WEEK 3-4

September 28, 2024

Student Information

Full Name : Deniz Karakoyun
Id Number : 2580678

While gathering this data and conducting experiments, I am using a 13.3-inch MacBook Pro with an Apple M2 chip.

Genel Özellikler

- **İşlemci:** Apple M2 çipi
- **Bellek:** 24 GB RAM
- **Depolama:** 1 TB SSD
- **Grafik Kartı:** Apple M2'nin entegre GPU'su

Introduction

This document explains a Python script designed to produce large messages to a Kafka server. The purpose of the script is to analyze the message delivery latency as a function of message size. The code is written using the `confluent_kafka` Python package and focuses on the interaction between message size and delivery time.

Code Explanation

The Python script is structured into several key components:

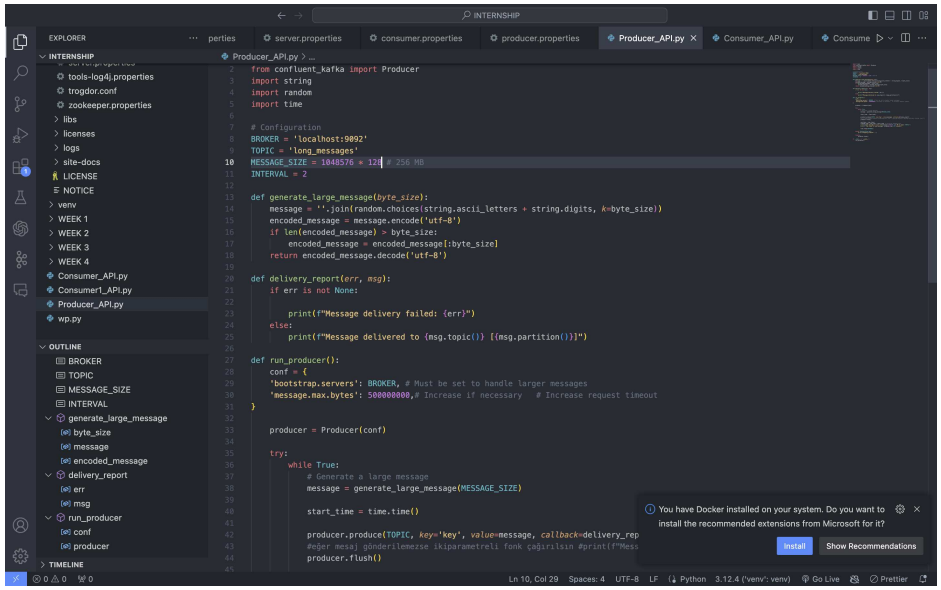

Figure 1: Producer API Python Code to Create Large Messages

Configuration

- `BROKER`: The Kafka broker address (`localhost:9092`).
- `TOPIC`: The Kafka topic where messages are sent (`long_messages`).
- `MESSAGE_SIZE`: Defines the size of each message in bytes. In this script, it is set to 1 GB.
- `INTERVAL`: The time interval between sending messages, set to 2 seconds.

Message Generation

The function `generate_large_message` creates a random string of characters of a specified byte size.

```python
def generate_large_message(byte_size):
    size = byte_size
    message = ''.join(random.choices(string.ascii_letters + string.digits, k=size))
    encoded_message = message.encode('utf-8')
    if len(encoded_message) > byte_size:
        encoded_message = encoded_message[:byte_size]
    return encoded_message.decode('utf-8')
```

Listing 1: Message Generation Function

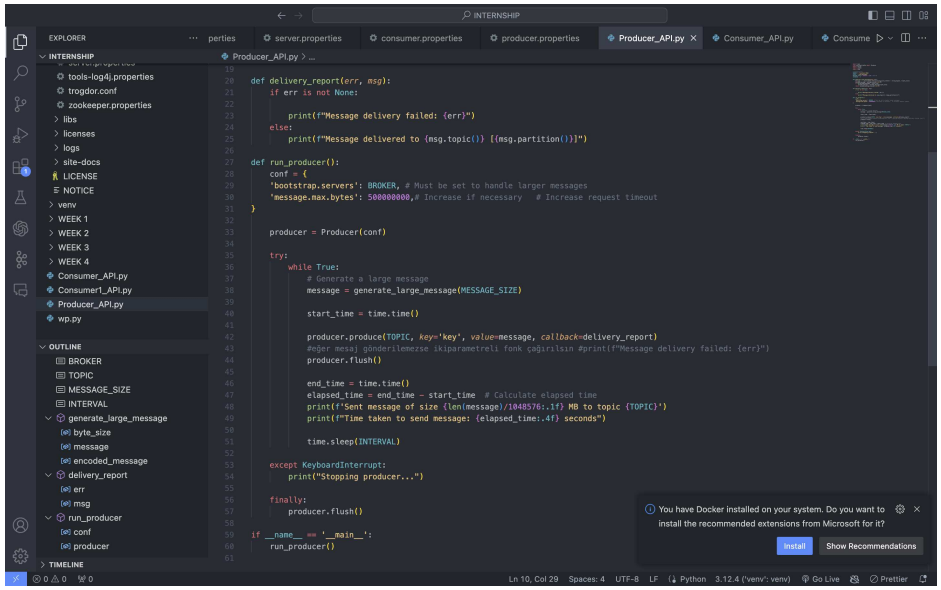

Figure 2: Producer API Python Code to Create Large Messages Continued

Delivery Report

The `delivery_report` function provides feedback on the delivery status of each message. It confirms successful deliveries and reports any errors.

```python
def delivery_report(err, msg):
    if err is not None:
        print(f"Message delivery failed: {err}")
    else:
        print(f"Message delivered to {msg.topic()} [{msg.partition()}]")
```

Listing 2: Delivery Report Callback

Producer Execution

The main function `run_producer` sets up the Kafka producer, generates large messages, and sends them to the specified topic. It also measures the time taken for each delivery.

```python
def run_producer():
    producer = Producer({'bootstrap.servers': BROKER})
    try:
        while True:
            conf = {
                'bootstrap.servers': 'localhost:9092',
                'message.max.bytes': 50000000
            }
            message = generate_large_message(MESSAGE_SIZE)
            producer = Producer(conf)
```

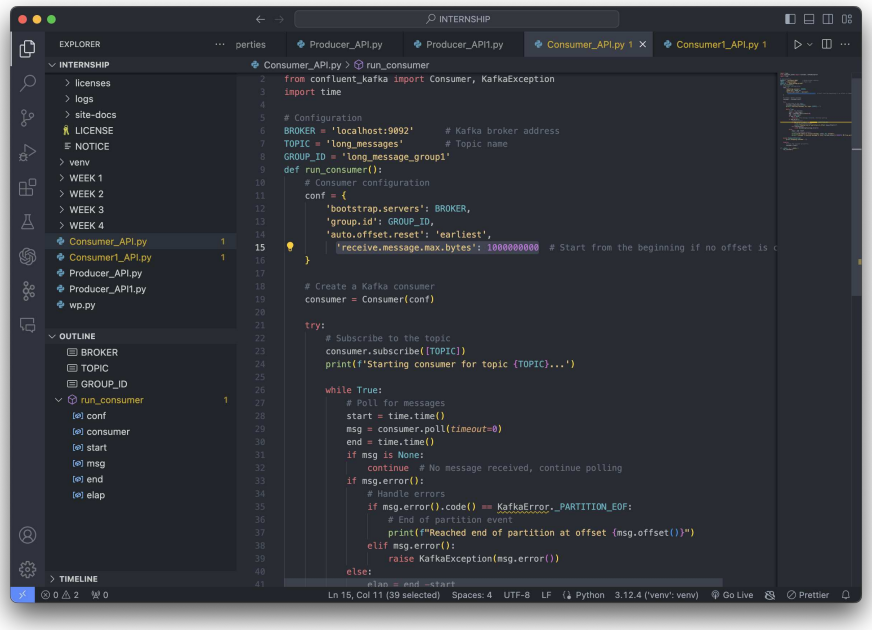

Figure 3: Consumer API Python Code to Create Large Messages

```
            start_time = time.time()
            print(f'Sent message of size {len(message)/1048576} MB to topic {TOPIC} and the random message is {message}')
            time.sleep(INTERVAL)
            end_time = time.time()
            elapsed_time = end_time - start_time
            print(f"Time taken to send message: {elapsed_time:.4f} seconds")
    except KeyboardInterrupt:
        print("Stopping producer...")
    finally:
        producer.flush()
```

Listing 3: Producer Execution

Conclusion

The script effectively generates large messages and measures the time taken to deliver them, demonstrating how message size affects delivery latency in Kafka. This analysis can be useful for optimizing Kafka performance in real-world applications where message size is a critical factor.

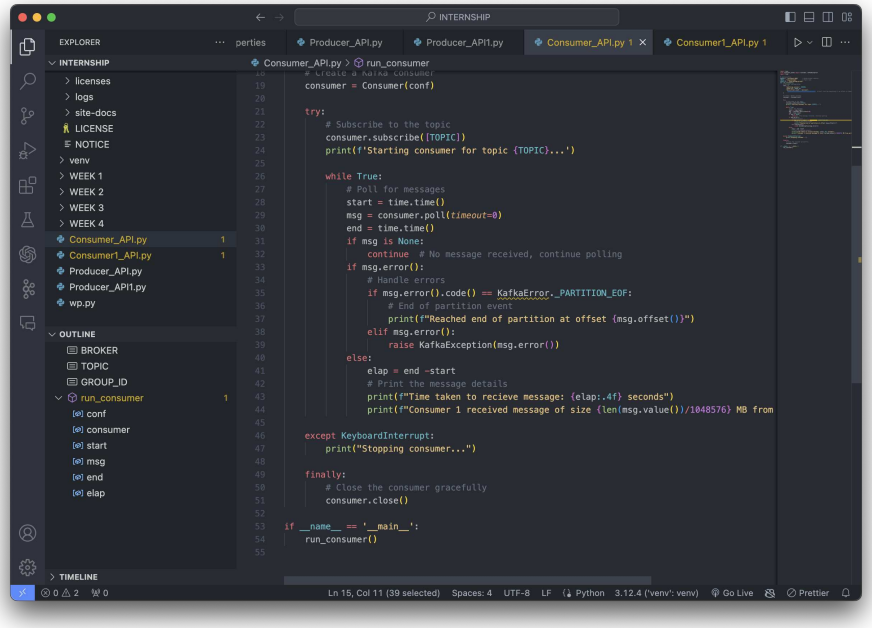

Figure 4: Consumer API Python Code to Create Large Messages Continued

Message Delivery Latency in Kafka Server

Message delivery latency in a Kafka server is an important metric that reflects how long it takes for a message to travel from the producer to the consumer. This latency can be influenced by several factors, including the size of the message. Larger messages typically take longer to be delivered due to increased processing and transmission times.

The following table presents latency data collected for various message sizes:

Message Size	Latency (ms)
1 MB	0.0020
2 MB	0.0037
4 MB	0.0075
8 MB	0.0142
16 MB	0.0272
32 MB	0.0530
64 MB	0.1018
128 MB	0.2110

Table 1: Message Size vs. Delivery Latency

Latency Visualization

The graph below visualizes the relationship between message size and delivery latency. It clearly shows that as the message size increases, the latency grows significantly, highlighting the importance of optimizing message sizes for performance-critical applications.

