Comparison Analysis of MQTT and HTTP Communication Protocol IOT Based on Website Node.Js

Fernanda Imansyah[1], Anggi Zafia*[2]
[1,2]Department of Informatics, Telkom University, Jl.DI.Panjaitan 128,
Banyumas, 53147, Jawa Tengah, Indonesia
e-mail: [1]19102233@ittelkom-pwt.ac.id, *[2]anggiz@telkomuniversity.ac.id

Abstract—The Internet of Things is one of the technologies that is widely used in monitoring systems to get maximum results. Within the IoT infrastructure, there are devices and servers, both of which must be listed. These two devices must be connected via a communication protocol such as ZigBee, MQTT, HTTP, and so on. The selection of an appropriate and effective communication protocol is one of the most important things. According to the results of the 2018 edition of the IoT developer survey by the Eclipse Foundation, it was found that the communication protocols that are widely used in IoT systems are the MQTT protocol, which uses a publish/subscribe architecture, and the HTTP protocol, which uses a request/response architecture. This study conducted a comparative analysis of the MQTT and HTTP communication protocols on the internet of things system based on the Node.js website on the Google Cloud platform for monitoring systems for plants as a research object. Data retrieval was carried out in the morning and at night using wireshark software. It was found that the MQTT protocol had better throughput and delay values compared to the HTTP protocol, with an average MQTT throughput value of 1730 in the morning and 1899 at night. The average MQTT delay value was 0.93485406 in the morning and 0.855531576 at night. While the HTTP protocol has better packet loss and jitter values compared to the MQTT protocol, with HTTP packet loss values of 0.028% in the morning and 0.018% at night, the average value of HTTP jitter is 0.044957 in the morning and 0.014849 at night

Keywords— HTTP, Internet of Things, MQTT, QoS

1. INTRODUCTION

Current technological developments make it possible to connect electronic devices to the internet network and exchange data [1], [2]. The Internet of Things is a technology that can be used for monitoring systems in various fields, such as industry, health, animal husbandry, agriculture, and so on [3], [4], [5]. Various research projects related to IoT have been developed. However, the researchers did not discuss the type of communication protocol used. Choosing an appropriate and effective communication protocol is an important thing [6]. Unlike websites that use a single standard communication protocol, HTTP, IoT cannot rely on one communication protocol for all needs, so there are many different types of communication protocols for IoT systems available today [7] [8]. In the IoT infrastructure, there are devices and servers, both of which must be connected. To connect an IoT system via the internet, you must build a client-server architecture so that the local system and server can be connected. These two devices must be connected via communication protocols such as ZigBee, MQTT, HTTP, and so on [9]. According to the results of the 2018 edition of the IoT developer survey by the Eclipse Foundation [10], it was found that the communication protocol that is widely used in IoT systems is MQTT at 62.61%, followed by the HTTP protocol at 54.10%. MQTT follows a publish/subscribe architecture and allows simple data transfer between different devices. The server or broker provides the information, while the client can read the information available after subscribing by accessing the appropriate topic. While HTTP was created for two systems communicating with each other at one time, this protocol forms the basis of data communication over the web and is used for IoT devices when there is a lot of data to be published [11] [12]. In this research, a comparative analysis was carried out regarding the MQTT and HTTP communication protocols for plant monitoring systems as implementation media and objects for data collection. The aim of this research is to compare the MQTT and HTTP protocols against the internet of things system based on the Node.js website using the Nodemcu microcontrol device using DHT22, BH1750, and capacitative soil moisture sensors.

2. BASIC THEORY

A. *MQTT*

Message Queuing Telemetry Transport is a publish/subscribe message-based communications protocol designed for lightweight machine-to-machine communications. MQTT was originally developed by International Business Machines Corporation (IBM). The MQTT protocol is message-oriented, and each message is published on a topic. The MQTT architecture has a client-server model where each sensor is a client connected to a server [13], [14].

