# INVESTIGATION AND PERFORMANCE COMPARISON OF IOT COMMUNICATION PROTOCOLS

***D.R.N. Perera[*] and H.M.S.M. Herath***
*Department of Electrotechnology, Wayamba University of Sri Lanka, Sri Lanka*

The Internet of Things (IoT) refers to a collective network of connected devices. Choosing the appropriate communication protocol is a major challenge in finding an effective protocol for various scenarios. Therefore, this paper presents a performance comparison of IoT protocols. In this study, Hypertext Transfer Protocol (HTTP), Message Queuing Telemetry Transport (MQTT), and WebSocket were used. The experimental setup involved the use of an IoT development board and sensors (Ultrasonic, IR, and PIR). The prototype was developed to simulate the process, followed by data collection using monitoring tools. Under the performance evaluation, metrics such as speed, latency, message size, and scalability were assessed. The transmitter and receiver nodes were ESP8266 NodeMCU. Local Area Network was used in this experiment. Five scenarios were investigated as described below. Thirty observations were recorded in each scenario. Results revealed the first and second scenarios of investigating transmission time and scalability in IoT communication protocols under the transmission of constant data when network connections are free and loaded, and in both scenarios, HTTP had the lowest mean at 60.40 ms in connection–free and 68.17 ms in loaded conditions. The third scenario, the investigation of transmission time in IoT communication protocols under the transmission of sensors, generated data that revealed WebSocket had a mean time of 78.39 ms, demonstrating a more consistent performance with a standard deviation of 32.27 ms. The results of the fourth scenario, of investigating transmission time and scalability in IoT communication protocols under transmitting data to multiple recipients, revealed MQTT had a mean time of 76.33 ms. The fifth scenario was of investigating transmission time in IoT communication protocols under gradual message size increments. Its results indicated that WebSocket was the most efficient protocol. The mean value of 688 ms. The network connection was wired (Copper). The Wi-Fi connection was backward due to environmental changes. This study concluded that the varied superiority in speed and effectiveness was exhibited by HTTP, MQTT, and WebSocket under diverse scenarios. This research contributes to the understanding of IoT communication protocol selection according to diverse conditions. Further research will be conducted to compare with other communication protocols such as CoAP, AMQP, and Zigbee to evaluate their performance.

Keywords: ESP8266, Hypertext Transfer Protocol (HTTP), Message Queuing Telemetry Transport (MQTT), WebSocket

[*]*Corresponding author: rajithanirmal2000info@gmail.com*

# INVESTIGATION AND PERFORMANCE COMPARISON OF IOT COMMUNICATION PROTOCOLS

***D.R.N. Perera* and H.M.S.M. Herath***
*Department of Electrotechnology, Wayamba University of Sri Lanka, Sri Lanka*

## INTRODUCTION

The investigation and performance comparison of IoT communication protocols involves evaluating how different protocols manage data transmission between devices (Moraes *et al.,* 2019). These protocols, such as HTTP (Hypertext Transfer Protocol), MQTT (Message Queuing Telemetry Transport), and WebSocket have varying efficiency and speed reliability. This analysis contributes to optimising IoT systems for better connectivity and stability. HTTP, MQTT, and WebSocket demonstrate valid application scenarios such as web browsing, and smart home and real-time chat applications.

HTTP operates on a client-server model, and communication between client and server is connection-oriented (Daud *et al.,* 2017). HTTP is a TCP/IP-based protocol that is used to deliver data on the World Wide Web with port 80 as the default port. MQTT is a messaging protocol that employs the publish-subscribe model to facilitate message transport between servers and clients over TCP/IP (Oliveira *et al.,* 2018). MQTT is particularly well-suited for machine-to-machine communication. A server known as the broker handles message exchanges between clients. The broker filters messages and distributes them based on topics, which are unique identifiers assigned to each message (Yakotani *et al.,* 2021). Those who send messages to the broker with specific topics are called publishers, while those who subscribe to one or more topics to receive specific messages are known as subscribers. WebSocket is a protocol that operates on a client-server model and establishes a connection-oriented communication channel between the client and server (Oliveira *et al.,* 2018), providing full-duplex channels over a single TCP connection. This investigation focusses on the transmission time of different IoT communication protocols at the home-scale level. It is especially relevant for smart home and greenhouse applications, where small data packets are commonly used**.**