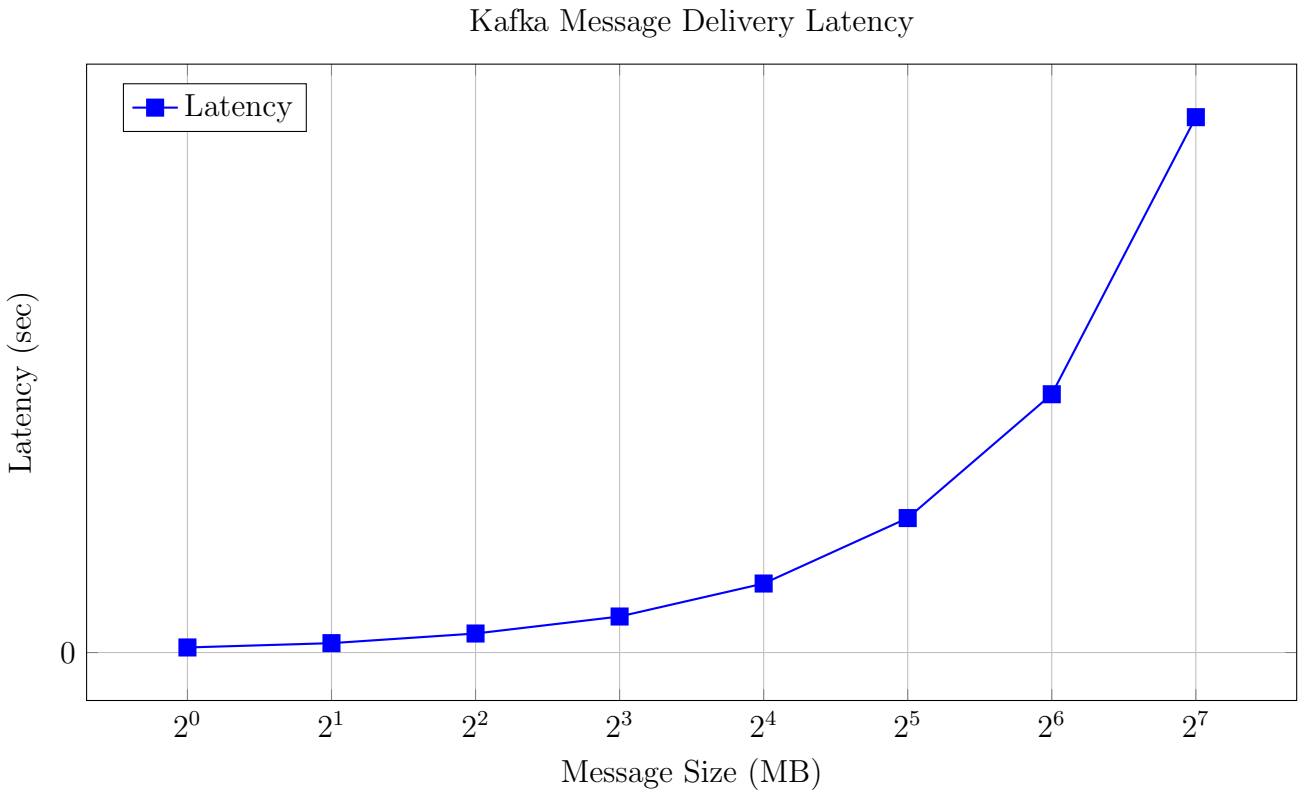

Figure 5: Message Size vs. Latency in Kafka Server

Sample Datas of Large Messaging in Kafka

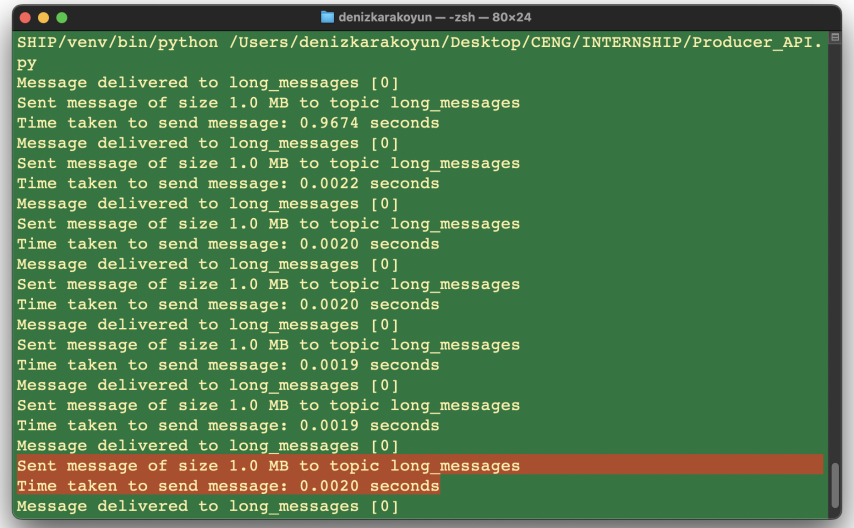

Figure 6: Kafka Messaging Size of 1 MB

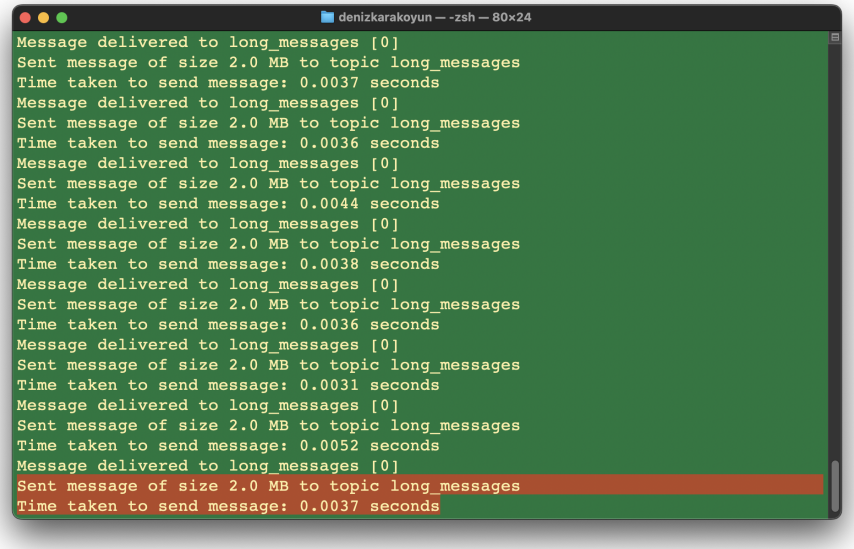

Figure 7: Kafka Messaging Size of 2 MB

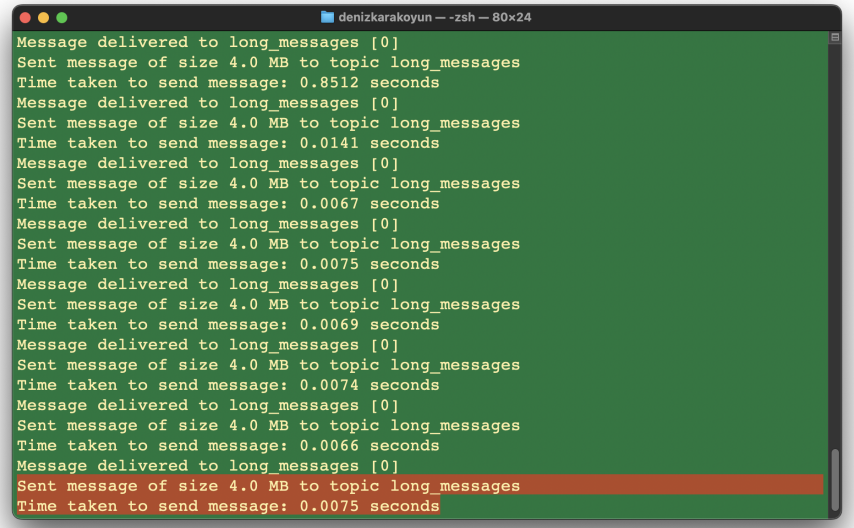

Figure 8: Kafka Messaging Size of 4 MB

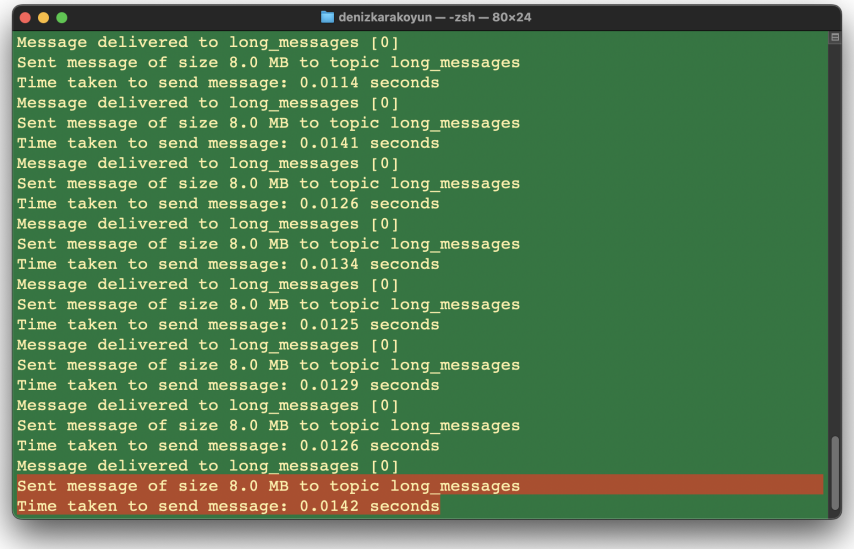

Figure 9: Kafka Messaging Size of 8 MB

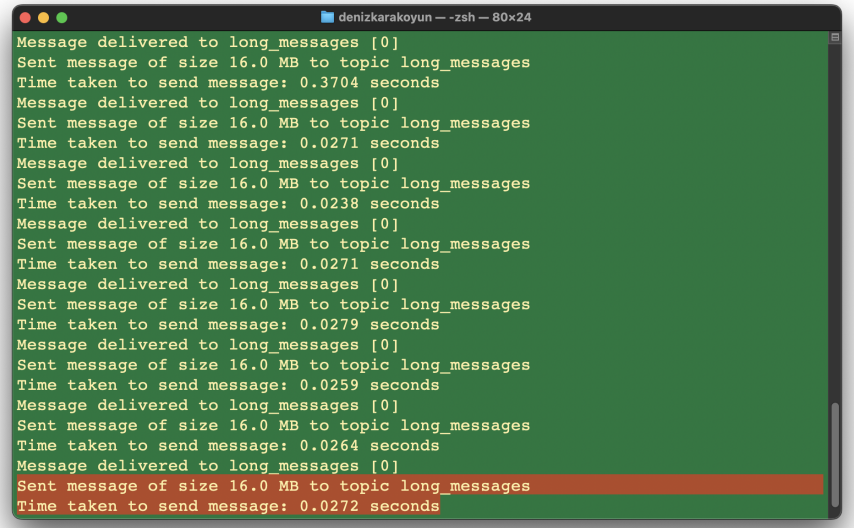

Figure 10: Kafka Messaging Size of 16 MB

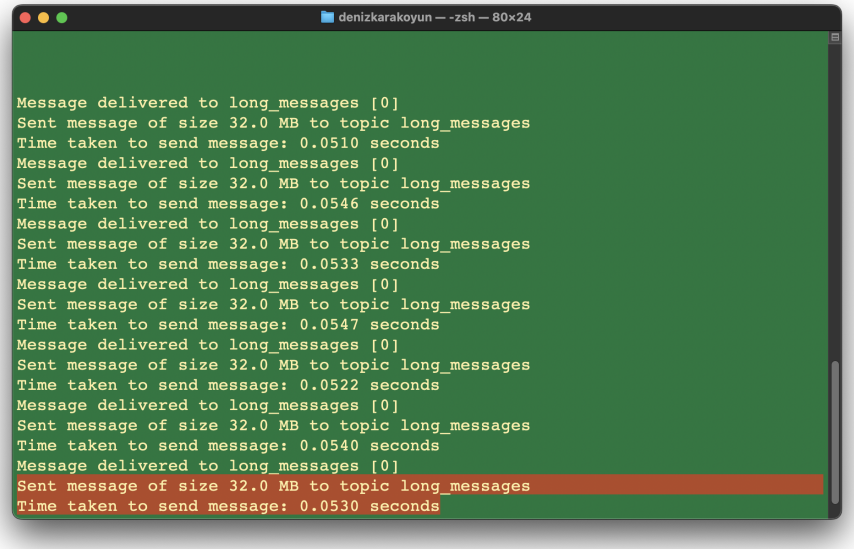

Figure 11: Kafka Messaging Size of 32 MB

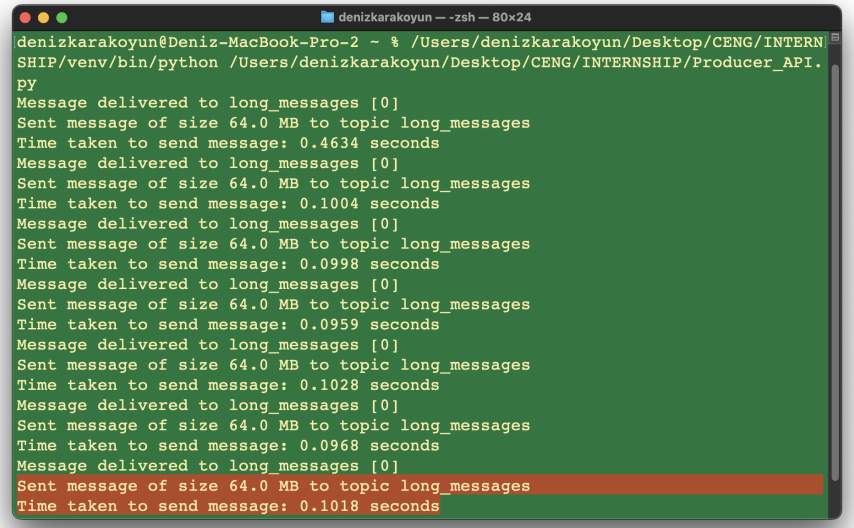

Figure 12: Kafka Messaging Size of 64 MB

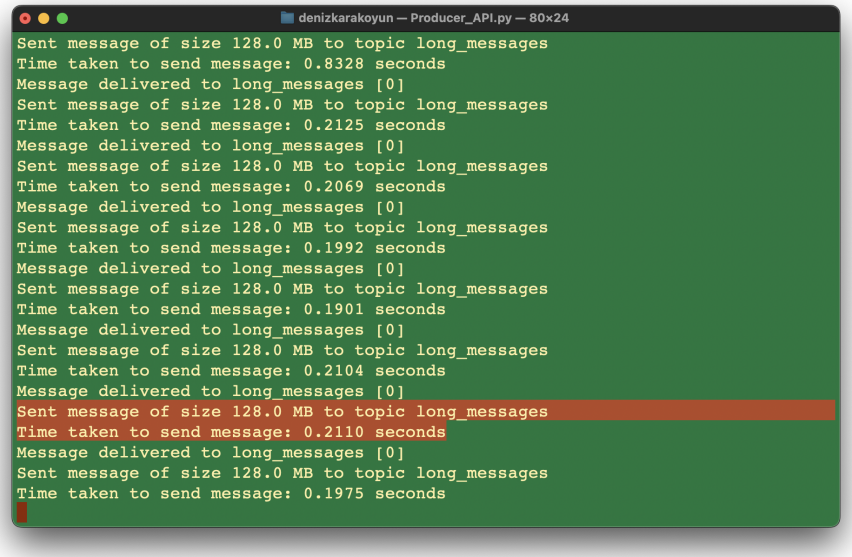

Figure 13: Kafka Messaging Size of 128 MB

Single Producer Single Consumer Message Recieve Time

Message Size	Latency (ms)
1 MB	0.0001
2 MB	0.0002
4 MB	0.0004
8 MB	0.0007
16 MB	0.0012
32 MB	0.0027
64 MB	0.0047