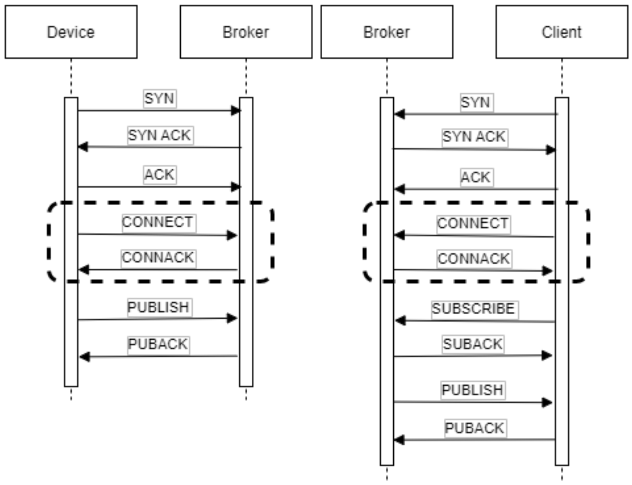

Fig.1. MQTT Communiaction

The server is known as a broker that uses TCP communication. Every client who subscribes to a topic will receive every message published on that topic. [15].

B. *HTTP*

Hypertext Transfer Protocol is a request/response protocol between a client and a server where the client requests information from the server (request) and the server responds to the request (response) accordingly [11], [16].

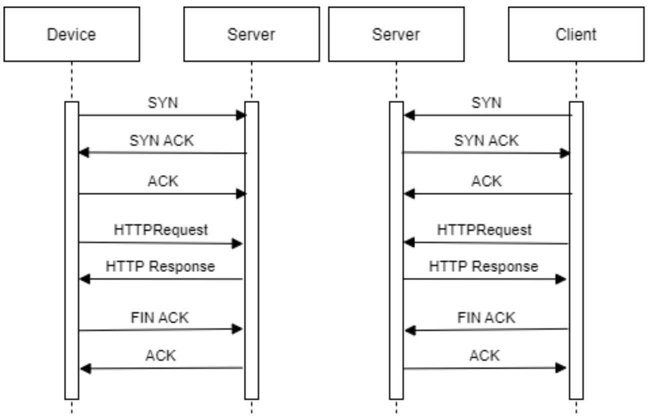

Fig.2. HTTP Communiaction

HTTP data sits on top of the TCP protocol, which ensures delivery reliability by breaking down large data requests and responses into network-manageable chunks. However, the connection is created on each access because the accessed data is transferred based on the IP address and URL, whose relationship changes dynamically. In short, after many attempts at releasing the connection, the communication is complete. Therefore, the HTTP Protocol is usually used for Internet of Things systems that have a lot of data to publish. [7], [15].

C. *Quality of Service*

QoS is the use of technological mechanisms that work on a network to control traffic and ensure application performance with limited network capacity [17].

1) Throuhput

Throughput is obtained from the number of packets that successfully reach the destination and the packet size of the total simulation time in seconds [18]. The formula for calculating Throughput is given in the following Equation

$$Throughput = \frac{Data\ packet\ received}{Observation\ time} \quad (1)$$

2) Jitter

Jitter is the variation in latency or time delay when a signal is transmitted and received. This difference is measured in milliseconds (ms), the higher the jitter score, the more inconsistent the response time. Jitter is often caused by network congestion and route changes [19]. The formula for calculating Jitter is given in the following Equation:

$$Jitter = \frac{Total\ variation\ of\ Delay}{Total\ received\ packets - 1} \quad (2)$$

3) Packet Loss

Packet loss is the total or amount of data lost, usually displayed as a percentage of lost packets sent. Packet loss can be described as lost data packets that do not reach their destination after being sent over the network. Packet loss occurs when network congestion, hardware problems, software bugs, and a number of other factors cause packets to drop during data transmission [20]. The formula for calculating Packet Loss is given in the following Equation:

$$Packet\ Loss = \frac{((Packets\ sent - packets\ received) \times 100\%}{Data\ packets\ sent} \quad (3)$$

