Design And Implementation Of An Mqtt-Based Internet Of Drone Things For Swarm Drone

1st Ratna Sari
Faculty of Electrical Engineering
Telkom University
Bandung
ratnasaru@student.telkomuniversity.ac.id

2nd Nyoman Bogi Aditya Karna,
Faculty of Electrical Engineering
Telkom University
Bandung
aditya@telkomuniversity.ac.id

3rd Arif Indra Irawan
Faculty of Electrical Engineering
Telkom University
Bandung
arifirawan@telkomuniversity.ac.id

***Abstract*— Swarm drones are drones that communicate with each other, the more drones that communicate, the heavier the communication network, therefore the aim of this research is to be able to make light communication between two drones using the Message Queuing Telemetry Transport (MQTT) data protocol. This thesis uses NodeMCU tools. The protocol used is MQTT in the use of communication between drones. Because of the shortcomings of the HTTP protocol, the MQTT server protocol must be implemented to support the development of the IoT platform. MQTT is a lightweight and simple communication protocol. The plan outcome is that two drones will be able to communicate reliably utilizing the MQTT data protocol. After conducting the test, the results of testing network quality with QoS parameters are delay, jitter, packet loss, and throughput. The average delay of MQTT QoS 0 is 0.103 s, MQTT QoS 1 is 0.111 s, and HTTP is 0.124 s. HTTP and MQTT QoS 0 get 0% packet loss, while MQTT QoS 1 gets 0.1% packet loss. Throughput on the MQTT protocol is faster than the HTTP protocol. And the network quality of the MQTT protocol is better than HTTP.**

***Keyword*— MQTT,HTTP, IoT, Subscribe, Publish and Broker**

I. INTRODUCTION

Air quality levels are constantly changing. Vehicle emissions and climate change [1]. The current air quality measurement system is deemed unreliable because it only measures specific points on the ground, resulting in an inaccurate result that is influenced by a variety of factors. Certain regions are in accessible when measuring on the ground, and only a limited range of area can be measured, Because of this ground access is usually hampered and obstructed, the most practical way is to implement a mobile air quality monitoring robot such as Unmanned Aerial Vehicles (UAV) [2]. The UAV, also known as a ”drone,” is a vehicle that does not have a human pilot aboard it. [3]

Many drones are required for mapping air quality in a location in order to control air quality. Drones must be able to communicate with one another in order to collect accurate data on air quality. The Hypertext Transfer-Transfer Protocol is being used in the development of existing technology to facilitate communication between drones (HTTP). The drone will be able to communicate via HTTP, allowing the data collected to be directly stored using the Internet of Things. However, in its use, HTTP has several drawbacks such as the bandwidth usage is quite large and the packet size is large, so it is not reliable to run on systems that have low bandwidth or high latency.

As a result, an innovative idea is made in this thesis by adding a MQTT system to the Drone of Things to facilitate communication and data storage. MQTT employs a ”publish-subscribe” communication model, which eliminates the need for clients to update themselves. MQTT is a very simple and lightweight communication protocol, it consumes fewer resources, making this model ideal for low bandwidth environments. The MQTT Protocol is also designed for devices with limited capabilities, low bandwidth, high latency, and less reliable networks. Previous research, namely the MQTT-based Secured Home Automation System, strengthens the case for using the MQTT protocol as the network’s ap plication protocol and performance evaluation of MQTT and CoAP via common 2 middleware. According to the two studies above, the MQTT protocol consumes very little energy when compared to other protocols and can function well in low bandwidth and high-latency environments [4].

II. DESIGN AND IMPLEMENTATION

A. System Design

In this thesis, a system innovation is made by integrating the MQTT system into Drone of Things to facilitate communication and data storage, so it can be used as a personal computer (PC) to access data protocols. In this thesis, we use a Pixhawk drone that functions on autopilot during the flight process and uses 2 NodeMCU, the first NodeMCU is installed on the Mother Drone and its function is to send messages between the drones so they can communicate in 2 directions. The second NodeMCU is installed on Followers Drone and its function is to receive messages between drones so it can follow the mother drone. NodeMCU is connected to the temperature sensor and will then publish data to the IoT platform using Antares, the IoT platform can also act as a broker which can receive MQTT messages and subscribe directly to the session. After that someone publishes, subscribes, and will immediately retrieve data using Wireshark.

