# Latency evaluation for MQTT and WebSocket Protocols: an Industry 4.0 perspective

Diego R. C. Silva, Guilherme M. B. Oliveira, Ivanovitch Silva
Federal University of Rio Grande do Norte
diego@ect.ufrn.br, guimatheus@ufrn.edu.br, ivan@imd.ufrn.br

Paolo Ferrari, Emiliano Sisinni
University of Brescia
paolo.ferrari@unibs.it, emiliano.sisinni@unibs.it

***Abstract*— Internet of things (IoT) is a trend which consists of providing connectivity for many consumer devices. Consequently, a considerable number of applications never seen before is emerging to make daily lifestyle more practical and connected through sharing, storing and processing personal data. When applied to industry, the same technologies can cause a real revolution in the manner how the information is treated. The value of the processed data is increasing fast and point to the fourth industrial revolution, or industry 4.0. This paper evaluates the performance of two ICT (Information and Communication Technology) protocols coming from the consumer world with an enormous potential to be used by industry. Experiments were performed to measure the round trip time using MQTT (Message Queuing Telemetry Transport) and WebSocket protocols with the exchange of data between one server in Italy and another in Brazil. The results can be analyzed for each possible application and different network paths. The average latency samples using each of the protocols were similar, defining the payload size of the requests, but changing according to the way.**

## I. INTRODUCTION

The industry world is changing. In the past, the industry was one of the big promoters of new technologies, new approaches and new ideas [1]. Today, the Industry has been absorbing the advances of other sectors, primarily from emerging ICT technologies and general consumer market. The reason is the fast-evolving situation and the opening of new markets, which made the slow to change industrial applications acting as trend followers in the last years. Now, the Industry 4.0 concepts are trying to harmonize the new tendencies, making the industry world again at the edge of technologies, giving back its leading role. The ICT solutions are seen as tools the new Industry 4.0 applications can exploit to get the grip again on the market [2]. Cloud computing, big data, data analytics, machine learning, and Internet of things are the backbone on which the new industry is growing. The low ROI forces to build new architecture using high-level API, and data-driven services are arising due to the requested scalability of devices.

For years, the industrial world had its own, dedicated communication protocols, designed with the only aim to fulfill the hard real-time requirements of the physical world things that industrial process need to create, control, and modify [3]. Today, the concept of proprietary protocols is declining and there are several attempts to elect existing ICT solution to be the standard method to guarantee reliable and timely communication in Industry 4.0 applications. The aim is twofold: the proposal must be able widely accepted by both industry and consumer market, and it must have a stable and short delivery latency to be used in real-time systems. In particular, the communication delay will soon become one of the key performance indicators (KPIs) to decide to cost/benefit tradeoff of future services in industry 4.0 [4].

Among others, MQTT offers the interesting possibilities of the third party broker that can provide additional information even if client and subscribers leave the network [5]. On the other hand, WebSocket technology is hugely diffused and can cope with the most stringent security rules [6].

In this paper, the performance of two typical ICT IoT consumer application protocols, MQTT and WebSocket, will be evaluated by means of a suitable experimental setup with intercontinental extent. In details, their performance will be validated against the critical latency requirements of Industry 4.0 for both supervisory tasks and slow control tasks.

The paper is structured as in the following. In section II a quick overview of Industry 4.0 is given. In section III, the MQTT and WebSocket technologies are presented with a special attention to the industry related features. In section 4 the experimental methodology is described and, in Section V, the results of a worldwide application scenario are presented. Finally, Section VI concludes the paper and indicate future studies.

## II. INDUSTRY 4.0 AND LATENCY CONCERNS

The Industry 4.0 vision was originally introduced in 2011 in Germany. In such situation, the different production steps and the final products are continuously exchanging data with the cloud, thanks to ubiquitous and pervasive communication [7]. This enables the creation of a digital twin of the real-world product, including a copy of all the product history from the design to the everyday use. On the other hand, the IIoT covers only the operational phase of the product lifecycle, and IIoT may improve the asset, the process, and the efficiency. As a side effect, the new architecture of industrial automation is flattened because the field devices are now able to vertically provide information to any level [8].

It is important to say that the final aim of IIoT is not to substitute the traditional automation applications. The IIoT today is not dealing with control loops at field level, because actual industrial controllers can efficiently work with automation and deterministic reaction time (i.e., bounded and known). Indeed, typical IIoT applications are for Cloud Manufacturing (CM), that is distributed supervision, optimization, and prediction. However, in the CM the man is progressively moved out from the supervisory loop and replaced by a direct machine to machine dialog, hence the availability on geographic scale of short latency (reliable) connections can increase the revenue [9]. This scenario is precisely the main topic of the paper, the investigation about the latency performance of Industry 4.0 services based on standard IoT protocols.

## III. ICT IoT Application Protocols

### A. *WebSocket*

WebSocket protocol was developed to meet constant data exchange between client and server not supported earlier from HTTP [10]. The protocol consists in a complete bidirectional communication channel that works through a single socket [11], as well as having an asynchronous communication (in contrast with HTTP protocol), in other words, both sides can send data anytime while the connection is established.