Table 2: Message Size vs. Delivery Latency

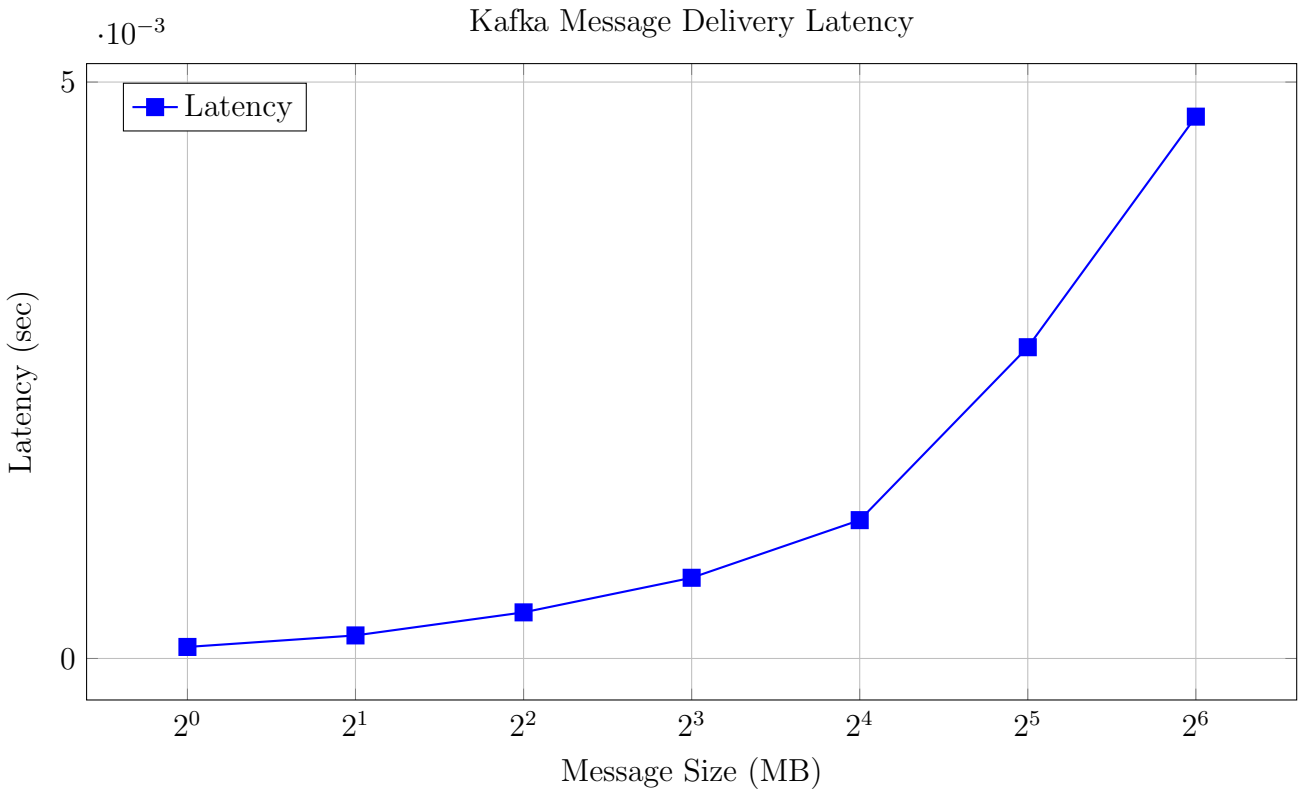

Figure 14: Message Size vs. Latency in Kafka Server

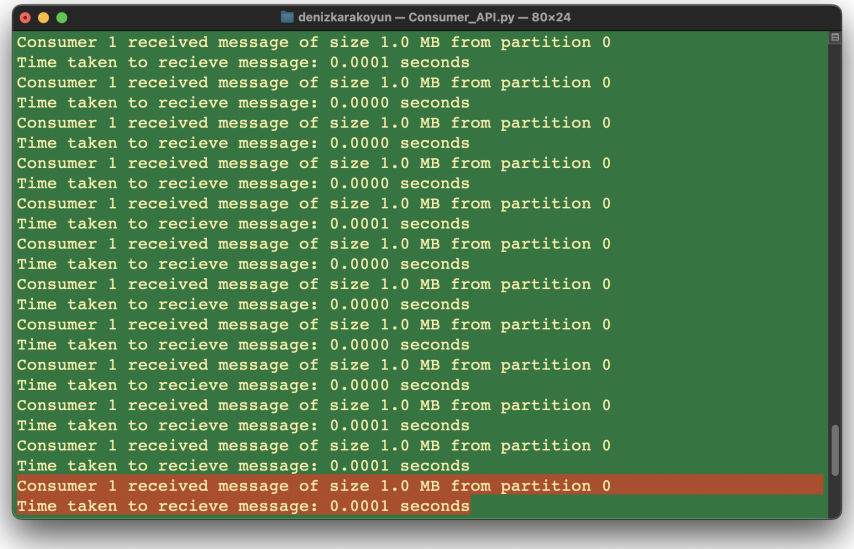

Figure 15: Kafka Messaging Size of 1 MB

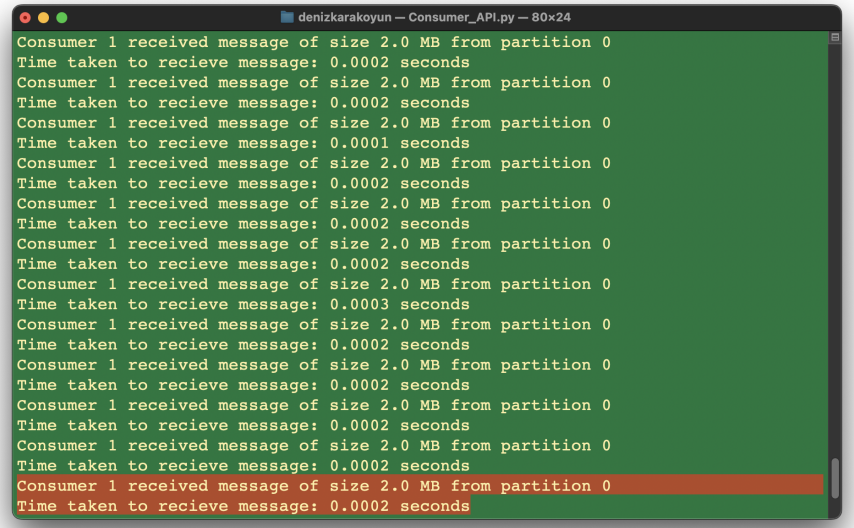

Figure 16: Kafka Messaging Size of 2 MB

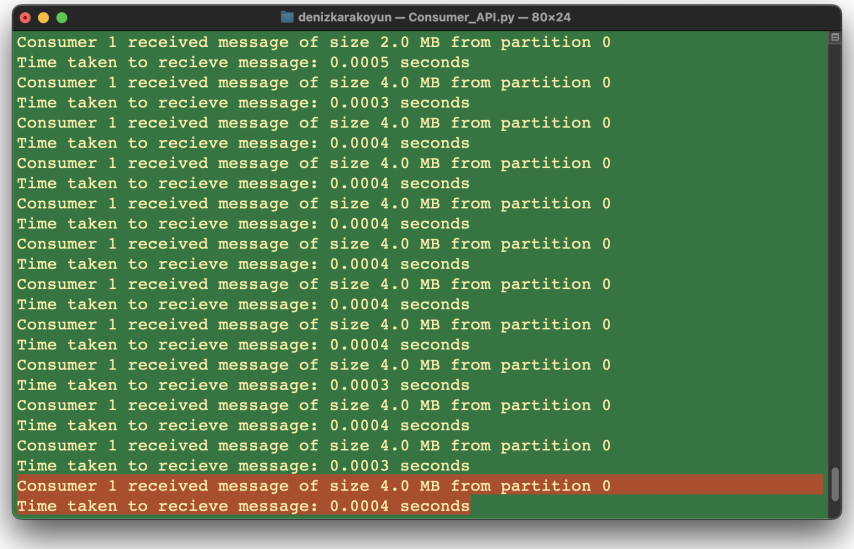

Figure 17: Kafka Messaging Size of 4 MB

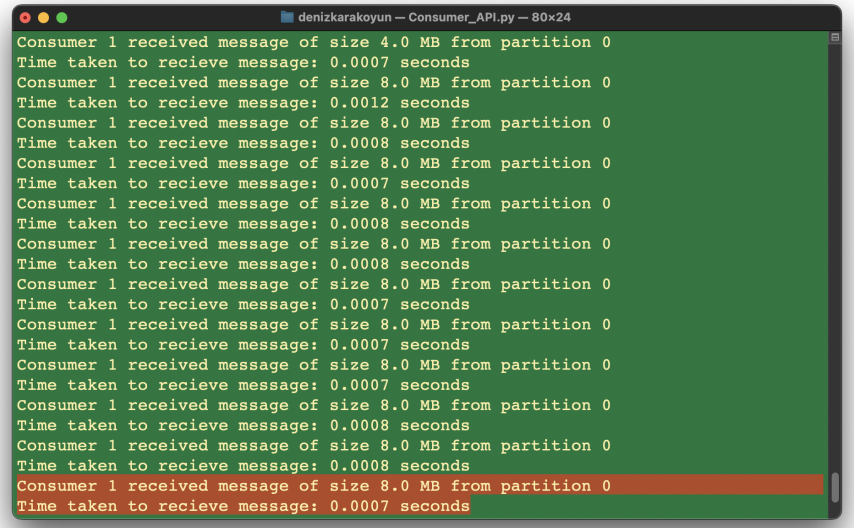

Figure 18: Kafka Messaging Size of 8 MB

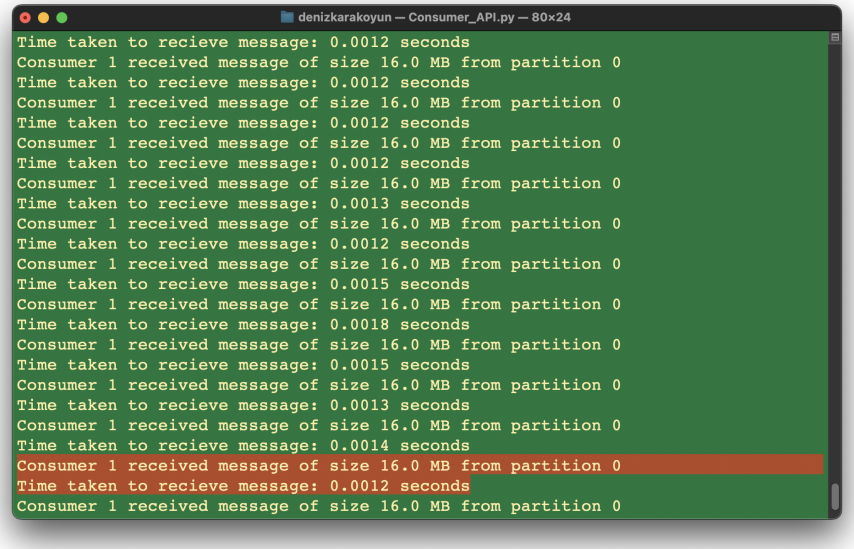

Figure 19: Kafka Messaging Size of 16 MB

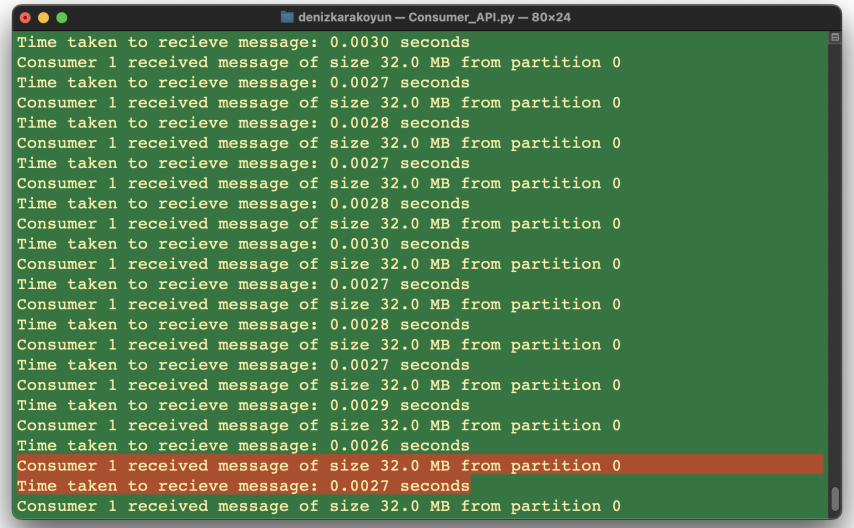

Figure 20: Kafka Messaging Size of 32 MB

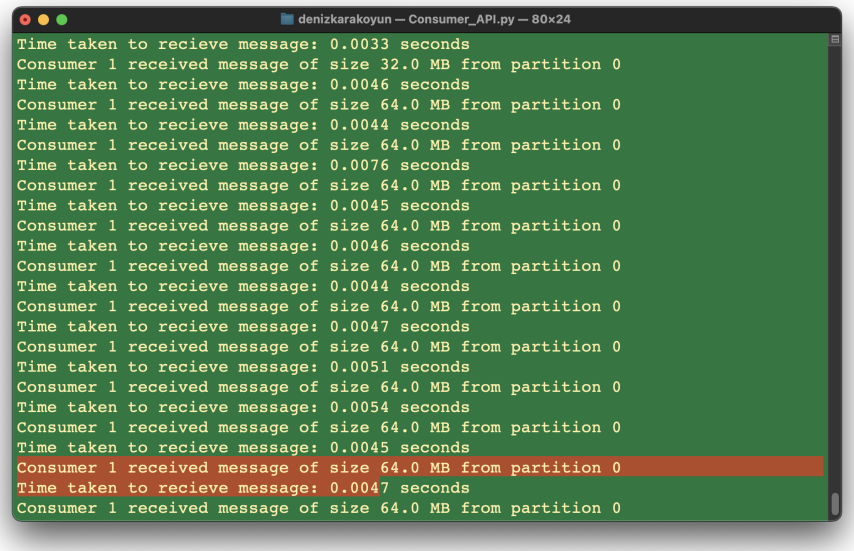

Figure 21: Kafka Messaging Size of 64 MB

Single Producer Multiple Consumer Message Recieve Time

Message Size	Recieve Time 1 (ms)	Recieve Time 2 (ms)
1 MB	0.0001	0.0001
2 MB	0.0003	0.0003
4 MB	0.0004	0.0004
8 MB	0.0009	0.0003
16 MB	0.0016	0.0016
32 MB	0.0035	0.0030
64 MB	0.0069	0.0066