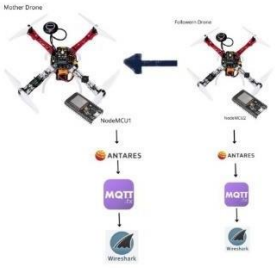

FIGURE 1
Design System

B. MQTT (Message Queue Telemetry Transport)

stated in the reference [4] Message Queue Telemetry Transport (MQTT) is an application layer protocol for machine-to-machine (M2M) data communication. MQTT is a lightweight message, which means that it communicates by sending data messages that have a small header of only 2bytes for each type of data, allowing it to work in resource-limited environments such as low bandwidth and limited power resources.

C. How MQTT Works

MQTT is a standards-based messaging protocol, or set of rules, used for machine-to-machine communication. Smart sensors, wearables, and other Internet of Things (IoT) devices typically must send and receive data over networks with limited resources and bandwidth.The MQTT protocol is described below. The MQTT client communicates with the MQTT broker. Clients can publish messages, subscribe to specific messages, or do both once connected. When the MQTT broker receives a message, it forwards it to any customers who are interested.

FIGURE 2
How MQTT Works

D. NodeMCU

Node Micro Controller Unit is a free and open source IoT platform that can be use anywhere. It includes firmware with a Espressif System ESP8266 Wi-Fi module, an ESP-12 microcontroller, and firmware written in a programming language. Because they are both microcontrollers, node MCU and Arduino have the same appearance. There is a lot of microcontroller development going on. NodeMCU is useful when used as a IoT based control [4].

E. Aduino IDE

The Arduino IDE is a piece of open source software that is primarily used for writing and compiling code for the Arduino Module. the term ”Integrated Development Environment” (IDE) refers to a piece of software developed by Arduino.The C language that is primarily used for editing, compiling, and uploading code to the Arduino device. undefined almost all Arduino modules are compatible with this open source software, which is simple to install and begin compiling code on the fly [5].

F. QoS Measurement Scenarios

This thesis topology consists of four components these are NodeMCU, access point, cloud, and laptop. Firstly, NodeMCU is a micro-controller, its function is to send data. Secondly is access point, that functions as an internet center and provides internet services for the NodeMcu. Thirdly cloud it is used as an iot platform in this thesis, by using Antares. Lastly the laptop is used as a subscriber or data receiver. The work flow of NodeMcu as a publisher or uploaded data. After that, it will immediately be kept on the IoT platform. It can also be called a broker if it is directly connected to the subscriber or the laptop.

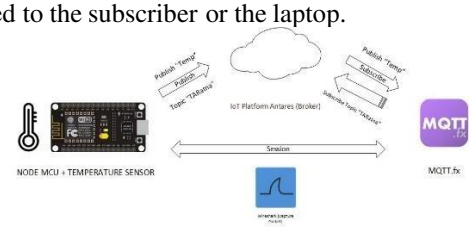

FIGURE 3
QoS Measurement Scenarios

III. Analysis of Test Results

A. Delay

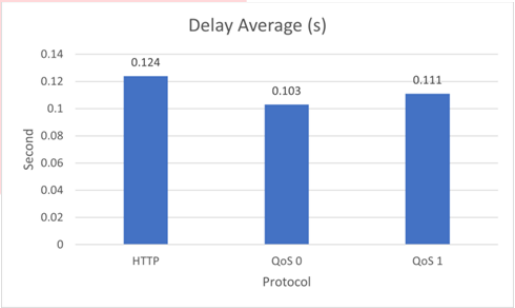

FIGURE 4
Delay average for HTTP and MQTT protocols

The results of the MQTT delay test that have been obtained will be made into an average delay and then compared with the HTTP delay test results obtained from previous research. In Figure 4.2, it can be seen that the average delay for HTTP is 0.124 s, while for MQTT QoS 0, it is 0.103, and for QoS 1, it is 0.111. It can be concluded that HTTP has a higher average delay value than MQTT, although the difference is not much, QoS 0 has a lower average delay on MQTT than QoS 1. This can be attributed to the different ways of sending on QoS 0 and QoS 1. In QoS 0, the delivery is only done once and without confirmation, while in QoS 1, confirmation is needed so that this can affect the delay, although not too significantly. Even though the delay between HTTP and MQTT is different, the two protocols are in the good category based on ITU-T parameters.