## METHODOLOGY

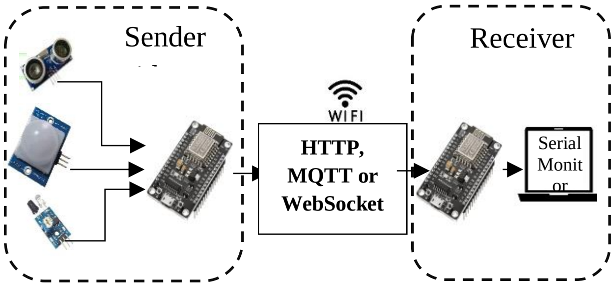


Figure 1: Diagram of transmission of data between sender and receiver by using sensors

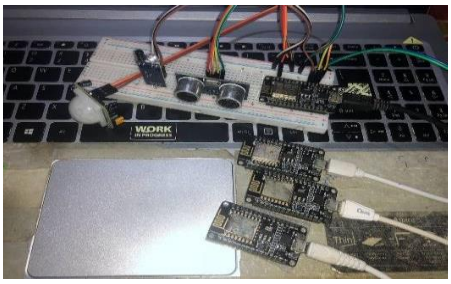

Figure 2: Experiment setup to obtain data

Experimental tests were conducted to measure the performance metrics such as the speed, latency, message size, and scalability of various IoT communication protocols. Figures 1 and 2 indicate the experimental setup. For this, the mosquitto-2.0.18a broker was installed on an Intel Core i5-1035G1 CPU @ 1.00 GHz laptop (Oliveira *et al.,* 2018). Communication was facilitated via a Prolink PRS1140 Wi-Fi router, with ESP8266 NodeMCUs serving as both the transmitter and receiver nodes. Transmission times were measured using the Arduino IDE 2.0.3 serial monitor, ensuring reliable data collection.

The *ESP8266WiFi* library was used across all protocols with *PubSubClient* for MQTT, *ESP8266WebServer* and *ESP8266HTTPClient* for HTTP, and *WebSocketsServer* and

*WebSocketsClient* for WebSocket. The protocols assessed included HTTP, MQTT, and WebSocket, which incorporates an IoT development board and sensors (Ultrasonic sensor, IR sensor, and PIR sensors) to simulate real-world conditions. Three sensors transmit a digital signal to the ESP8266 sender Module that transmits the data to another ESP8266 Module (receiver) via a string type of data.

Five different scenarios were investigated to study the parameters of the aforementioned performance metrics: the transmission of constant datasets in a loaded WiFi connection (achieved by accessing social media, and downloading and uploading were performed) and free network connections, sensor-generated datasets using three types of sensors, data to multiple receivers, and progressively increasing message size. In the multiple-receiver data transmission scenario, three ESP8266 nodes were used as receivers while the transmission time calculation was done by the time difference between the start time of the transmission and the ESP826 node where the receiver last received the message. Throughout these experiments, 30 observations were recorded for each scenario. Statistical analysis of the collected data was done using the Minitab 21 version.

## RESULTS AND DISCUSSION

A. Table 1: Investigation of transmission time in IoT communication protocols under the transmission of constant data when network connections are free and in network connections loaded conditions.

| Connection status | Protocol | Transmission time in milliseconds (ms) | | | | |
|---|---|---|---|---|---|---|
| | | Mean | St.dev | Min | Median | Max |
| Wi-Fi Network free | HTTP | 60.40 | 19.22 | 31.00 | 64.50 | 90.00 |
| | MQTT | 91.60 | 27.62 | 46.00 | 91.50 | 140.00 |
| | WebSocket | 71.90 | 21.34 | 33.00 | 77.00 | 101.00 |
| Wi-Fi Network loaded | HTTP | 68.17 | 21.01 | 38.00 | 68.00 | 110.00 |
| | MQTT | 96.43 | 19.19 | 61.00 | 92.00 | 138.00 |
| | WebSocket | 79.30 | 28.61 | 35.00 | 78.00 | 139.00 |

In this experiment, the parameters of speed, latency, and scalability were investigated under two different conditions: Wi-Fi network free and Wi-Fi network-loaded. Thirty observations were recorded. Since the same amount of data was transmitted (45 bytes) in both cases, the effects on speed and latency were comprehensively analysed. The lowest mean value indicated the highest efficiency of the IoT communication protocol. Specifically in both conditions, HTTP had the lowest mean transmission time of 60.40 ms in connection-free conditions and 68.17 ms in connection-loaded conditions. The results are shown in Table 1. This indicates that HTTP outperforms MQTT and WebSocket in terms of transmission efficiency and good scalability under the tested conditions. The measurement of scalability was the principal parameter in this performance investigation. The results revealed the performance of transmitting the data via three communication protocols while the network connection was loaded.