4) Delay

Delay is the total time of a packet, which is calculated from the time it starts sending the packet until the packet is successfully sent to the destination. Delay is calculated using the time difference between packets sent from the source and received at the destination [19]. So the average delay value is obtained by dividing the total packet delay by the total number of packets received. The formula for calculating Packet Loss is given in the following Equation:

$$Delay = \frac{Total\ Delay}{Total\ packets\ received} \quad (4)$$

3. METHOD

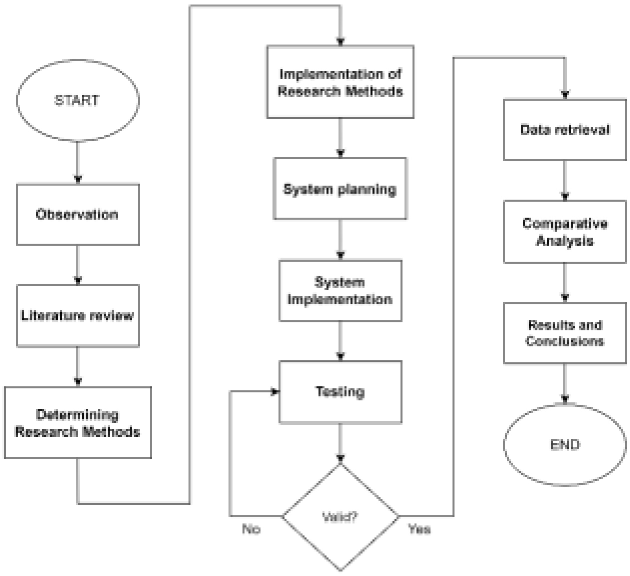

Fig.3. Research Flow Chart

Observations are made on the types of communication protocols used for various Internet of Things systems. Observations were also made by reading several references regarding the advantages and disadvantages of several IoT communication protocols. Literature review was carried out to determine the relevant relationship between this research and previous research so that it can be used as a reference and also to avoid similarities in the research to be conducted. A literature study was carried out by reading and comparing books, journals, papers about communication protocols used in internet of things systems, and related articles. The method used in this research is the R&D method. This research uses research and development research because it makes it possible to understand the validity of a product in a shorter time. The R&D process often involves repeated testing and evaluation to ensure that the system functions well in a real environment and meets specified requirements.

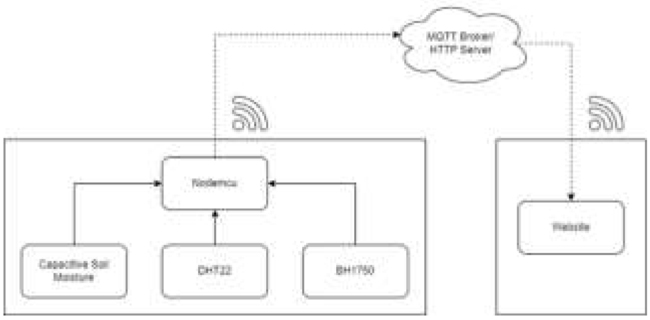

Fig.4. System Block Diagram

Block diagram of the system or how the plant monitoring system works. Sensor data is sent by Nodemcu using the MQTT and HTTP protocols via the internet network. Nodemcu also functions as a link between sensors, the internet, an MQTT broker, and an HTTP server. Soil moisture, temperature, and light intensity data will be displayed via the Node.js website using MQTT and HTTP protocols.

The AsyncMqttClient library is needed to use MQTT with the ESP8266. Then initiate the MQTT host or broker, port, and MQTT topic that will be used. The broker used in this research is Mosquitto with port 1883. Mosquitto is a broker that uses the MQTT protocol so that devices can communicate by sending and receiving messages. Mosquitto consists of components that connect to a server for sending messages, publishing, and subscribing to a topic.