B. Packet Loss

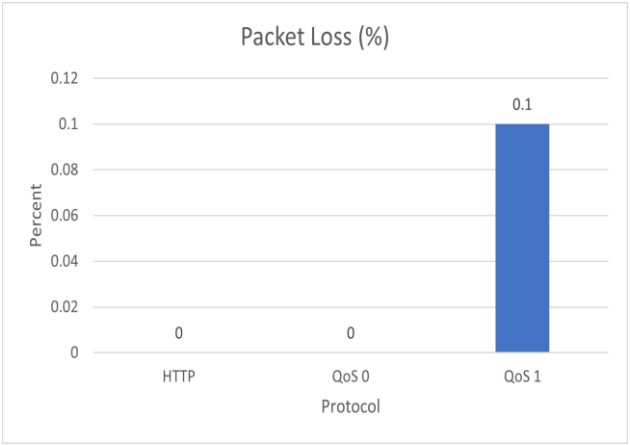

FIGURE 5
Packet Loss in the HTTP and MQTT Protocols

Based on the results obtained from the previous test, HTTP and MQTT QoS 0 obtained results of 0% while MQTT QoS 1 was 0.1%. Even though MQTT QoS 0 should have a higher packet loss due to the fire and forget delivery system which when sent, will be immediately forgotten and will not be re-transmitted if it is not conveyed to the recipient. Even so, there are other factors that can affect packet loss, namely the quality of the network on the sending and receiving sides. In this case, it is possible that when testing on QoS 1, there is a momentary loss of connection, causing packet loss. So, based on the ITU-T HTTP protocol, MQTT QoS 0 is in the ”very good” category, and MQTT QoS 1 is in the ”good” category.

C. Firebase System Work Flow Throughput

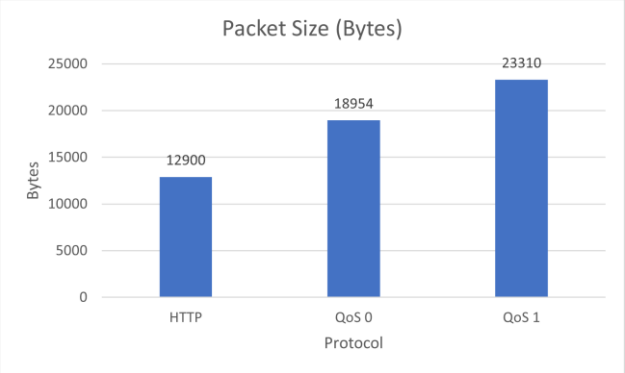

FIGURE 6
Packet Size for HTTP and MQTT Protocols

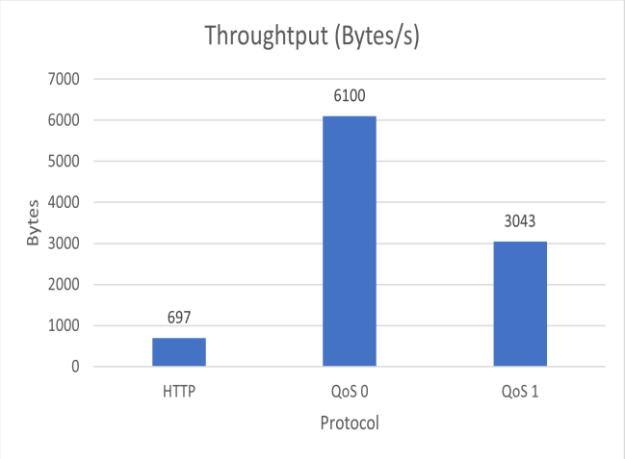

FIGURE 7
Throughput of the HTTP and MQTT Protocols

Based on the results of the throughput test that has been carried out, the results obtained are HTTP of 697 bytes/s, MQTT QoS 0 of 6100 bytes/s, and MQTT QoS 1 of 3043 bytes/s. MQTT QoS 0 gets the highest value when compared to HTTP and MQTT QoS 1, this can happen because of two main factors that influence it, namely the size of the package and the observation time. In the test conducted QoS 0 has a packet size of 18954 Bytes with an observation time of 3107 so that 6100 bytes/s are obtained, while in QoS 1, the packet size is 23310 bytes with an observation time of 7.660 s, so that 3043 bytes/s are obtained. So it can be concluded that the large throughput of each protocol can be caused by these two factors.

