Latency Testing

But first...

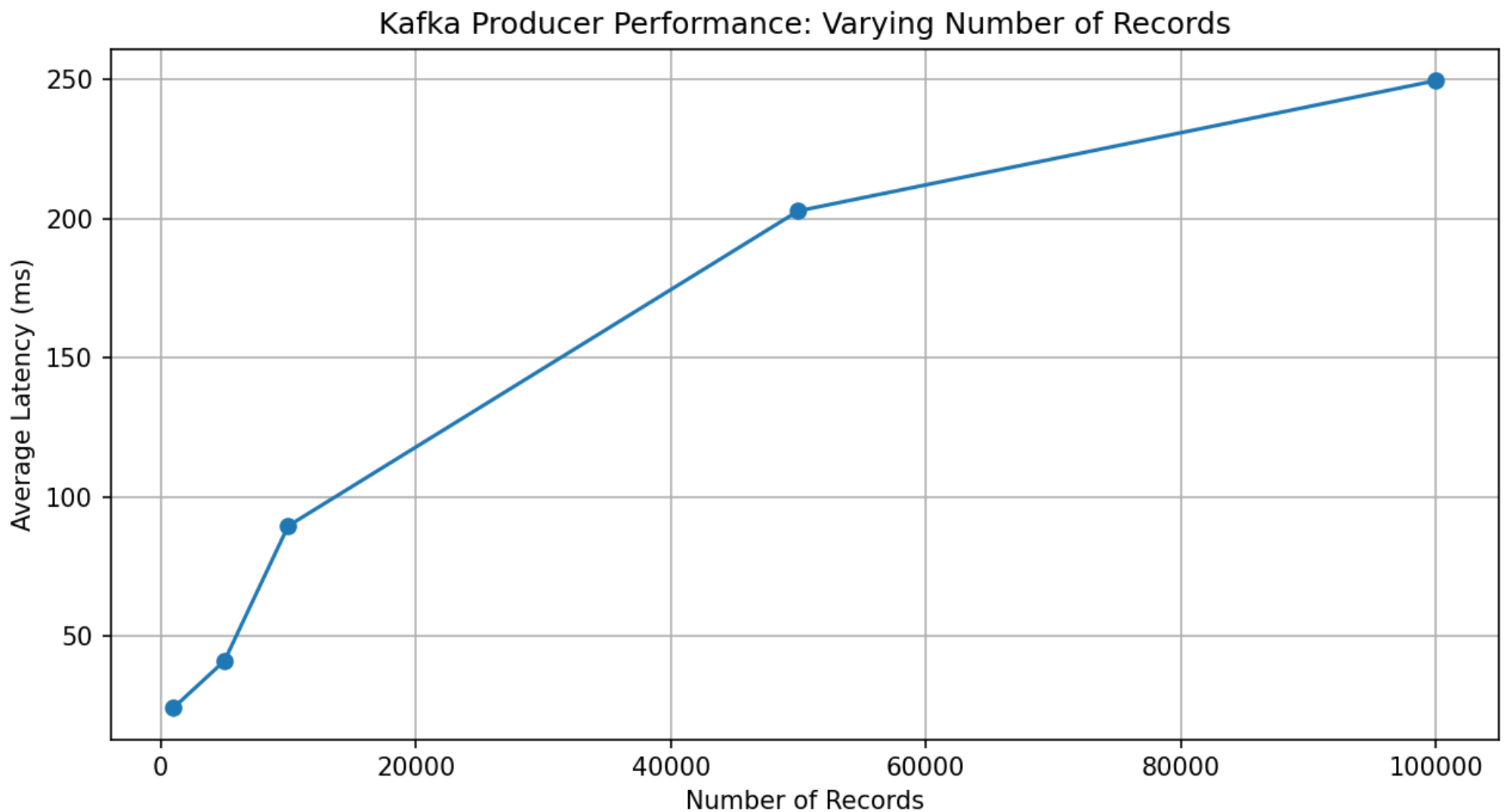
As you may be aware, the government is facing a potential shutdown on March 14th at 11:59pm EST. We want to inform you of the following updates regarding your spring internship:

1. **Orderly Shutdown:** In the event of a shutdown, interns will report to work on Monday, March 17, to carry out orderly shutdown activities.

2. **Your Internship Will Be Paused**: Internships are not considered mission critical and will be paused for the entire duration of a government shutdown. You will not report to work during this time.

First Test: Generic Test Based on numRecords

- Using Kafka's built-in performance testing functionality
- Varying number of records from 1-100,000


Kafka Producer Performance: Varying Number of Records
Average Latency (ms)
250
200
150
100
50
0
20000
40000
60000
80000
100000
Number of Records

Second Test: Authentication API

- Testing average latency of authentication (using private_key.pem to see if a user is permitted to consume from a given topic [ex. example_agent_4]) functionality of HSML API

Screenshot of Python Script

```
Received message: {"status": "active", "producer_timestamp": 1741703883941}
End-to-end latency: 3 ms
Received message: {"status": "active", "producer_timestamp": 1741703885039}
End-to-end latency: 4 ms
Received message: {"status": "active", "producer_timestamp": 1741703886158}
End-to-end latency: 3 ms
Received message: {"status": "active", "producer_timestamp": 1741703887249}
End-to-end latency: 3 ms
Received message: {"status": "active", "producer_timestamp": 1741703888371}
End-to-end latency: 3 ms
Received message: {"status": "active", "producer_timestamp": 1741703889446}
End-to-end latency: 5 ms
Received message: {"status": "active", "producer_timestamp": 1741703890539}
End-to-end latency: 3 ms
```

How Latency is Calculated

- producer.py uses producer.timestamp(), saving the timestamp of when the message is first sent to the consumer
- This timestamp is included with the message sent to consumer
- consumer.py takes the timestamp value and subtracts it from the current time at consumption, to determine how many milliseconds passed between production and consumption

Average Latency of Authentication

- Average Latency: 3.42ms
  - On a local server (localhost producer to localhost consumer)
  - Across 66 connection attempts

Thoughts

- In a local environment (ex. FUTURAMA), seemingly very unlikely we'll ever go over 50ms
- Tested record size is close to the size of real-world positional data we'll eventually send:
  - Tested authentication record size: 57 bytes
  - Record size of a JSON containing X, Y, Z, and rX, rY, rZ: 58 bytes

Next Steps

- New test:
  - Set up producer.py on EC2
  - Set up consumer.py on local machine
- But first I need to:
  - Set up DDNS to resolve changing IP issue of EC2

Addendum: DDNS

- Provider:
  - DuckDNS
- Why:
  - Allows us to set up a stable subdomain (HSML.duckdns.net) that will handle redirecting to our dynamic EC2 IP
  - Alternatives: manually editing IP address on all scripts frequently, or paying for a static IP from AWS
- Subtasks:
  - Set up CRON job to pass dynamic IP to DuckDNS servers (DONE)
  - Set up port forwarding so we can call the EC2 from our local machines (WIP)