

# Przetwarzanie strumieni danych i data science, Zajęcia Zintegrowane 1

Krzysztof Rudnicki, 307585

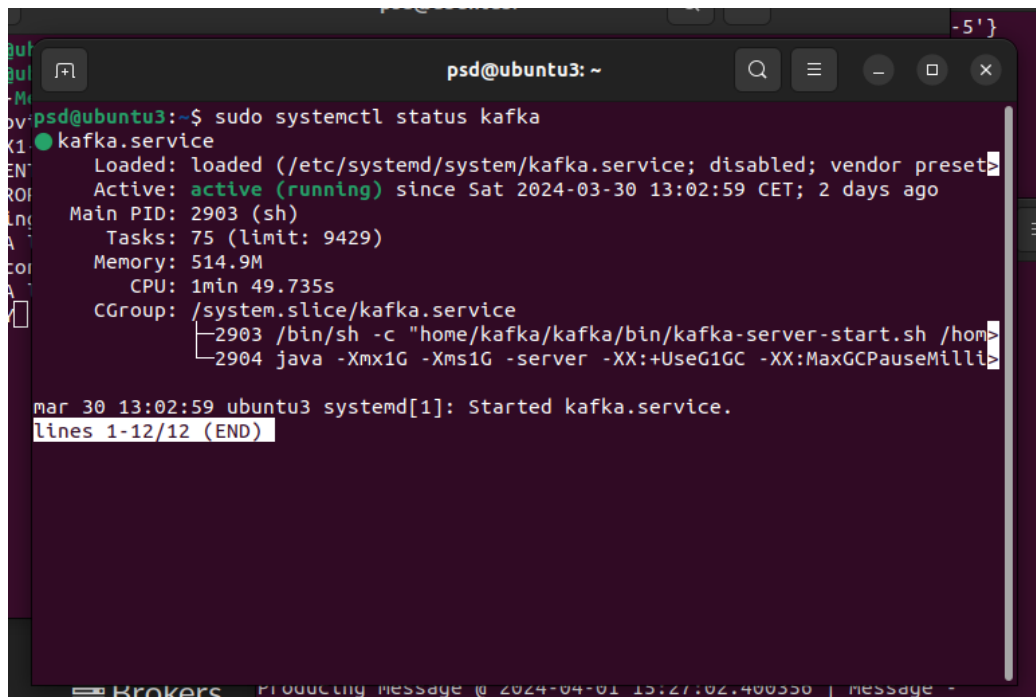
27 marca 2025

## 1 Przygotowanie maszyny wirtualnej

Przekopiowałem i uruchomiłem maszynę wirtualną z pliku .ova

Uruchomiłem serwis kafka korzystając z systemctl

Zweryfikowałem, że kafka działa (zarówno systemctl) jak i Hello World

A screenshot of a terminal window titled 'psd@ubuntu3: ~'. The terminal shows the command 'sudo systemctl status kafka' and its output. The output indicates that 'kafka.service' is loaded from '/etc/systemd/system/kafka.service', is currently 'active (running)', and was started on 'Sat 2024-03-30 13:02:59 CET'. It also shows the main PID as 2903 (sh) and lists tasks with their PIDs and commands. At the bottom, it states 'mar 30 13:02:59 ubuntu3 systemd[1]: Started kafka.service.' and 'lines 1-12/12 (END)'.

```
psd@ubuntu3:~$ sudo systemctl status kafka
● kafka.service
   Loaded: loaded (/etc/systemd/system/kafka.service; disabled; vendor preset: enabled)
   Active: active (running) since Sat 2024-03-30 13:02:59 CET; 2 days ago
     Main PID: 2903 (sh)
        Tasks: 75 (limit: 9429)
      Memory: 514.9M
         CPU: 1min 49.735s
       CGroup: /system.slice/kafka.service
              └─2903 /bin/sh -c "home/kafka/kafka/bin/kafka-server-start.sh /home/kafka/kafka/conf/kafka.properties"
                └─2904 java -Xmx1G -Xms1G -server -XX:+UseG1GC -XX:MaxGCPauseMilli>

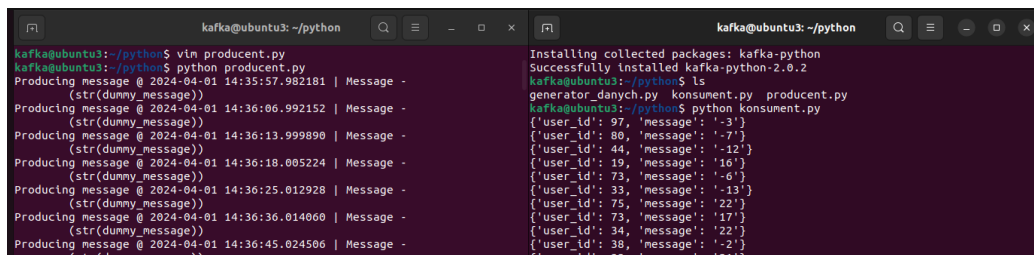
mar 30 13:02:59 ubuntu3 systemd[1]: Started kafka.service.
lines 1-12/12 (END)
```

```
kafka@ubuntu3: ~  
kafka@ubuntu3:~/kafka$ ls  
bin  config  kafka.log  libs  LICENSE  licenses  logs  NOTICE  site-docs  
kafka@ubuntu3:~/kafka$ ls  
bin  config  kafka.log  libs  LICENSE  licenses  logs  NOTICE  site-docs  
kafka@ubuntu3:~/kafka$ cd ..  
kafka@ubuntu3:~$ ls  
Downloads  kafka  logs  
kafka@ubuntu3:~$ ~kafka/kafka/bin/kafka-topics.sh --create --bootstrap-server localhost:9092 --replication-factor 1 --partitions 1 --topic PSDTopic  
Created topic PSDTopic.  
kafka@ubuntu3:~$ echo "Hello, World" | ~kafka/kafka/bin/kafka-console-producer.sh --broker-list localhost:9092 --topic PSDTopic > /dev/null  
kafka@ubuntu3:~$ ~kafka/kafka/bin/kafka-console-consumer.sh --bootstrap-server localhost:9092 --topic PSDTopic --from-beginning  
Hello, World  
24-03-30 13:02:59 CET; 2 days ago
```