D. Delay Variation Analysis

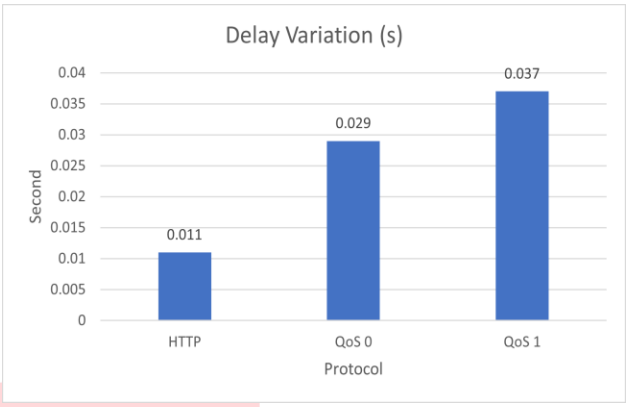

FIGURE 8
Delay Variation in HTTP and MQTTProtocols

Based on the results of the delay variation test with the calculation of the previous formula applied to the delay test, it can be concluded that QoS 0 has the lowest delay variation of 0,011 s the highest is MQTT QoS 1 of 0.037 s. So, HTTP is the best among the others because the lower the jitter, the better the quality of a network.

IV. Conclusions

A. Conclusion

In this thesis, the author uses the MQTT communication protocol, which it is implemented on the Antares IoT platform. MQTT QoS 0 is better than HTTP and MQTT QoS 1 in terms of delay, packet loss, throughput, and delay variations. HTTP is better, but from a study of the literature, there is a MQTT communication protocol, a protocol that is created to be low-power and for the IoT communication protocol.

The results of the MQTT delay test that have been obtained will be made into an average delay and then compared with the HTTP delay test results obtained from previous research. In Figure 4.2, it can be seen that the average delay for HTTP is 0.124 s, for MQTT QoS 0 is 0.103, and for QoS 1 is 0.111. It can be concluded that HTTP has a higher average delay value than MQTT, even though the difference is not much. MQTT QoS 0 has a lower average delay compared to MQTT QoS 1 this can be caused by the different ways of sending QoS 0 and QoS 1. In QoS 0, the delivery is only done once and without confirmation, while in QoS 1, it needs confirmation so that it can affect the delay, even though it is not too significant. So, even though the delay between HTTP and MQTT is different, the two protocols are in the good category based on ITU-T parameters. The results of the MQTT test on the IoT platform with the packet loss parameter obtained from the previous HTTP and MQTT QoS 0 tests obtained a result of 0%, while MQTT QoS 1 was 0.1%. Even though it should be MQTT QoS 0 which has a higher packet loss due to the nature of the delivery system, when sent it will be immediately forgotten and will not retransmit if it is not conveyed to the recipient. Another factor that can affect packet loss is the quality of the network on both the sending and receiving sides. In this case, it is possible that when testing on QoS 1, there is a momentary loss of connection and causing packet loss. So, based on the ITU-T HTTP protocol, MQTT QoS 0 is in the ”very good” category, and MQTT QoS 1 is in the ”good” category. he

results of the MQTT test on the IoT platform with the throughput parameters that have been carried out are HTTP results of 697 bytes/s, MQTT QoS 0 of 6100 bytes/s and MQTT QoS 1 of 3043 bytes/s. MQTT QoS 0 gets the highest score when compared to HTTP and MQTT QoS 1. This can happen because of two main factors that influence it, namely the size of the packet and the observation time. In the test carried out, QoS 0 has a packet size of 18954 bytes with an observation time of 3107 s to obtain 6100 bytes/s while in QoS 1, the packet size is 23310 bytes with an observation time of 7.660 s to obtain 3043 bytes/s. So it can be concluded that the large throughput of each protocol can be caused by these two factors. The results of the MQTT test on the IoT platform with delay variation parameters can be concluded that QoS 0 has the lowest delay variation of 0.103 s and the highest is HTTP of 0.124 s. So, QoS 0 is the best among the others because the lower the jitter, the better the quality of a network.

BIBLIOGRAPHY

[1] D. J. Jacob and D. A. Winner, “Effect of climate change on air quality,” Atmospheric environment, vol. 43, no. 1, pp. 51–63, 2009.

[2] M. Dunbabin and L. Marques, “Robots for environmental monitoring: Significant advancements and applications,” IEEE Robotics & Automation Magazine, vol. 19, no. 1, pp. 24–39, 2012.

[3] H. Eisenbeiss et al., “A mini unmanned aerial vehicle (uav): system overview and image acquisition,” International Archives of Photogrammetry. Remote Sensing and Spatial Information Sciences, vol. 36, no. 5/W1, pp. 1–7, 2004.

[4] H. A. Rochman, “Sistem kendali berbasis mikrokontroler menggunakan protokol message queuing telemetry transport (mqtt) pada smarthome,” Ph.D. dissertation, Universitas Brawijaya, 2017.

[5] M. Fezari and A. Al Dahoud, “Integrated development environment “ide” for arduino,” WSN applications, pp. 1–12, 2018.