Table 3: Message Size vs. Delivery Latency

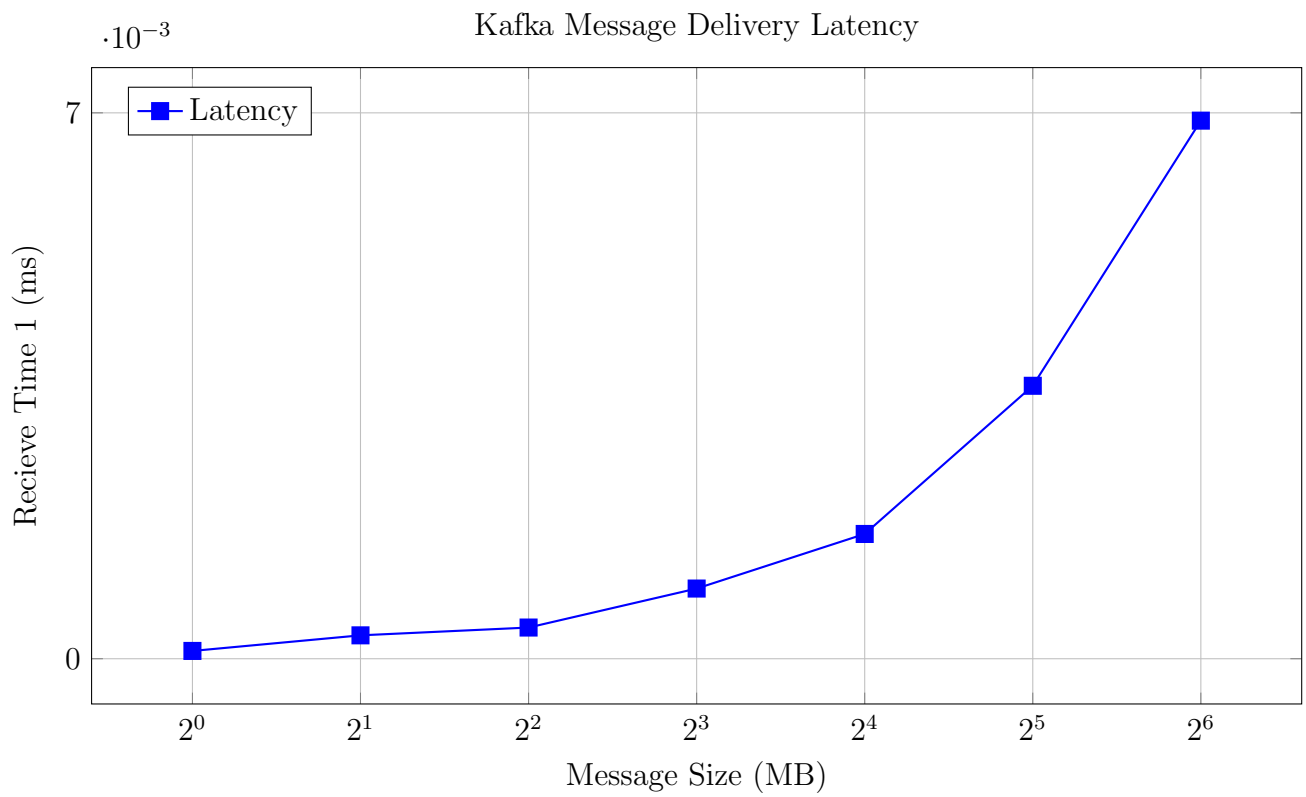

Figure 22: Message Size vs. Receive Time in Consumer 1

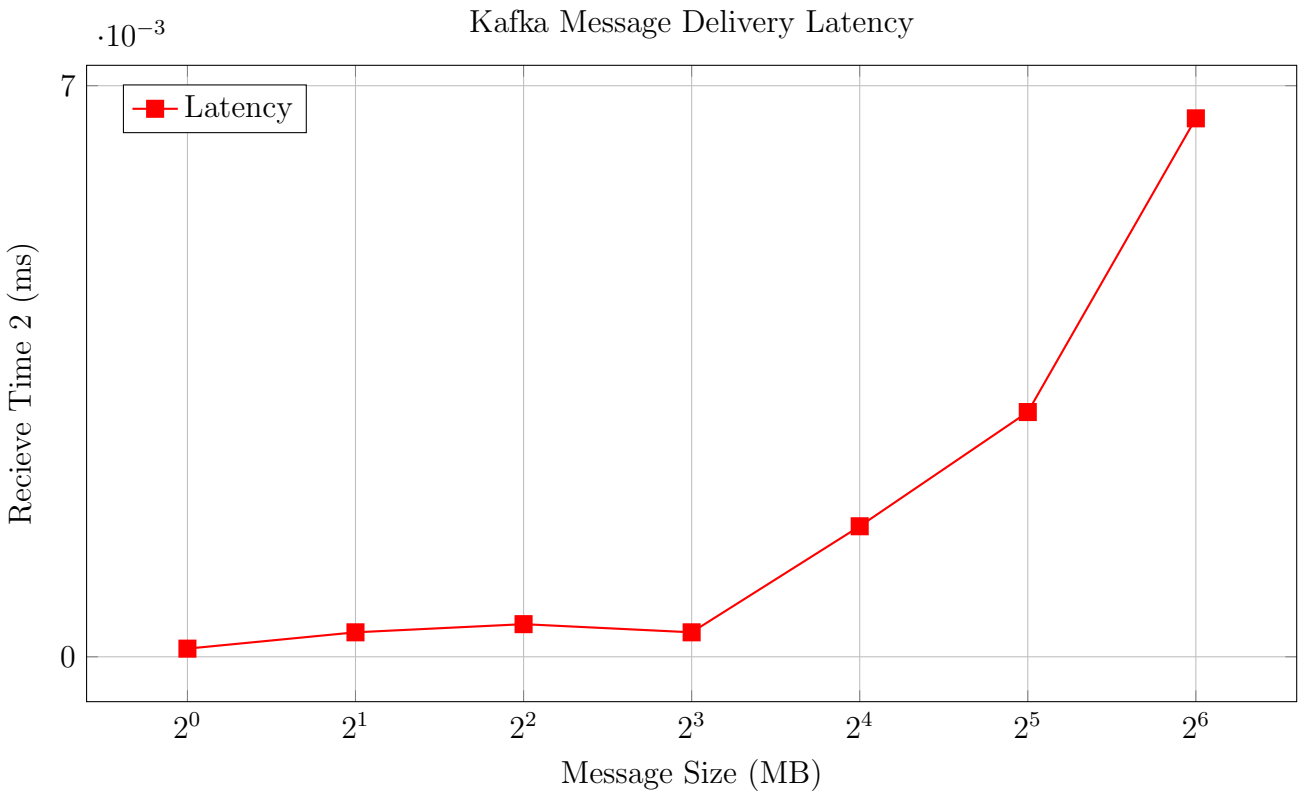

Figure 23: Message Size vs. Receive Time in Consumer 2

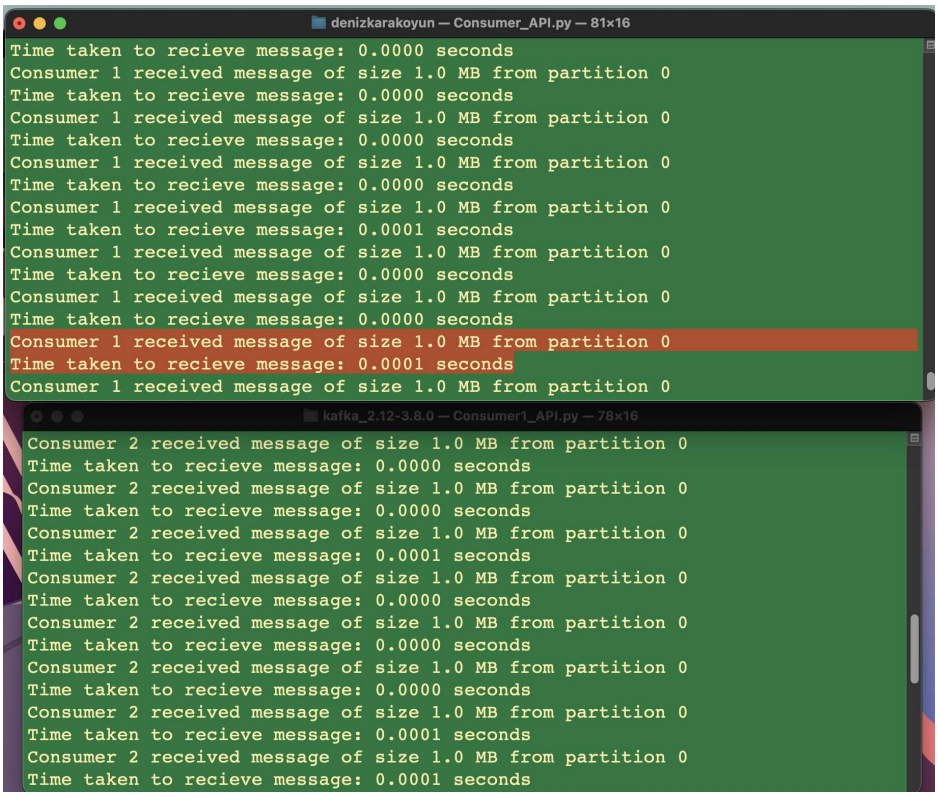

Figure 24: Kafka Messaging Size of 1 MB

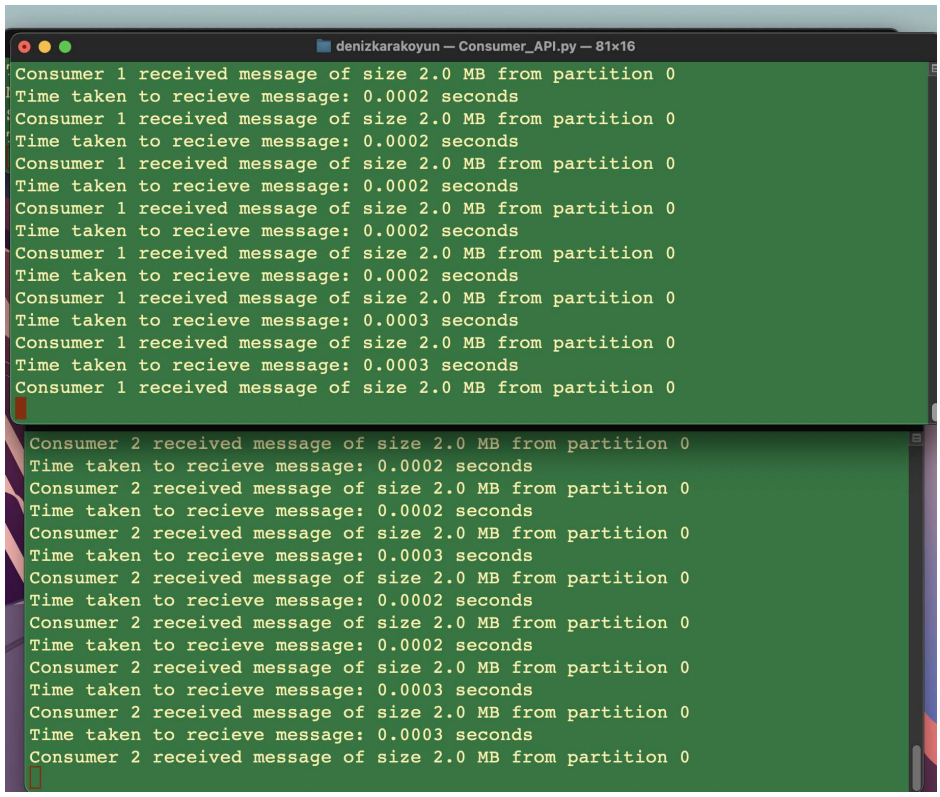

Figure 25: Kafka Messaging Size of 2 MB

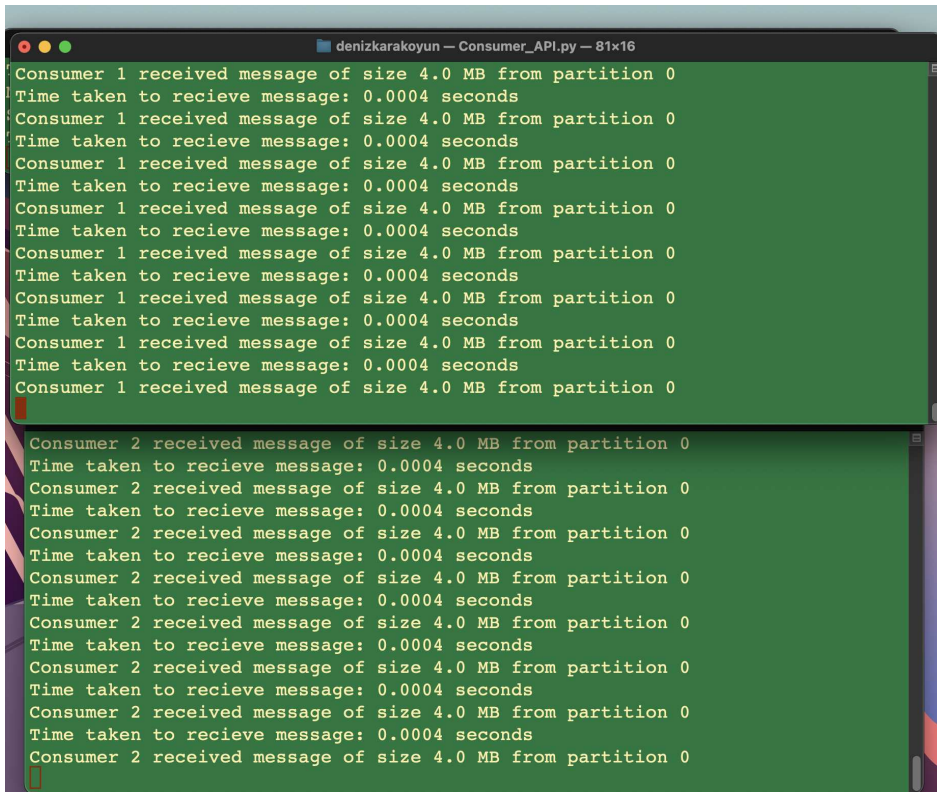

Figure 26: Kafka Messaging Size of 4 MB

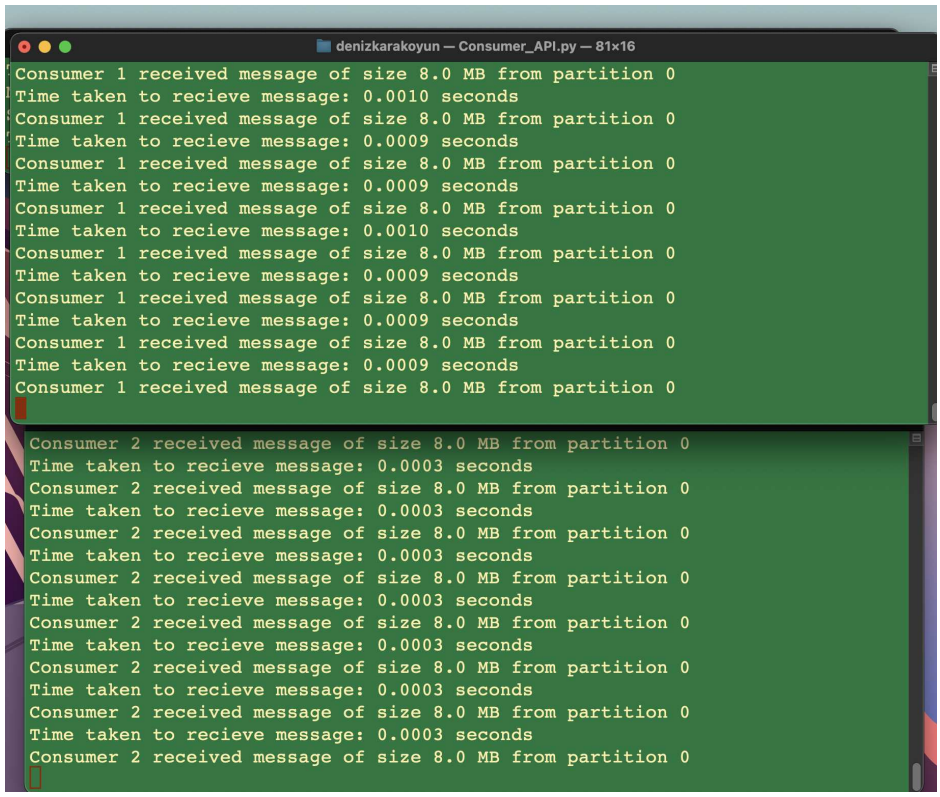

Figure 27: Kafka Messaging Size of 8 MB

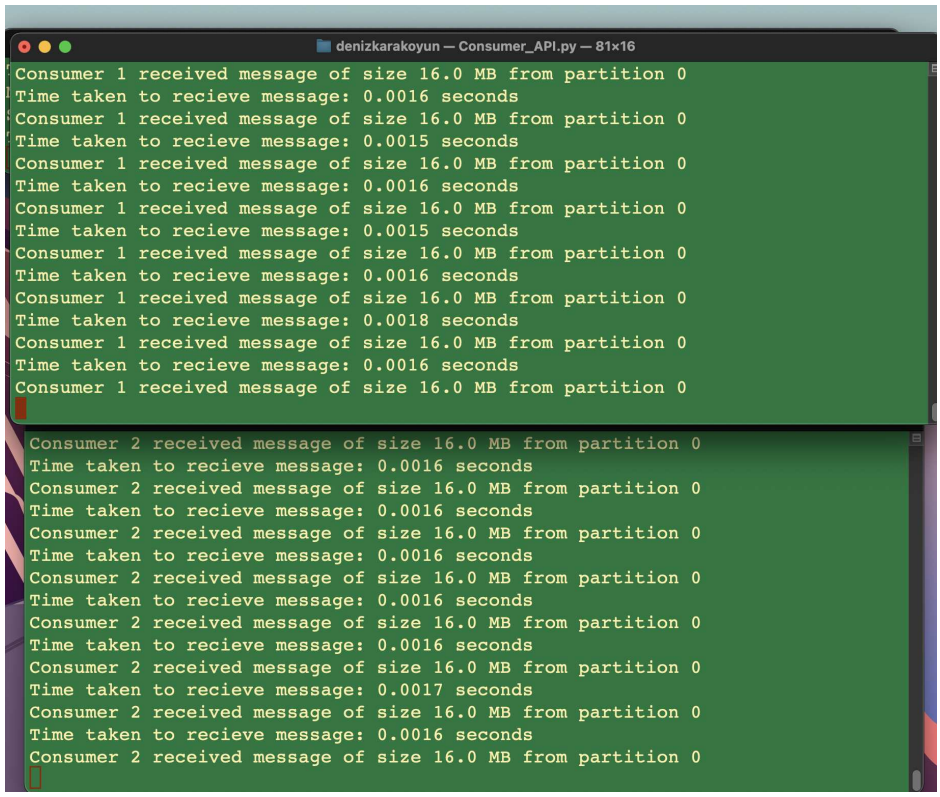

Figure 28: Kafka Messaging Size of 16 MB

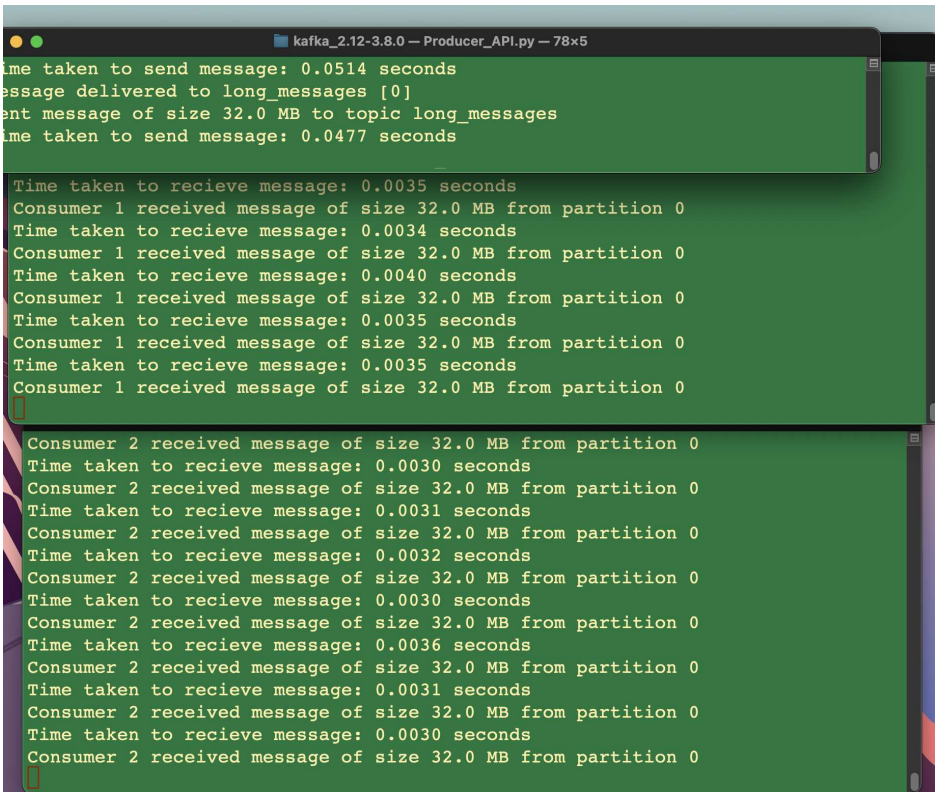

Figure 29: Kafka Messaging Size of 32 MB

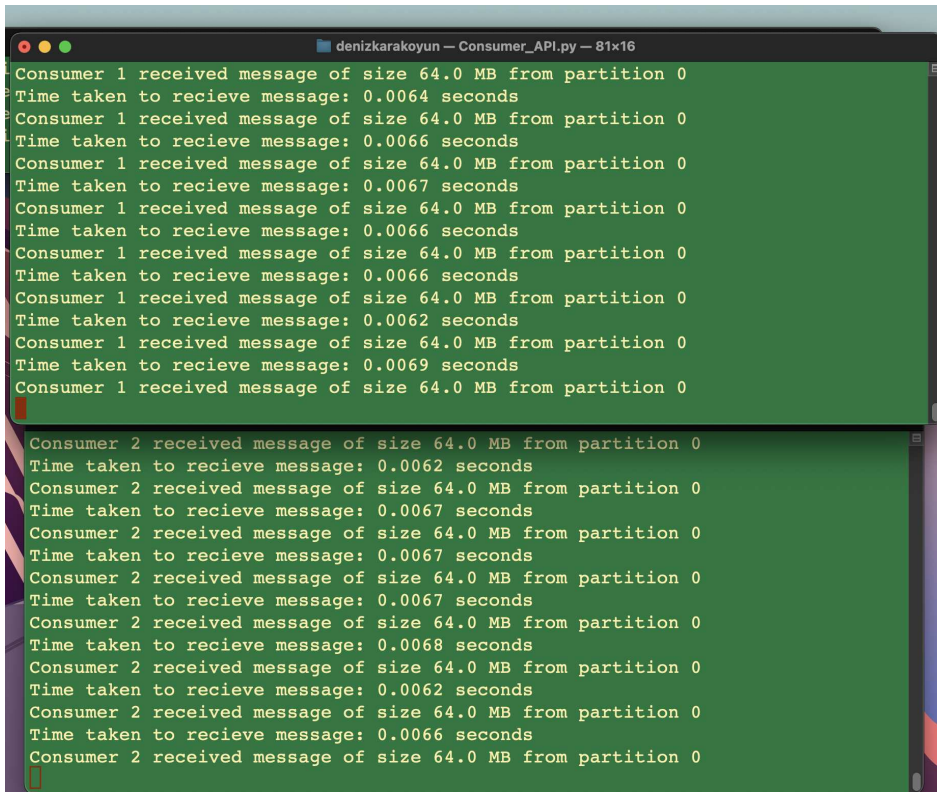

Figure 30: Kafka Messaging Size of 64 MB

Multiple Producer Single Consumer Kafka Messaging Timeline