B. Table 2: Investigation of transmission time in IoT communication protocols under the transmission of sensors generated data

| Variable | Mean | St.dev | Min | Median | Max |
|---|---|---|---|---|---|
| HTTP | 84.07 | 22.90 | 35.00 | 80.00 | 159.00 |
| MQTT | 97.75 | 43.99 | 24.00 | 93.50 | 203.00 |
| WebSocket | 78.39 | 32.27 | 29.00 | 78.00 | 130.00 |

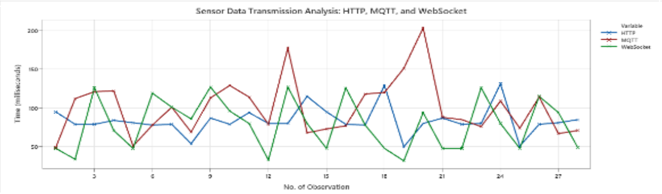


Figure 3: Sensor data transmission analysis HTTP MQTT and WebSocket

The basic parameters of speed and latency were also investigated in this experiment using a specific case involving the continuous transmission of data generated by three sensors (Ultrasonic sensor, IR sensor, and PIR sensors) through the three protocols. Thirtyobservations were recorded. The data size varied between 58 bytes and 68 bytes. WebSocket demonstrated the highest efficiency compared to MQTT and HTTP protocol. Specifically, WebSocket had a mean transmission time of 78.39 ms, demonstrating a more consistent performance with a standard deviation of 32.27 ms. The results of the investigation of the speed and latency, by transmitting varying amounts of data via three IoT communication protocols, are shown in Table 2 and Figure 3.

C. Table 3: Investigation of transmission time in IoT communication protocols under transmitting data to multiple recipients

| Variable | Mean | St.dev | Min | Median | Max |
|---|---|---|---|---|---|
| HTTP | 545.3 | 241.6 | 179.0 | 578.0 | 924.00 |
| MQTT | 76.33 | 29.38 | 32.00 | 82.50 | 131.00 |
| WebSocket | 79.07 | 36.61 | 31.00 | 78.00 | 189.00 |

This experiment also investigated the scalability of protocols by increasing the workload of the data sender (increasing the number of receivers). Thirty observations were recorded. MQTT exhibited the highest efficiency, better than WebSocket and HTTP. MQTT had a mean transmission time of 76.33 ms. The results are shown in Table 3. This indicates that MQTT performs at the highest efficiency when transmitting data with various communication protocols. HTTP showed the highest mean value numerically. The delay in data transmission over HTTP is due to the separate transmission by the sender to the receiver's IP addresses.

D. Table 4: Investigation of transmission time in IoT communication protocols under gradual message size increments

4.1 Transmission of data using HTTP

| Message size (bytes) | Mean | St.dev | Min | Median | Max |
|---|---|---|---|---|---|
| 10 | 79.27 | 28.49 | 36.00 | 74.00 | 132.00 |
| 500 | 252.5 | 103.1 | 118.0 | 315.0 | 371.0 |
| 5000 | 535.5 | 136.5 | 446.0 | 480.0 | 894.0 |
| 10000 | 2168 | 966 | 1024 | 2981 | 3021 |
| 12000 | 2692 | 977 | 1171 | 2897 | 3607 |
| Overall mean | 1145.4 | | | | |

4.2 Transmission of data using MQTT

| Message size (bytes) | Mean | St. dev | Min | Median | Max |
|---|---|---|---|---|---|
| 10 | 89.3 | 51.9 | 41.0 | 76.0 | 209.0 |
| 50 | 83.7 | 37.2 | 40.0 | 95.0 | 154.0 |
| 100 | 97.5 | 51.4 | 46.0 | 78.0 | 222.0 |

4.3 Transmission of data using WebSocket

| Message size (bytes) | Mean | St.dev | Min | Median | Max |
|---|---|---|---|---|---|
| 10 | 74.3 | 37.5 | 31.0 | 76.0 | 161.0 |
| 500 | 104.1 | 36.2 | 33.0 | 117.0 | 145.0 |
| 5000 | 531.9 | 85.0 | 449.0 | 493.0 | 722.0 |

| | | | | | |
|---|---|---|---|---|---|
| 10000 | 1003.5 | 127.5 | 900.0 | 952.0 | 1323.0 |
| 12000 | 1726 | 778 | 1098 | 1171 | 3210 |
| Overall mean | 688 | | | | |