## Uruchomiłem kafdrop

```
kafka@ubuntu3: ~/Downloads  
kafka@ubuntu3:~/Downloads$ java --add-opens=java.base/sun.nio.ch=ALL-UNNAMED -jar kafdrop-3.31.0.jar --kafka.brokerConnect=localhost:9092  
2024-03-30 13:05:34.027 INFO 4760 [kground-preinit] o.h.v.i.u.Version  
: HV000001: Hibernate Validator 6.2.5.Final  
2024-03-30 13:05:34.059 INFO 4760 [ main] o.s.b.StartupInfoLogger  
: Starting Kafdrop v3.31.0 using Java 11.0.22 on ubuntu3 with PID 4760 (/home/kafka/Downloads/kafdrop-3.31.0.jar started by kafka in /home/kafka/Downloads)  
2024-03-30 13:05:34.065 INFO 4760 [ main] o.s.b.SpringApplication  
: No active profile set, falling back to 1 default profile: "default"  
2024-03-30 13:05:35.361 INFO 4760 [ main] i.u.s.s.ServletContextImpl  
: Initializing Spring embedded WebApplicationContext  
2024-03-30 13:05:35.362 INFO 4760 [ main] w.s.c.ServletWebServerApplicationContext: Root WebApplicationContext: initialization completed in 1251 ms  
2024-03-30 13:05:35.541 INFO 4760 [ main] k.c.KafkaConfiguration  
: Checking truststore file kafka.truststore.jks  
2024-03-30 13:05:35.541 INFO 4760 [ main] k.c.KafkaConfiguration  
: Checking keystore file kafka.keystore.jks  
2024-03-30 13:05:35.541 INFO 4760 [ main] k.c.KafkaConfiguration  
: Checking properties file kafka.properties  
2024-03-30 13:05:35.616 INFO 4760 [ main] k.c.KafkaConfiguration  
: Checking truststore file kafka.truststore.jks  
2024-03-30 13:05:35.617 INFO 4760 [ main] k.c.KafkaConfiguration  
psd@ubuntu3: ~
```

Przepisałem i uruchomiłem przykładowe skrypty python producenta i konsumenta



The image shows two terminal windows. The left window shows the execution of a Kafka producer script, which prints messages like 'Producing message @ 2024-04-01 14:35:57.982181 | Message - (str(dummy\_message))'. The right window shows the execution of a Kafka consumer script, which prints messages like 'Installing collected packages: kafka-python', 'Successfully installed kafka-python-2.0.2', and then a list of messages received, such as {'user\_id': 97, 'message': '-3'}.

## 2 Przerobienie skryptu Python

Postanowiłem przerobić pythonowy skrypt na taki który imituje sensor temperatury

Chciałem stworzyć skrypt który z jednej strony jest prosty do napisania, a z drugiej imituje faktyczną praktyczną funkcjonalność którą można by użyć na przykład przy urządzeniach internetu rzeczy

Generator danych:

---

```
import random
import time

def generate_temperature_reading():
    location_id = random.randint(1, 10) # Simulate 10 different locations
    temperature = random.uniform(20, 40) # Temperature range from 20 to 40
    timestamp = time.time() # Current Unix timestamp
    return {
        'location_id': location_id,
        'temperature': round(temperature, 2),
        'timestamp': timestamp
    }
```

---

Producent:

---

```
import json
import random
import time
from kafka import KafkaProducer
from simulate_temperature_sensor import generate_temperature_reading

def serializer(message):
```

```

        return json.dumps(message).encode('utf 8')

producer = KafkaProducer(
    bootstrap_servers=['localhost:9092'],
    value_serializer=serializer
)

if __name__ == '__main__':
    while True:
        reading = generate_temperature_reading()
        print(f"Sending reading: {reading}")
        producer.send('temperature_readings', reading)
        time.sleep(random.randint(1, 5)) # Simulate readings sent at

```

---

Konsument:

---

```

import json
from kafka import KafkaConsumer

# Thresholds
TEMP_TOO_COLD = 10
TEMP_TOO_HOT = 35

def process_temperature_reading(reading):
    temperature = reading['temperature']
    if temperature < TEMP_TOO_COLD:
        alert = f"WARNING: Temperature is too cold! ({temperature}řC)"
    elif temperature > TEMP_TOO_HOT:
        alert = f"WARNING: Temperature is too hot! ({temperature}řC)"
    else:
        alert = f"Temperature is normal. ({temperature}řC)"
    return alert

if __name__ == '__main__':
    consumer = KafkaConsumer(
        'temperature_readings',
        bootstrap_servers='localhost:9092',
        auto_offset_reset='earliest'
    )

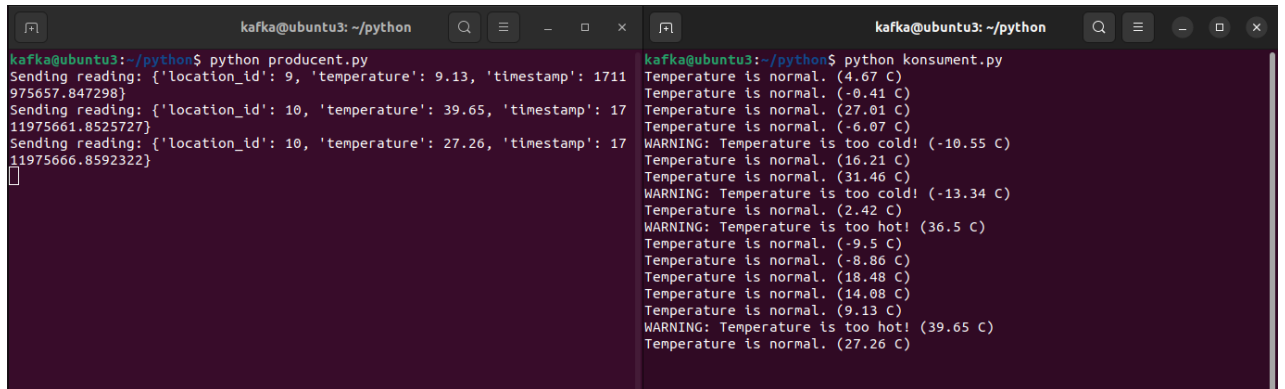
    for message in consumer:
        reading = json.loads(message.value)

```

```
alert_message = process_temperature_reading(reading)
print(alert_message)
```

---

Przykładowe działanie:



The image shows two terminal windows side-by-side. The left window, titled 'kafka@ubuntu3: ~/python', shows the output of 'python producent.py'. It displays three JSON messages being sent to a Kafka topic, each containing 'location\_id', 'temperature', and 'timestamp'. The right window, also titled 'kafka@ubuntu3: ~/python', shows the output of 'python konsument.py'. It displays a series of temperature readings in Celsius, with some warnings for temperatures that are 'too cold' or 'too hot'.

```
kafka@ubuntu3:~/python$ python producent.py
Sending reading: {'location_id': 9, 'temperature': 9.13, 'timestamp': 1711
975657.847298}
Sending reading: {'location_id': 10, 'temperature': 39.65, 'timestamp': 17
11975661.8525727}
Sending reading: {'location_id': 10, 'temperature': 27.26, 'timestamp': 17
11975666.8592322}
█

kafka@ubuntu3:~/python$ python konsument.py
Temperature is normal. (4.67 C)
Temperature is normal. (-0.41 C)
Temperature is normal. (27.01 C)
Temperature is normal. (-6.07 C)
WARNING: Temperature is too cold! (-10.55 C)
Temperature is normal. (16.21 C)
Temperature is normal. (31.46 C)
WARNING: Temperature is too cold! (-13.34 C)
Temperature is normal. (2.42 C)
WARNING: Temperature is too hot! (36.5 C)
Temperature is normal. (-9.5 C)
Temperature is normal. (-8.86 C)
Temperature is normal. (18.48 C)
Temperature is normal. (14.08 C)
Temperature is normal. (9.13 C)
WARNING: Temperature is too hot! (39.65 C)
Temperature is normal. (27.26 C)
```