Message Size	Send Time 1 (ms)	Send Time 2 (ms)	Receive Time (ms)
1 MB	0.0018	0.0017	0.0000
2 MB	0.0032	0.0034	0.0002
4 MB	0.0067	0.0062	0.0004
8 MB	0.0127	0.0126	0.0003
16 MB	0.0279	0.0277	0.0014
32 MB	0.0498	0.0574	0.0026
64 MB	0.1021	0.0974	0.0045
128 MB	0.2177	0.2306	0.0135

Table 4: Message Size vs. Delivery Latency

Analysis of Kafka Messaging Latency

The table above (Table 4) illustrates the relationship between message size and delivery latency for multiple producers and a single consumer setup in Kafka. As the message size increases, the send times and receive times exhibit expected growth patterns due to the increased data load.

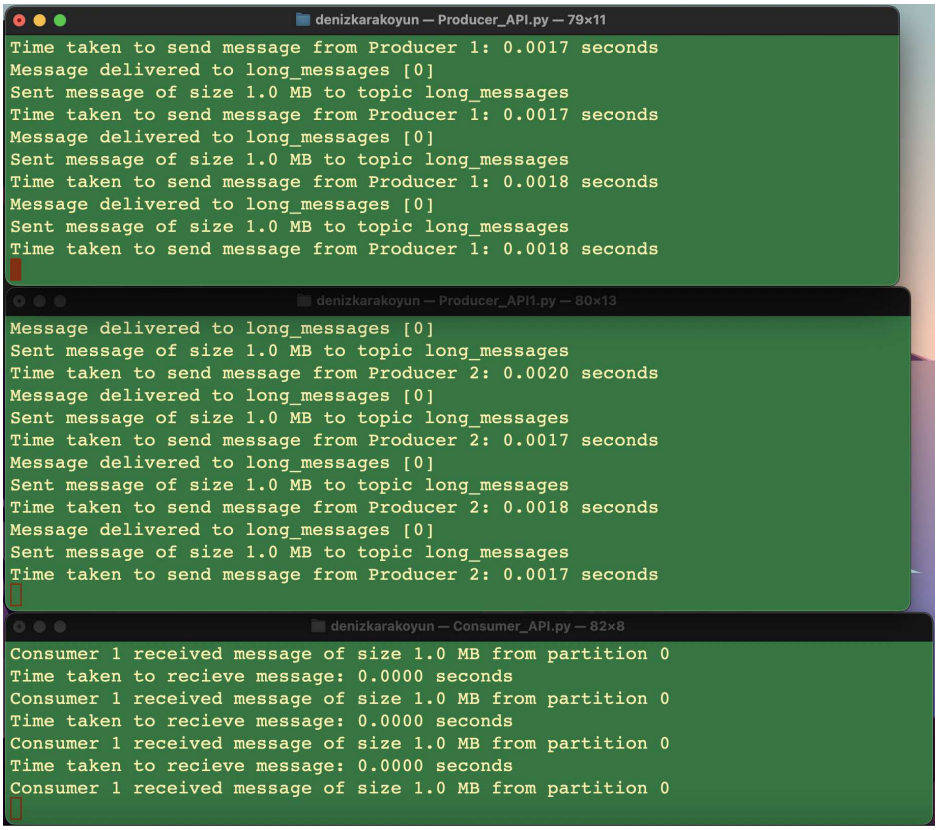

Figure 31: Shows the Kafka messaging timeline for a message size of 1 MB. The send and receive times are minimal, demonstrating efficient handling of smaller messages.