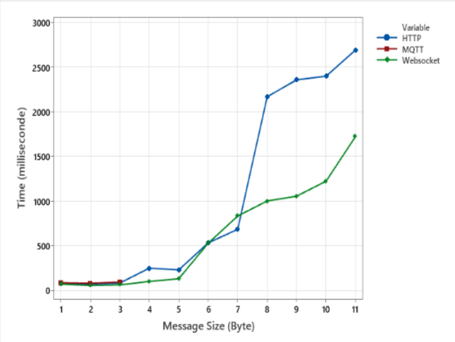


Figure 4: Data transmission across gradually increase message sizes

This experiment investigated the speed, transmission time, and scalability, it was demonstrated that the amount of message size transmitted gradually increased. This study analysed the efficiency of IoT communication protocols such as HTTP, MQTT, and WebSocket by varying message sizes. Thirty observations were recorded. Results indicated that WebSocket was the most efficient protocol. The overall mean value of the transmission time of HTTP is 1145.4 ms. WebSocket exhibited an overall mean value of 688 ms. According to the mean values, WebSocket expressed higher efficiency when compared with HTTP. The results are shown in Table 4 and Figure 4. WebSocket successfully transmitted larger messages, handling over 12,000 bytes without losses, whereas HTTP-encountered messages randomly declined at this size. When comparing the mean transmission efficiency, WebSocket outperformed HTTP. However, when sending smaller data packets (100 bytes), all three protocols performed well while for larger packets (500 bytes), the MQTT broker failed to respond, indicating a limitation. These observations were made while transmitting data between two ESP8266 modules. The results revealed that WebSocket showed high efficiency when compared with the other two IoT communication protocols.

## CONCLUSIONS

This research contributes to the understanding of the efficiency of IoT communication protocol selection according to diverse conditions. The research findings show that HTTP, MQTT, and WebSocket demonstrate different efficiencies for different scenarios. Table 1 demonstrates HTTP's ability to send a constant amount of data rapidly within low latency and when the Wi-Fi network is loaded (scalability when there is a growing amount of work in the network). HTTP is the most effective over other protocols. Table 2 shows that the WebSocket protocol is more suitable than other protocols for managing data transmission with constantly changing data sizes. MQTT's scalability is demonstrated by its ability to efficiently handle increasing receivers via its publish/subscribe model, maintaining low latency. Table 3 shows that the choice of MQTT protocol is more efficient when the sender sends data to more users. When quite large amounts of data have to be transmitted, the WebSocket protocol can achieve very good performances because unlike HTTP, which requires additional headers and a new connection for each set of data (request/response cycle), WebSocket maintains a single, persistent connection between the client and server, reducing the need for repeated handshakes. This allows for continuous and efficient larger data transmission. This is demonstrated in Table 4. In this research, the behavior of the above-mentioned protocols were analysed in these parameters of speed, delay, message size, and scalability, and their performance was evaluated. This research contributes to better understandings when selecting a suitable protocol in specific cases.

## REFERENCES

Daud, M. A., & Suhaili, W. S. H. (2017). Internet of Things (IoT) with CoAP and HTTP protocol: A study on which protocol suits IoT in terms of performance. In Computational Intelligence in Information Systems: Proceedings of the Computational Intelligence in Information Systems Conference (CIIS 2016) (pp. 165-174). Springer International Publishing

Moraes, T., Nogueira, B., Lira, V., & Tavares, E. (2019, October). Performance comparison of IoT communication protocols. In 2019 IEEE International Conference on Systems, Man, and Cybernetics (SMC) (pp. 3249-3254). IEEE.

Oliveira, G. M., Costa, D. C., Cavalcanti, R. J., Oliveira, J. P., Silva, D. R., Nogueira, M. B., & Rodrigues, M. C. (2018, April). Comparison between MQTT and WebSocket protocols for IoT applications using ESP8266. In 2018 Workshop on Metrology for Industry 4.0 and IoT (pp. 236-241). IEEE.

Yakotani, T., Ohno,S., Mukai, H., & Ishibashi, K. (2021). IoT platform with distributed brokers on MQTT. International Journal of Future Computer and Communication, 10 (1), 7-12.