Fig.5. MQTT Workflow

First, connect Nodemcu with the Mosquitto broker (test.mosquitto.org). If it is connected, the nodemcu will then read the sensor and publish the sensor data into a topic on the mosquitto broker. Node.js acts as a subscriber to a topic that is used to obtain and display sensor data on the website.

The HTTP protocol uses the HTTP POST request method with an encoded URL request type. In implementing the HTTP protocol, the WiFiClientSecure library is needed to make nodemcu a client.

Fig.6. HTTP Workflow

First, connect the nodemcu to the http server host or website used. Next, the nodemcu will read the sensor and send the sensor data using the HTTP post method. After that, Node.js will perform an HTTP get to get and display sensor data on the website.

4. RESULT AND ANALYZE

A. System Implementation

Fig.7. Hardware Systems

This research uses several hardware devices consisting of a nodemcu, capacitive soil moisture sensor, DHT22 sensor, and BH1750 sensor.

Fig.8. Software Systems

In this research, we created a website Node.js programming language uses Express.js Framework. This website uses several required libraries, including the http library, which is used to create an API server to post sensor data and also make requests to the API server to obtain sensor data. Then there is the MQTT library, which functions as a subscriber for a topic in the MQTT broker. This library is used to get sensor data from the MQTT broker by subscribing to a topic. Then there is a pusher library, which functions as a communication protocol layer between client servers so that it can run in real time and scalably via WebSocket.

Wireshark software captures packets sent to the Node.js website. Data retrieval is done by accessing the Node.js website and connecting Wireshark to the WiFi network used. Then filter the data packets using the IP address of the Node.js website that is being accessed.

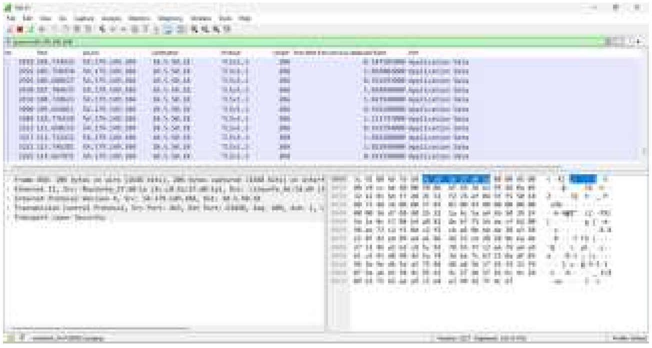

Fig.9. Wireshark Packet Capture

Data collection was carried out in the morning between 06.00 and 23.00 (peak hours) and in the night between 23.00 and 06.00 (off-peak hours). In Fig.9, there are several MQTT or HTTP packets published by Nodemcu. The packet data will be analyzed to measure throughput, jitter, packet loss, and delay parameters.

B. *Throughput Analysis*

Throughput is the effective data transfer rate, measured in bps or bits per second. Throughput parameter testing is used to find out how much bandwidth a communication protocol requires in actual conditions.

Table 1. Throughput Results

No	Data Collection Time		*Throughput*	
			MQTT	HTTP
1	Morning	10 Minute	1715	1717
2		20 Minute	1722	1709
3		30 Minute	1732	1712
4		40 Minute	1735	1711
5		50 Minute	1742	1716
6		60 Minute	1736	1709
Average			**1730**	**1712**
7	Night	10 Minute	1892	1823
8		20 Minute	1901	1835
9		30 Minute	1901	1834
10		40 Minute	1901	1834
11		50 Minute	1900	1830
12		60 Minute	1896	1829
Average			**1899**	**1831**

Table 1 shows the data from MQTT and HTTP throughput testing results. Data collection was carried out for 60 minutes in the morning and night. The unit used for throughput is bits per second.