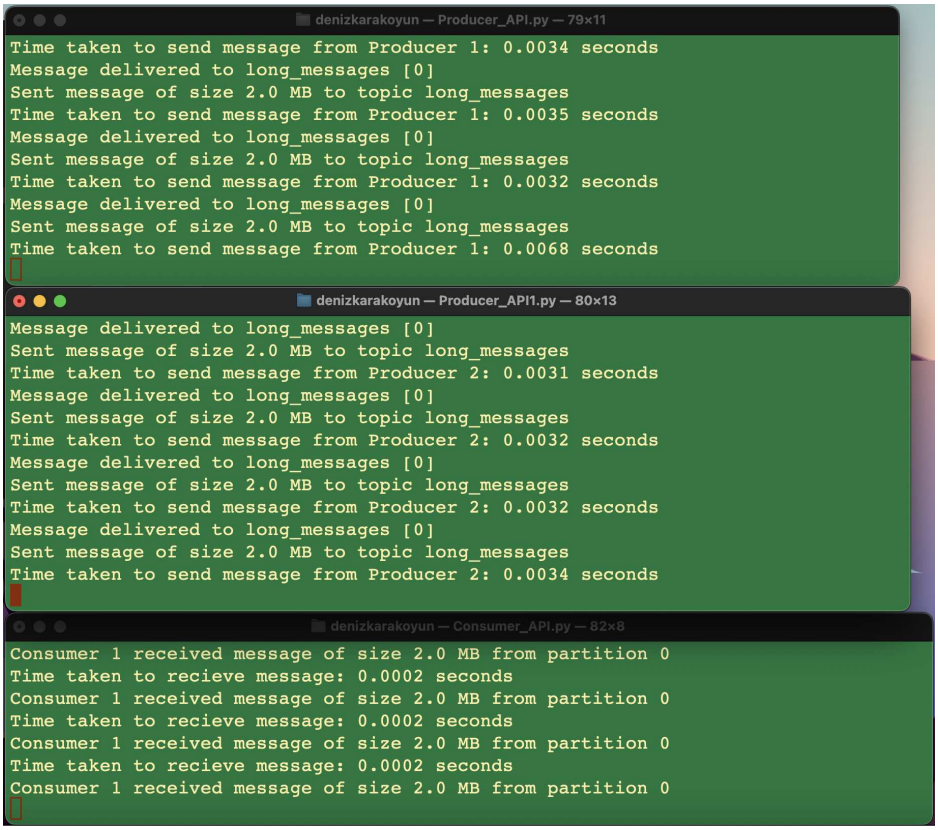

Figure 32: Shows a slight increase in both send and receive times when the message size is doubled to 2 MB, indicating a consistent increase in latency proportional to the message size.

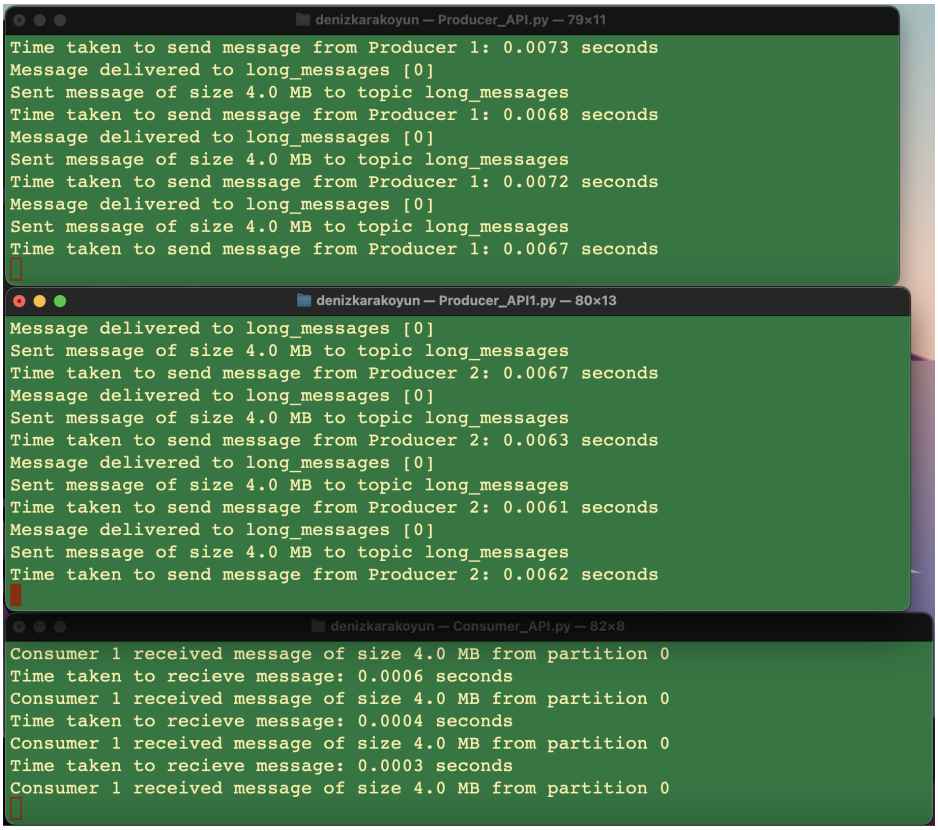

Figure 33: Depicts the timeline for a 4 MB message, showing further increased latency in sending and receiving messages compared to smaller sizes, following the trend observed in previous figures.

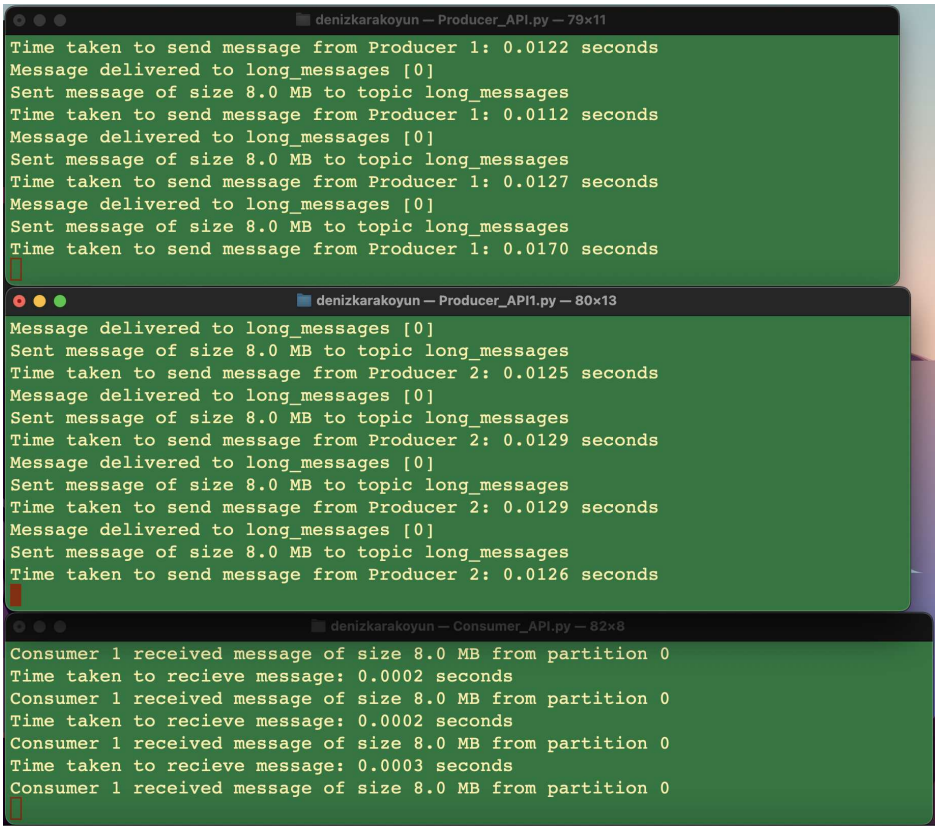

Figure 34: Demonstrates a steady increase in latency as the message size reaches 8 MB, maintaining the linear pattern of growth seen in smaller sizes.

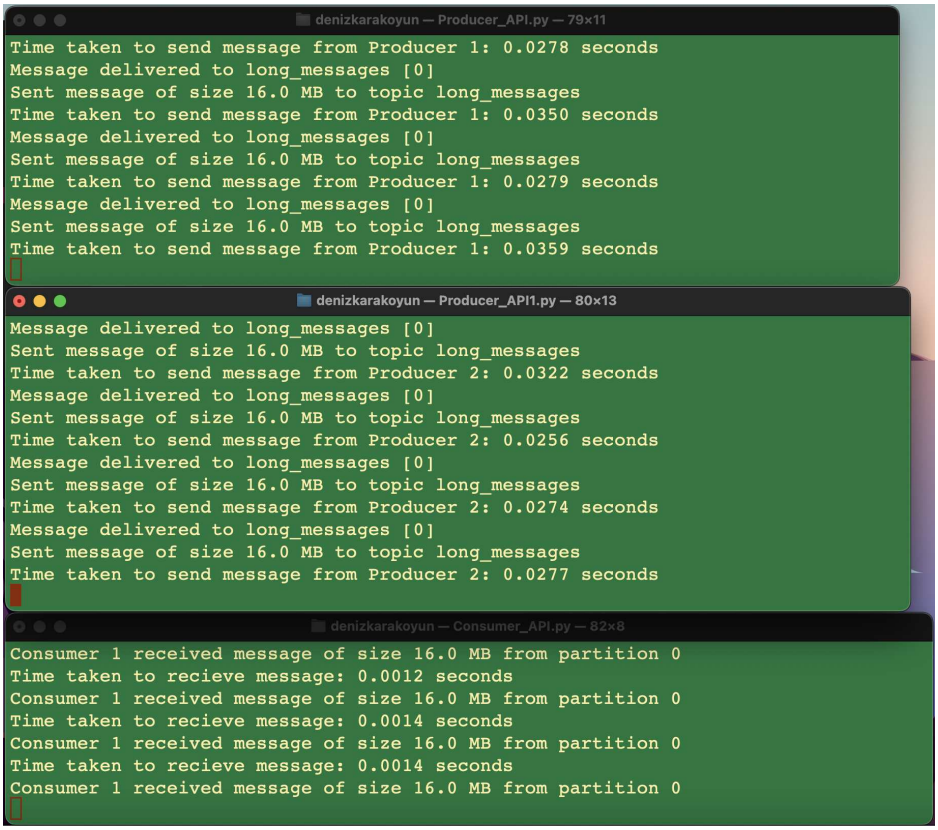

Figure 35: Shows the messaging timeline for a 16 MB message size, reflecting a noticeable increase in both send and receive times as data volume increases, aligning with the overall trend.

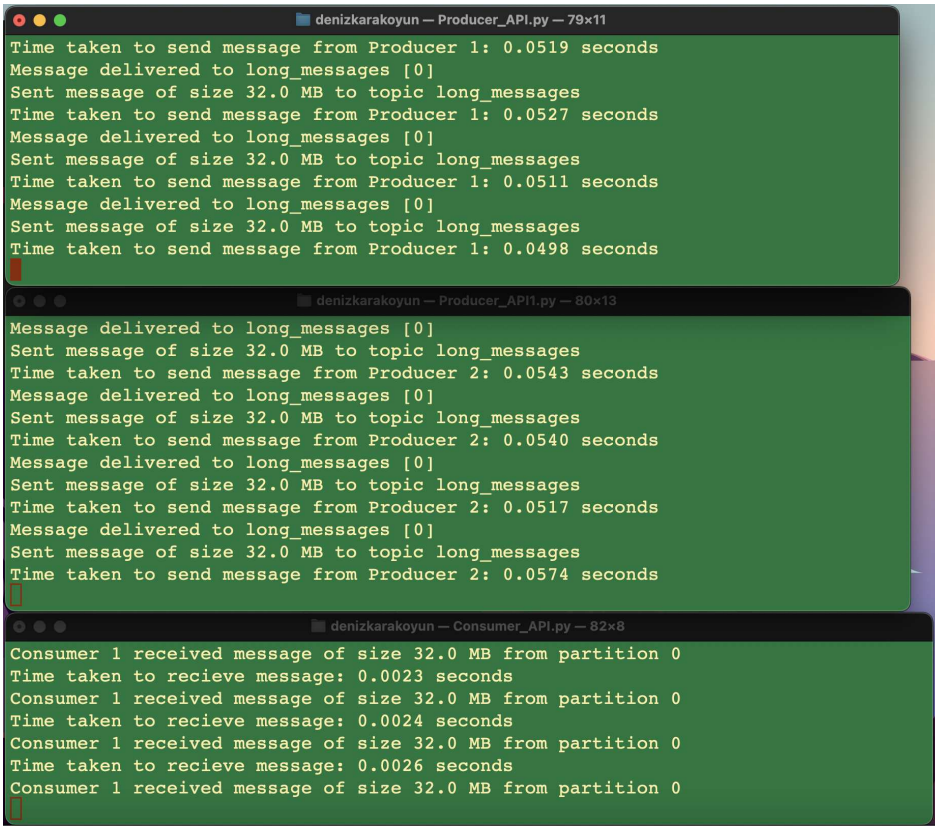

Figure 36: Captures the performance for 32 MB messages, where latency continues to rise, reinforcing the direct correlation between message size and delivery time.

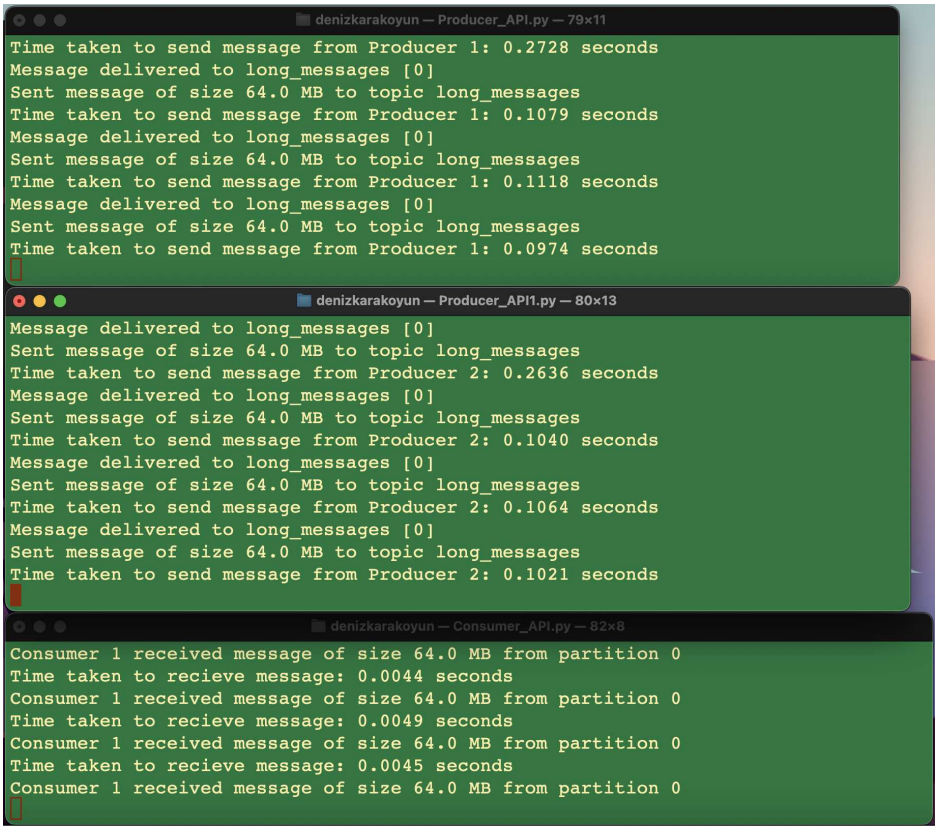

Figure 37: Presents the 64 MB message scenario, illustrating a significant jump in both sending and receiving times. Although there are variations, the increase is still consistent with previous patterns, as seen in the table.

