MQTT - asynchronous messaging

Overview

This project demonstrates the implementation of MQTT (Message Queue Telemetry Transport) protocol for IoT device communication. MQTT is a lightweight messaging protocol designed for constrained devices and low-bandwidth, high-latency, or unreliable networks, making it ideal for IoT applications.

System Architecture

The implementation consists of three main components:

1. **MQTT Broker (Mosquitto)**
    - Acts as the central message hub
    - Handles message routing between publishers and subscribers
    - Manages client connections and message queues
2. **Publisher (IoT Sensor Simulator)**
    - Simulates an IoT device sending sensor data
    - Publishes 1,000,000 messages to demonstrate high-volume data transmission
    - Implements message counting and delivery confirmation
3. **Subscriber (Data Consumer)**
    - Receives and processes messages from the publisher
    - Tracks message reception
    - Verifies data integrity through message counting

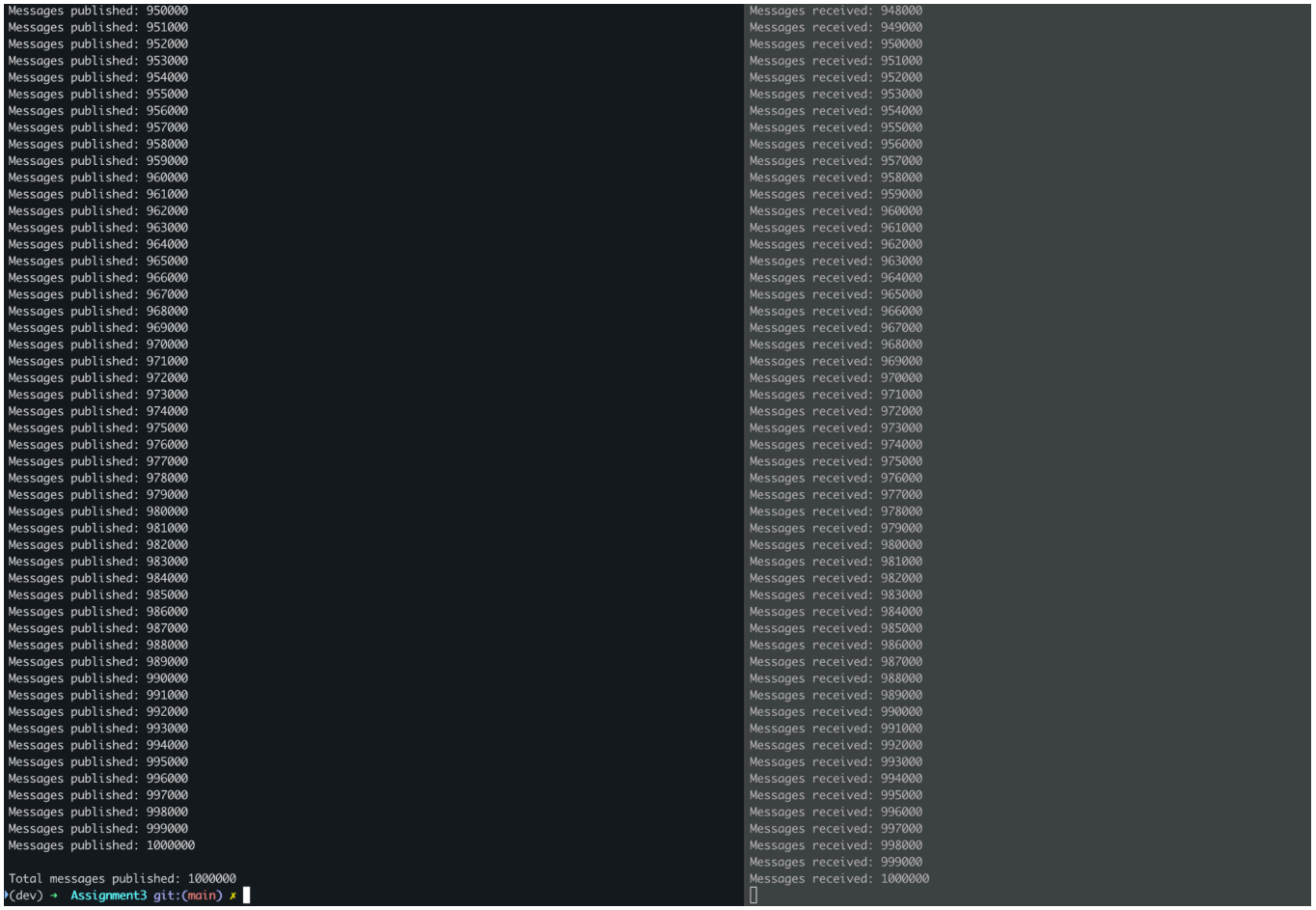
Command Reference

1. Start Mosquitto:
   brew services start mosquitto
2. Run Subscriber
   python3 mqtt_subscriber.py
3. Run Publisher
   python3 mqtt_publisher.py

Implementation Features

- **Quality of Service (QoS) Level 1**: Ensures at-least-once message delivery
- **Message Tracking**: Implements counters on both publisher and subscriber sides
- **Progress Monitoring**: Reports progress every 1000 messages
- **Error Handling**: Robust error management and reporting
- **Clean Shutdown**: Proper resource cleanup and connection termination

Execution



Conclusion

This implementation demonstrates the practical application of MQTT in IoT scenarios, highlighting:

- Scalability for high-volume message handling
- Reliability through QoS implementation
- Robust error handling and monitoring
- Practical considerations for real-world deployment

The successful transmission and verification of 1,000,000 messages proves the system's capability to handle substantial IoT communication loads while maintaining reliability and performance.