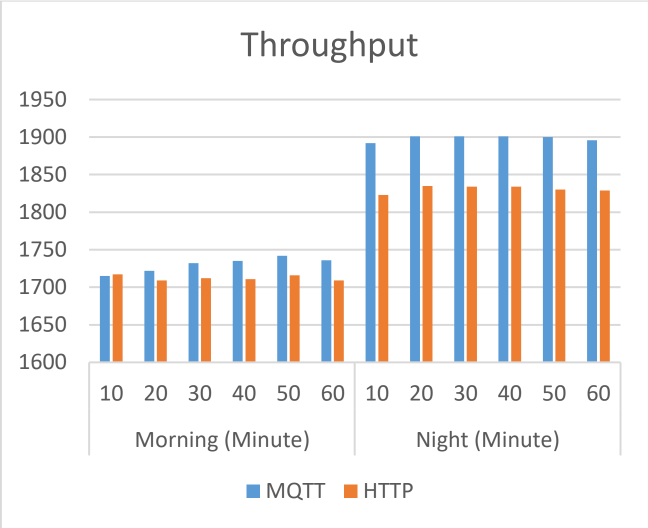

Fig.10. Throughput Graph

Based on Fig.10, it can be concluded that the throughput generated by the MQTT protocol has a better value when compared to the throughput generated by the HTTP protocol, with an average MQTT throughput of 1730 in the morning and 1899 in the night. Meanwhile, the HTTP protocol has an average throughput of 1712 in the morning and 1831 in the night. This shows that the MQTT protocol can handle more transactions or packets more efficiently and faster compared to HTTP.

C. Jitter Analysis

Jitter is a change in latency from a delay or variation in packet arrival time. Jitter can result in packet loss, especially when sending packets at high speeds.

Table 2. Jitter Results

No	Data Collection Time		*Jitter*	
			MQTT	HTTP
1	Morning	10 Minute	0,094953	0,002229
2		20 Minute	0,033675	0,081063
3		30 Minute	0,028766	0,004074
4		40 Minute	0,012703	0,072771
5		50 Minute	0,092997	0,080121
6		60 Minute	0,010212	0,029483
Average			**0,045551**	**0,044957**
7	Night	10 Minute	0,087981	0,018712
8		20 Minute	0,028353	0,035648
9		30 Minute	0,021988	0,007661
10		40 Minute	0,015424	0,013756
11		50 Minute	0,010541	0,005821
12		60 Minute	0,005314	0,007498
Average			**0,028267**	**0,014849**

Table 2 shows the data from the MQTT and HTTP jitter testing results. Data collection was carried out for 60 minutes in the morning and night. The unit used for jitter is milliseconds.

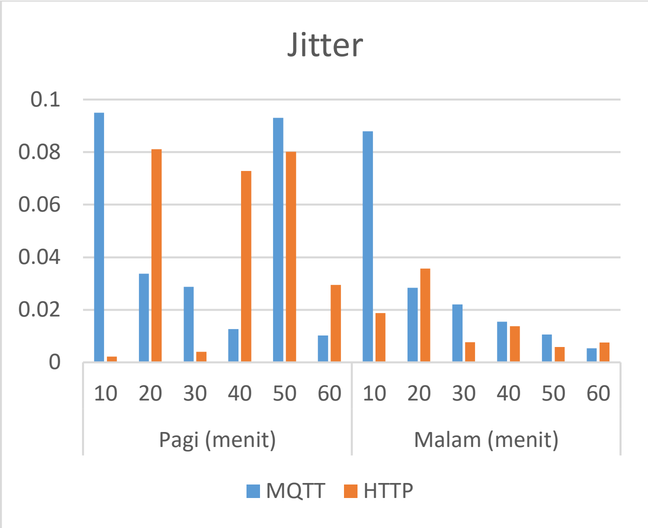

Fig.11. Jitter Graph

Based on Fig.11, it can be concluded that the jitter produced by the HTTP protocol has a smaller value compared to the jitter produced by the MQTT protocol, with an average HTTP jitter of 0.044957 in the morning and 0.014849 in the night. Meanwhile, the MQTT protocol has an average jitter of 0.045551 in the morning and 0.028267 in the night. This shows that the HTTP protocol has a more stable response time compared to the MQTT protocol.