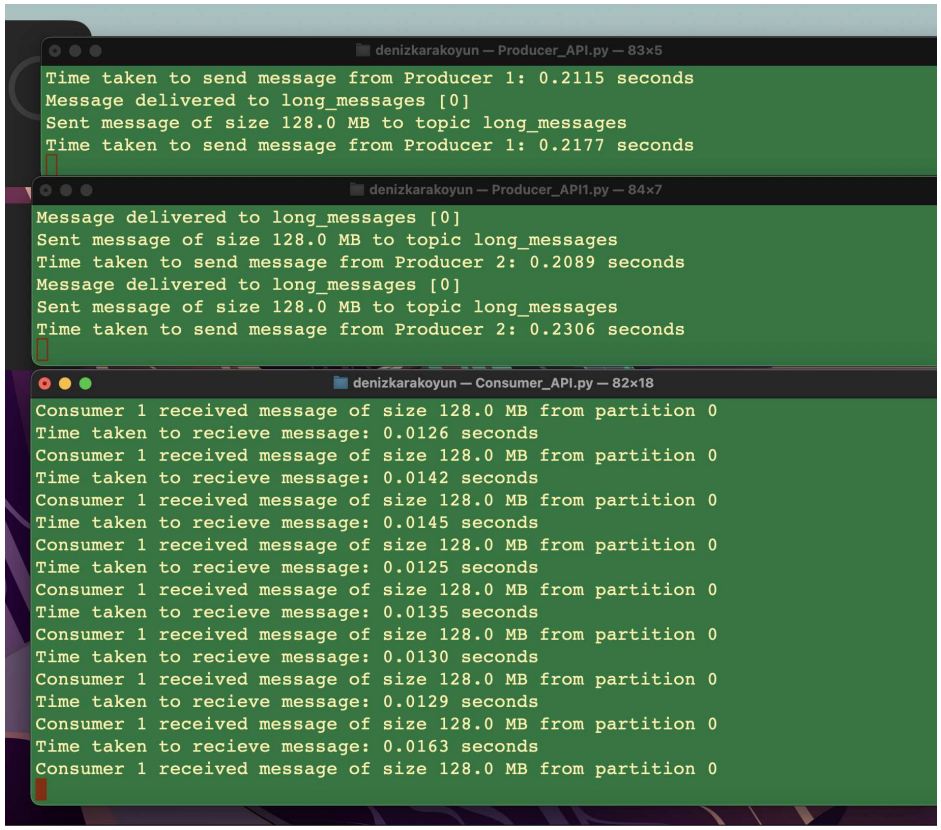

Figure 38: Shows the messaging timeline for the largest message size of 128 MB. The sending time is significantly higher, closely matching the previous trends seen in earlier graphs. With the aid of the table, it's clear that the latency grows progressively with the message size, allowing us to effectively analyze and compare the performance across different scenarios.

Multiple Producer Multiple Consumer Kafka Messaging Timeline

In this part, we will show that the number of consumers and producers does not matter in terms of delivery and recieve time.

Message Size	Send Time 1 (ms)	Send Time 2 (ms)	Receive Time 1 (ms)	Receive Tim
1 MB	0.0020	0.0020	0.0001	0.000
2 MB	0.0038	0.0035	0.0002	0.000
4 MB	0.0065	0.0071	0.0004	0.000
8 MB	0.0141	0.0131	0.0003	0.000
16 MB	0.0275	0.0247	0.0016	0.001
32 MB	0.0494	0.0529	0.0031	0.003
64 MB	0.1018	0.1109	0.0063	0.006
128 MB	0.2159	0.2326	0.0180	0.017

Table 5: Message Size vs. Delivery Latency

Analysis of Kafka Messaging Latency

The table above (Table 5) illustrates the relationship between message size and delivery latency for multiple producers and multiple consumer setup in Kafka. As the message size increases, the send times and receive times exhibit expected growth patterns due to the increased data load and it seems that there is no significant difference between single producers and multiple consumer or multiple producers and single consumer in terms of timeline.

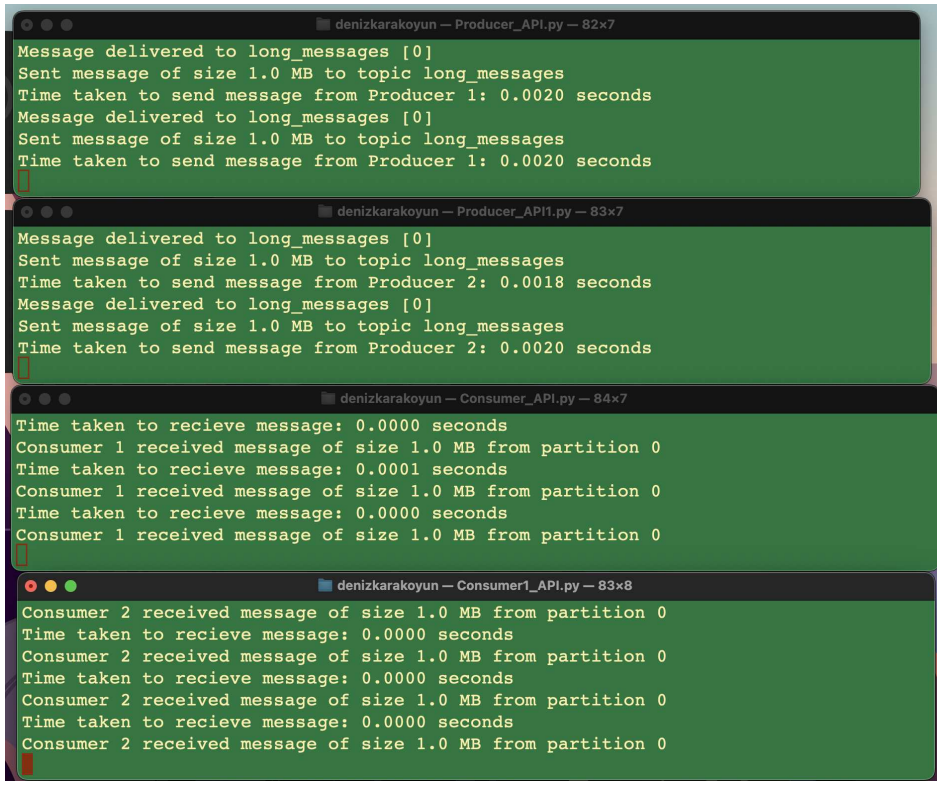

Figure 39: Shows the Kafka messaging timeline for a message size of 1 MB. The send and receive times are minimal, demonstrating efficient handling of smaller messages.

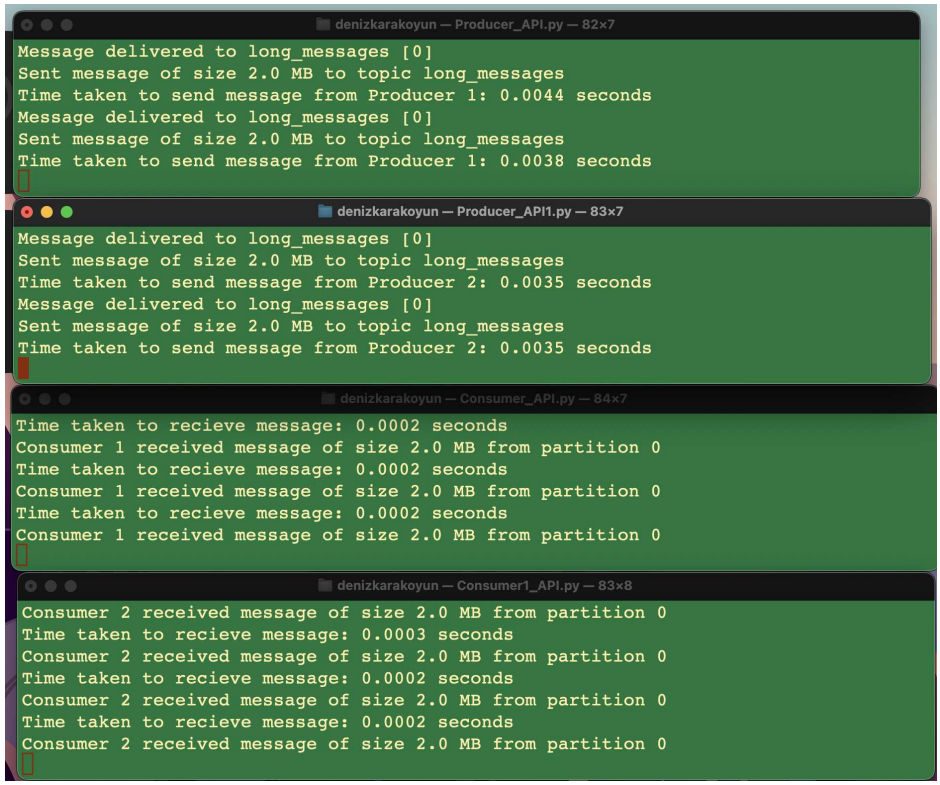

Figure 40: Shows a slight increase in both send and receive times when the message size is doubled to 2 MB, indicating a consistent increase in latency proportional to the message size.

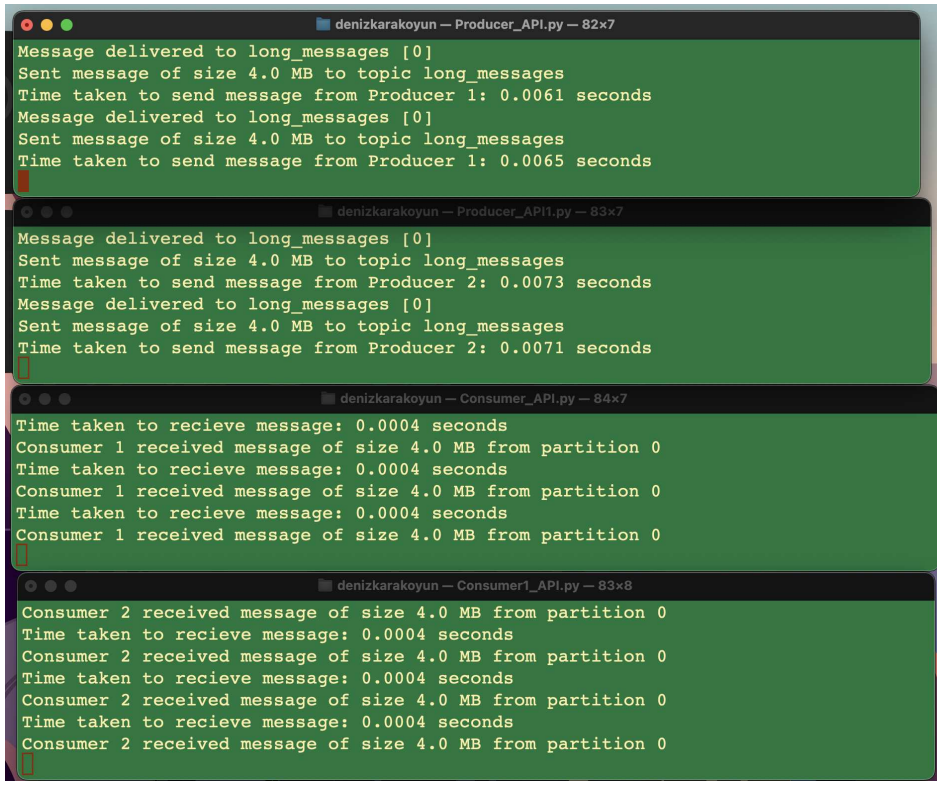

Figure 41: Depicts the timeline for a 4 MB message, showing further increased latency in sending and receiving messages compared to smaller sizes, following the trend observed in previous figures.

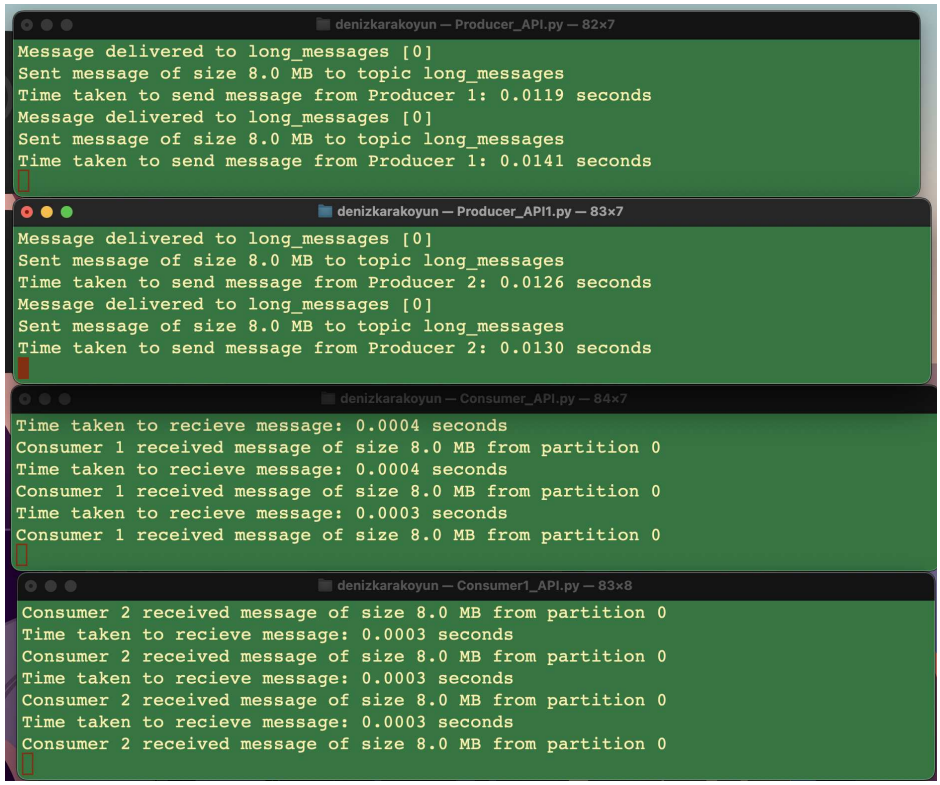

Figure 42: Demonstrates a steady increase in latency as the message size reaches 8 MB, maintaining the linear pattern of growth seen in smaller sizes.

