947744'}
Отправлено сообщение: {'id': 35, 'message': 'Тестовое сообщение #35', 'source': 'python-producer', 'timestamp': '2025-10-10T13:51:11.950232'}
-----
*Остановка продюсера..
stanik@archlinux: /home/stanik/programmer/++Programmer/5_1/BIG_DATA/lab/3/src >
```

3. Вывод

В ходе выполнения лабораторной работы была изучена технология создания и использования очередей сообщения, в том числе создания издателей и подписчиков, чтения и записи сообщений в очереди.

Был изучен инструмент Kafdgor, предоставляющий web-интерфейс для взаимодействия с Kafka.

Приложение 1. docker-compose.yml

```
services:
  zookeeper:
    image: confluentinc/cp-zookeeper:7.0.1
    environment:
      ZOOKEEPER_CLIENT_PORT: 2181
      ZOOKEEPER_TICK_TIME: 2000
    ports:
      - "2181:2181"

  kafka:
    image: confluentinc/cp-kafka:7.0.1
    depends_on:
      - zookeeper
    ports:
      - "9092:9092"
      - "29092:29092"
    environment:
      KAFKA_BROKER_ID: 1
      KAFKA_ZOOKEEPER_CONNECT: zookeeper:2181
      KAFKA_ADVERTISED_LISTENERS: PLAINTEXT://kafka:29092,PLAINTEXT_HOST://localhost:9092
      KAFKA_LISTENER_SECURITY_PROTOCOL_MAP: PLAINTEXT:PLAINTEXT,PLAINTEXT_HOST:PLAINTEXT
      KAFKA_INTER_BROKER_LISTENER_NAME: PLAINTEXT
      KAFKA_OFFSETS_TOPIC_REPLICATION_FACTOR: 1

  kafdrop:
    image: obsidiandynamics/kafdrop:latest
    depends_on:
      - kafka
    ports:
      - "9000:9000"
    environment:
      KAFKA_BROKER_CONNECT: kafka:29092
      JVM_OPTS: "-Xms32M -Xmx64M"
      SERVER_SERVLET_CONTEXTPATH: "/"
```

Приложение 2. Скрипт consumer.py

```
from kafka import KafkaConsumer
import json

# Настройки подключения
KAFKA_BROKER = 'localhost:9092'
TOPIC_NAME = 'test-topic'

# Создание консьюмера
consumer = KafkaConsumer(
    TOPIC_NAME,
    bootstrap_servers=[KAFKA_BROKER],
    auto_offset_reset='earliest', # начинать чтение с начала топика
    enable_auto_commit=True,
    group_id='python-consumer-group',
```

```

        value_deserializer=lambda x: json.loads(x.decode('utf-8'))
    )

def consume_messages():
    """Чтение сообщений из Kafka"""
    print("Запуск консьюмера Kafka...")
    print("Ожидание сообщений. Для остановки нажмите Ctrl+C")

    try:
        for message in consumer:
            value = message.value
            print(f"Получено сообщение: {value}")
            print(f"Топик: {message.topic}, Партиция: {message.partition}")
            print(f"Offset: {message.offset}, Timestamp: {message.timestamp}")
            print("-" * 50)

    except KeyboardInterrupt:
        print("Остановка консьюмера...")
    finally:
        consumer.close()

if __name__ == "__main__":
    consume_messages()

```


Приложение 3. Скрипт producer.py

```
from kafka import KafkaProducer
import json
import time
from datetime import datetime

# Настройки подключения
KAFKA_BROKER = 'localhost:9092'
TOPIC_NAME = 'test-topic'

# Создание продюсера
producer = KafkaProducer(
    bootstrap_servers=[KAFKA_BROKER],
    value_serializer=lambda x: json.dumps(x).encode('utf-8')
)

def send_message(message):
    """Отправка сообщения в Kafka"""
    try:
        # Добавляем временную метку
        message['timestamp'] = datetime.now().isoformat()

        # Отправляем сообщение
        producer.send(TOPIC_NAME, value=message)
        producer.flush()

        print(f"Отправлено сообщение: {message}")
        return True
    except Exception as e:
        print(f"Ошибка при отправке: {e}")
        return False

if __name__ == "__main__":
    print("Запуск продюсера Kafka...")
    print("Для остановки нажмите Ctrl+C")

    try:
        counter = 0
        while True:
            # Формируем тестовое сообщение
            message = {
                'id': counter,
                'message': f'Тестовое сообщение #{counter}',
                'source': 'python-producer'
            }

            # Отправляем сообщение
            send_message(message)

            # Увеличиваем счетчик и ждем
            counter += 1
            time.sleep(2)

    except KeyboardInterrupt:
```

```
        print("Остановка продюсера...")
finally:
    producer.close()
```