D. *Packet Loss Analysis*

Packer loss parameter testing is used to determine the number of lost or damaged packets in network traffic.speeds.

Table 3. Packet Loss Results

No	Data Collection Time		MQTT		HTTP	
			Total Packet	*Packet Loss*	*Total Packet*	*Packet Loss*
1	Morning (Minute)	10	635	16 (0,025 %)	638	18 (0,028 %)
2		20	1277	40 (0,031 %)	1282	30 (0,023 %)
3		30	1928	71 (0,037 %)	1932	49 (0,025 %)
4		40	2575	97 (0,038 %)	2566	76 (0,030 %)
5		50	3233	134 (0,041 %)	3192	99 (0,031 %)
6		60	3866	149 (0,039 %)	3827	106 (0,028 %)
Average				**0,035 %**		**0,028 %**
7	Night (Minute)	10	638	18 (0,046 %)	643	18 (0,012 %)
8		20	1282	40 (0,051 %)	1274	32 (0,019 %)
9		30	1932	65 (0,052 %)	1906	49 (0,019 %)
10		40	2566	76 (0,052 %)	2539	62 (0,020 %)
11		50	3192	93 (0,052 %)	3170	75 (0,018 %)

No	Data Collection Time		MQTT		HTTP	
			Total Packet	*Packet Loss*	*Total Packet*	*Packet Loss*
12		60	3827	108 (0,050 %)	3689	86 (0,019 %)
Average				**0,050 %**		**0,018 %**

Table 3 shows the data from MQTT and HTTP packet loss testing results. Data collection was carried out for 60 minutes in the morning and night. The Total Packet column is the total number of packets sent, while the Packet Loss column is the number of lost or damaged packets from the total packets.

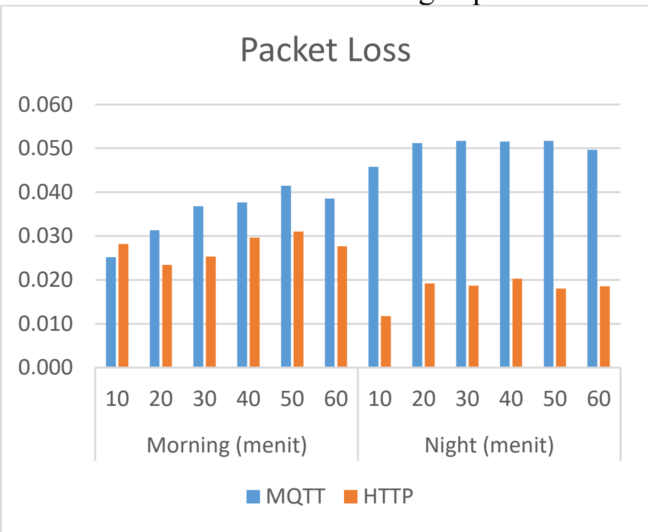

Fig.12. Packet Loss Graph

Based on Fig.12, it can be concluded that the HTTP protocol has a lower number of packet losses compared to the MQTT protocol, with an average HTTP packet loss of 0.028% in the morning and 0.018% in the night. Meanwhile, the MQTT protocol has an average packet loss of 0.035% in the morning and 0.050% in the night. This shows that the HTTP protocol has better packet delivery quality in terms of packet loss parameters.

E. Delay Analysis

Delay parameter testing is carried out to determine how long it takes for a packet to travel from origin to destination.

Table 4. Delay Results

No	Data Collection Time		*Delay*	
			MQTT	*HTTP*
1	Morning	10 Minute	0,943253523	0,938906587
2		20 Minute	0,939724806	0,944096754
3		30 Minute	0,933688113	0,942278125
4		40 Minute	0,932460826	0,943208264
5		50 Minute	0,928541312	0,94150398
6		60 Minute	0,931455780	0,945833544
Average			**0,934854060**	**0,934854060**
7	Night	10 Minute	0,938950766	0,941014098
8		20 Minute	0,936812025	0,943545567
9		30 Minute	0,935863514	0,944596850
10		40 Minute	0,939598921	0,946083264
11		50 Minute	0,940307796	0,946378656
12			0,941219787	0,946612544
Average			**0,855531576**	**0,855531576**

Table IV shows the data from the MQTT and HTTP jitter testing results. Data collection was carried out for 60 minutes in the morning and night. The unit used for jitter is milliseconds.