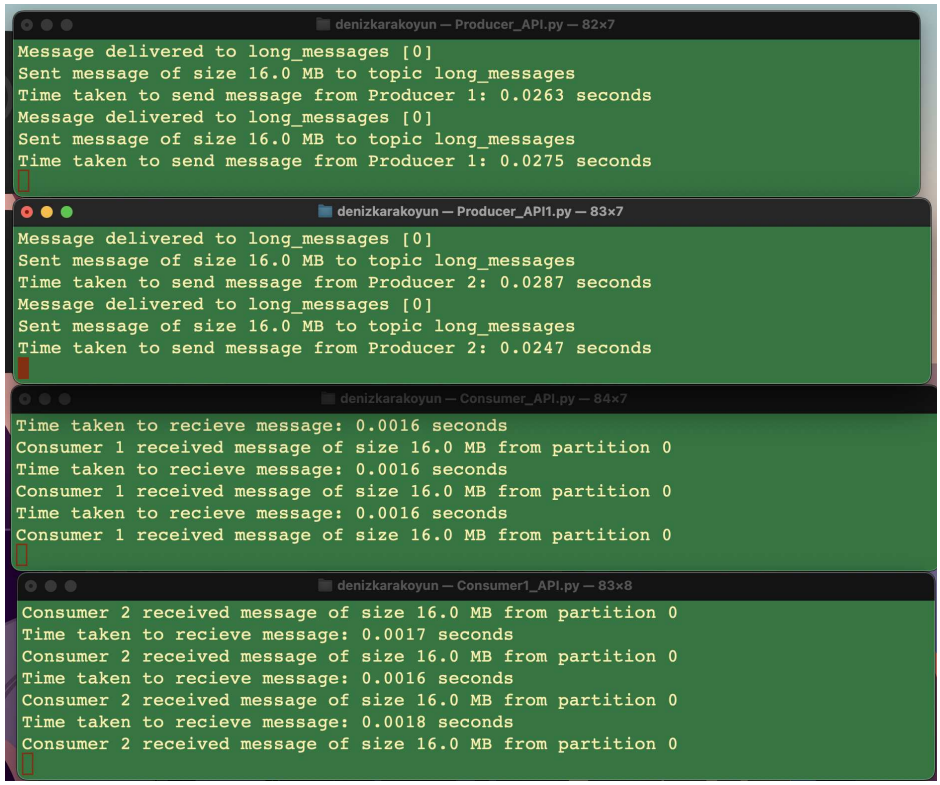

Figure 43: Shows the messaging timeline for a 16 MB message size, reflecting a noticeable increase in both send and receive times as data volume increases, aligning with the overall trend.

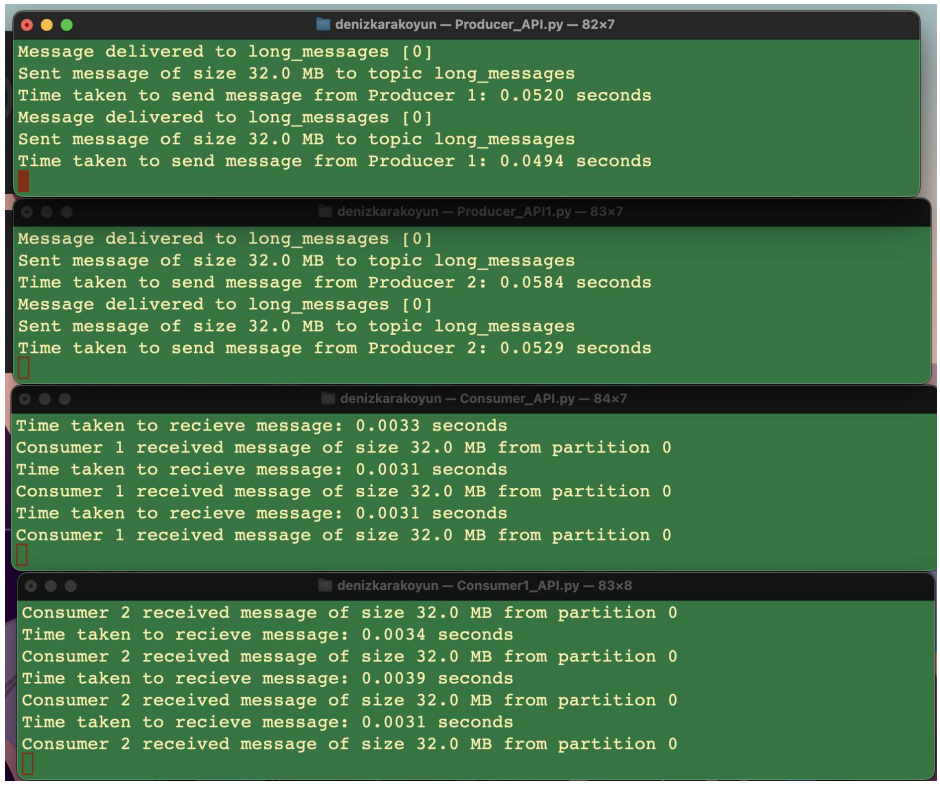

Figure 44: Captures the performance for 32 MB messages, where latency continues to rise, reinforcing the direct correlation between message size and delivery time.

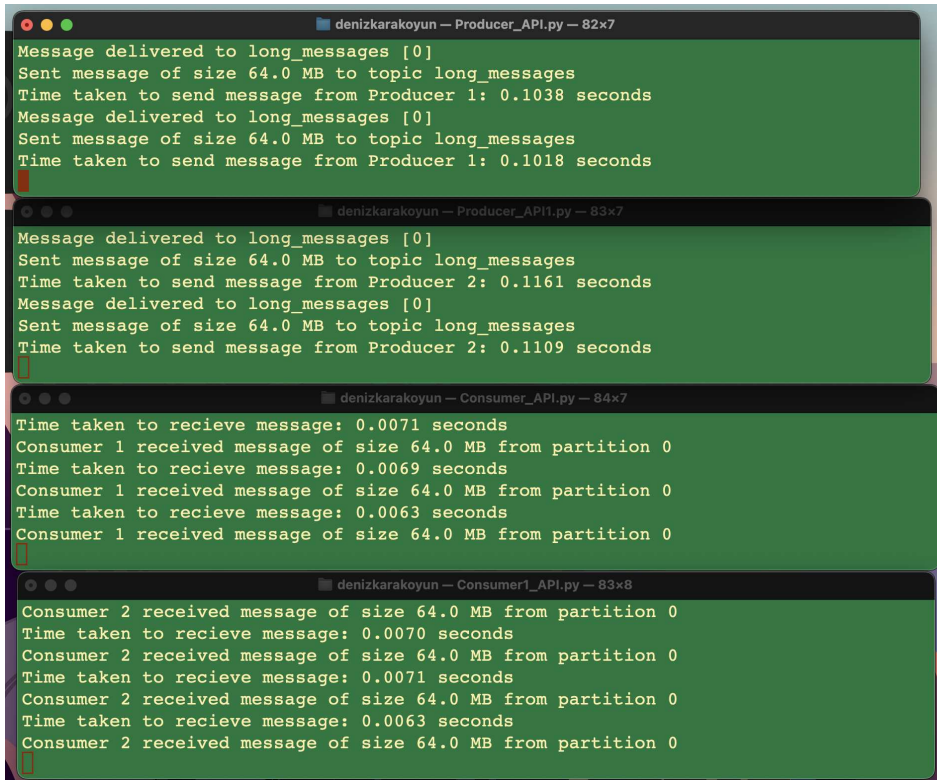

Figure 45: Presents the 64 MB message scenario, illustrating a significant jump in both sending and receiving times. Although there are variations, the increase is still consistent with previous patterns, as seen in the table.

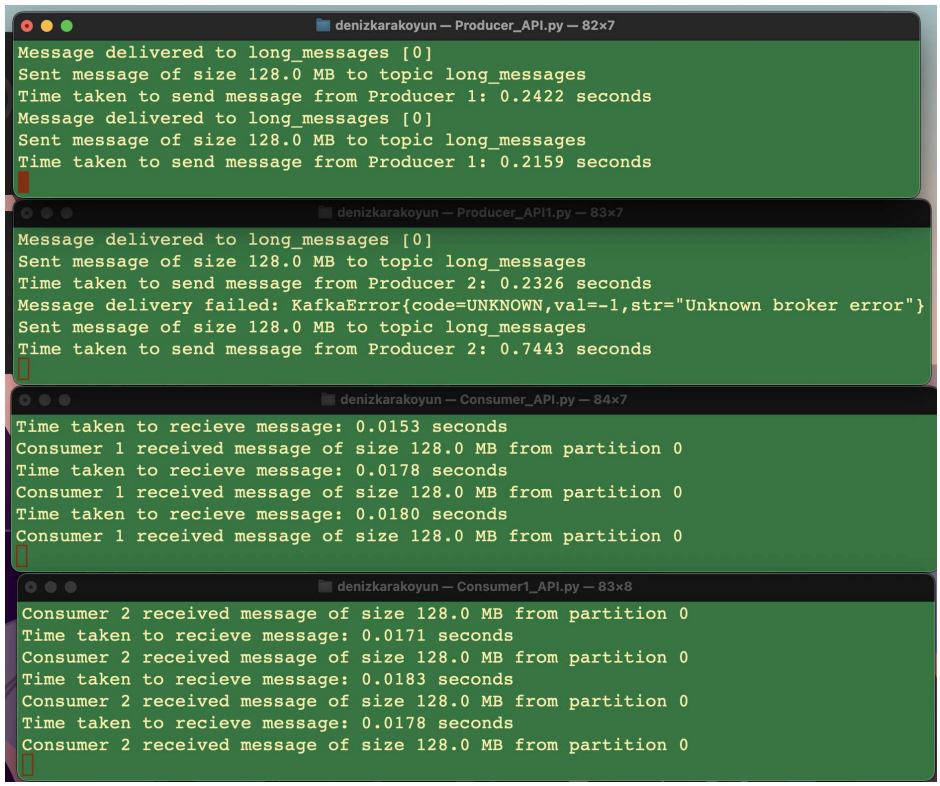

Figure 46: Shows the messaging timeline for the largest message size of 128 MB. The sending time is significantly higher, closely matching the previous trends seen in earlier graphs. With the aid of the table, it's clear that the latency grows progressively with the message size, allowing us to effectively analyze and compare the performance across different scenarios.

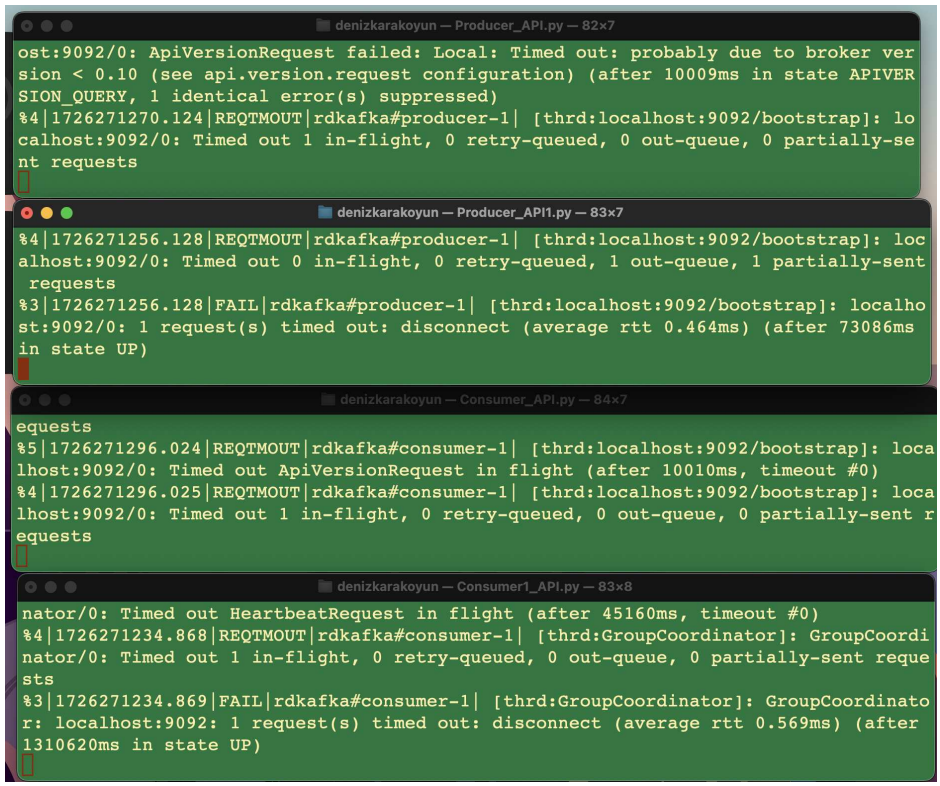

Figure 47: Shows the messaging timeline for 256MB which is larger than the limit for my computer which is 128MB