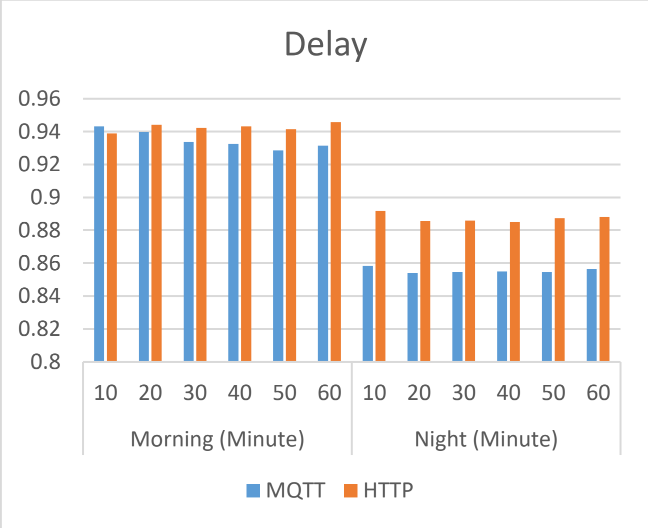

Fig.13. Delay Graph

Based on Fig.13, it can be concluded that the smallest delay is produced by the MQTT protocol with an average delay of 0.93485406 seconds in the morning and 0.855531576 seconds in the night. Meanwhile, the HTTP protocol has an average delay of 0.942637876 seconds in the morning and 0.895162903 seconds in the night. This shows that the speed of sending MQTT data is faster than HTTP.

5. CONCLUSION

Based on the results of testing for 60 minutes in the morning and night, it was found that the MQTT protocol obtained better throughput results compared to the HTTP protocol. This shows that the MQTT protocol is able to handle more transactions or packets more efficiently and quickly. Then, in the jitter parameter, the HTTP protocol has a better jitter value compared to the MQTT protocol; this shows that the HTTP protocol has a more stable response time compared to the MQTT protocol. Furthermore, regarding the packet loss parameter, the HTTP protocol has a smaller number of packet losses compared to the MQTT protocol, which shows that the HTTP protocol has better packet delivery quality. Then the delay parameter of the MQTT protocol has a smaller delay time compared to the HTTP protocol, which shows that the speed of sending MQTT data is faster than that of HTTP.

ACKNOWLEDGMENT

Thank you to the team that has helped complete this research. We hope that this research can continue to be developed and obtain something new that will continue to remain at the top level in the field of IoT communication protocols.

REFERENCES

[1] S. Wahyu, M. Syafaat, and A. Yuliana, “Rancang Bangun Sistem Monitoring Pertumbuhan Tanaman Cabai Menggunakan Arduino Bertenaga Surya Terintegrasi Internet of Things (IoT),” *Jurnal Teknologi*. repository.binawan.ac.id, 2020.

[2] A. Rifa’i, M. U. H. Al Rasyid, and ..., “Sistem Pemantauan Dan Kontrol Otomatis Kualitas Air Berbasis Internet of Things (Iot) Menggunakan Platform Node-Red Untuk Budidaya Udang,” *JTT (Jurnal Teknol. ...*, 2021.

[3] A. Faroqi, “Design of arduino uno based duck egg hatching machine with sensor DHT22 and PIR sensor,” *Proceedings - 2020 6th International Conference on Wireless and Telematics, ICWT 2020*. 2020, doi: 10.1109/ICWT50448.2020.9243640.

[4] J. Kumar, “Automatic plant watering and monitoring system using NodeMCU,” *Proceedings of the 9th International Conference On Cloud Computing, Data Science and Engineering, Confluence 2019*. pp. 545-550, 2019, doi: 10.1109/CONFLUENCE.2019.8776956.

[5] N. Widyaningrum and U. Y. Oktiawati, “Sistem Pemantauan dan Pengendalian Debit Fluida Berbasis Arduino dan Website,” *J. Nas. Tek. Elektro*, 2020.

[6] H. J. J. Ochoa, “Comparative analysis of power consumption between MQTT and HTTP protocols for an IoT platform designed and implemented for remote real-time monitoring of …,” *repositorio.tec.mx*.

[7] N. Naik, “Choice of effective messaging protocols for IoT systems: MQTT, CoAP, AMQP and HTTP,” *2017 IEEE Int. Syst. Eng. …*, 2017.

[8] M. A. Amrullah, K. M. Lhaksmana, and ..., “Pembangunan dan pengujian protokol MQTT & WebSocket untuk Aplikasi IoT Rumah Cerdas berbasis Android,” *eProceedings …*, 2018.

[9] H. Hairatunnisa, H. A. Nugroho, and ..., “Analisis Kinerja Protokol MQTT dan HTTP Pada Akuisisi Data Magnet Berbasis Internet of Things,” *J. Ilm. …*, 2021.

[10] E. Foundation, “IoT Developer Survey 2018,” *Eclipse Foundation*, 2018. https://iot.eclipse.org/community/resources/iot-surveys (accessed Dec. 06, 2022).

[11] B. Wukkadada, “Comparison with HTTP and MQTT in Internet of Things (IoT),” *Proceedings of the International Conference on Inventive Research in Computing Applications, ICIRCA 2018*. pp. 249–253, 2018, doi: 10.1109/ICIRCA.2018.8597401.

[12] C. F. Permatasari and H. Dhika, “Optimasi Jalur Transfer Data dari HTTP menjadi MQTT pada IoT menggunakan Cloud Services,” *JISA (Jurnal Inform. dan Sains)*, 2018.

[13] IBM, “Introduction to MQTT,” *International business machines corporation*, 2022. https://www.ibm.com/docs/en/ibm-mq/7.5?topic=mobile-messaging-m2m (accessed Dec. 06, 2022).

[14] F. Luthfi, E. A. Juanda, and I. Kustiawan, “Optimization of data communication on air control device based on internet of things with application of HTTP and MQTT protocols,” *IOP Conf. Ser.*, 2018.

[15] T. Yokotani, “Comparison with HTTP and MQTT on required network resources for IoT,” *ICCEREC 2016 - International Conference on Control, Electronics, Renewable Energy, and Communications 2016, Conference Proceedings*. pp. 1-6, 2017, doi: 10.1109/ICCEREC.2016.7814989.

[16] A. D. Prakoso, F. T. Syifa, and ..., “Analisis Perbandingan Kualitas Layanan Sistem Antara Protokol HTTP dan MQTT Pada Monitoring Kelembaban Tanah,” *PROtek J. Ilm. …*, 2020.

[17] S. Kalyani, “Measurement and Analysis of QoS Parameters in RPL Network,” *2018 10th International Conference on Advanced Computing, ICoAC 2018*. pp. 307-312, 2018, doi: 10.1109/ICoAC44903.2018.8939052.

[18] V. Andini, L. Sugiyanta, and B. Zaini, “Analisis kinerja parameter throughput dan delay akses inetrnet di smk karyaguna jakarta selatan,” *PINTER J. Pendidik. Tek. …*, 2020.

[19] H. Babbar, “Performance evaluation of QoS metrics in software defined networking using ryu controller,” *IOP Conference Series: Materials Science and Engineering*, vol. 1022, no. 1. 2021, doi: 10.1088/1757-899X/1022/1/012024.

[20] W. Y. Pusvita and Y. Huda, “Analisis kualitas layanan jaringan internet wifi. id menggunakan parameter QoS (Quality of Service),” *Voteteknika (Vocational Tek. Elektron.*, 2019.