The WebSocket functioning is divided into two parts: handshake and data transfer. On the handshake, the client and the server establish initial communication using HTTP. On this first communication, the client requests a communication type update. Once verified, the request is validated, and the data exchange can be done using the WebSocket protocol (Fig. 1).

For communication, disregarding IP, TCP, and TLS framing overhead, a single HTTP request could carry an additional 500-800 bytes of metadata plus cookies. In contrast, WebSocket protocol uses a custom binary framing format that divides each message into one or more frames. When these frames reach a destination, they are joined, and the sender is notified that the entire message has been received. Each frame header can have 2 to 10 bytes in size if sent by the server and 6 to 14 bytes if sent by the client (a client must add a masking key to prevent cache poisoning attacks). WebSocket is also considered one of the most versatile data transport methods available, because of the customization capabilities through Application Programming Interfaces (API's), extensions and sub-protocols. Features like compression and multiplexing can be seen in [12].

Figure 1 shows the information flow of HTTP and WebSocket. On the first case, data from the server is only sent in response to a client requisition. The client always initiates the communication. Many applications in consumer and industry context depend on the ability to communicate data or commands to the client, and although HTTP is not the ideal protocol for this kind of task, there are techniques, like pooling, used to overcome this limitation.

WebSocket, however, is appropriate for bidirectional applications, allowing a completely asynchronous communication channel, making possible a full range of functionalities.

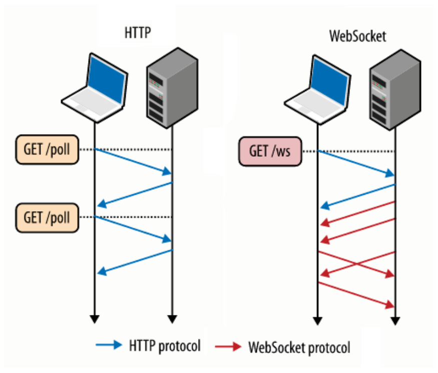

Fig. 1. HTTP and WebSocket protocols communication flow, source: hpbn.co (modified)

### B. *MQTT*

MQTT is an exchange messaging protocol that uses the publish-subscribe standard for transporting messages between a server and clients. It runs over TCP/IP and can be run on network protocols that provide ordered, lossless and bidirectional connections. It is standardized by a technical committee of the Organization for the Advancement of Structured Information Standards. MQTT becomes a good candidate to be used for Machine-to-Machine communication and IoT contexts, designed to be lightweight, open and easy to implement, especially in contexts where the internet can be expensive, has low bandwidth, is not secure or when utilizing an embedded device with limited memory resources or processing [13].

In publish-subscribe pattern used at the MQTT, the messages exchanged between different clients is through of a server, called the broker. The broker filters the messages and distributes them to the clients according to the topic - an identifier contained in each message. The client can be an IoT device, Web application, mobile application among others. Those who publish a message to the broker with a topic are called publishers, and those who subscribe one or more topics for reading specific messages are called subscribers. The subscribers can receive messages from some publishers and can send them to others subscribers, and a client can be both publisher and subscriber. All the clients establish the connection with the broker. The publishers do not know the destination of messages sent, and the subscribers do not know the origin of messages received [14]. An example of architecture using this pattern is shown in Fig. 2.

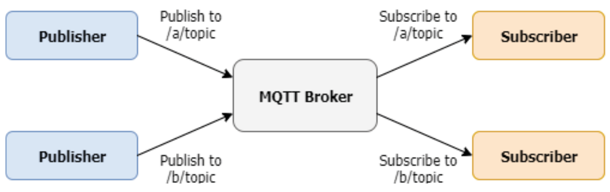

Fig. 2. Publish/subscribe architecture example

The format for a control packet of a message using the protocol is divided between fixed header, variable header, and payload. The fixed header has the size of two bytes, and the variable header and payload size can range from zero to N bytes.

MQTT has three Quality of Services (QoS) levels to message delivery:

- QoS 0 (at most once delivery): messages are delivered at most once, either party does not store it and there is no acknowledgment of delivery on the network.
- QoS 1 (at least once delivery): messages must be delivered at least once, the sender stores and tries to send a message until it receives a confirmation so that the receiver can receive and process a message several times.
- QoS 2 (exactly once delivery): messages are delivered exactly once, the message is stored in the sender and receiver until both parties receive confirmation that the message has been delivered exactly once to the receiver.

The levels of QoS can be used according to requirements of applications and have different times for delivery [15].

## IV. Experiment Methodology

Experimental tests were performed to obtain the latency of packets transferred between two machines. One located in Italy and another in Brazil, using both MQTT and WebSocket protocols. Considering the client-server model, the tests were performed with the machine in Italy acting as the server and the machine in Brazil as the client and vice versa (see Fig. 3). It is not possible to specify the server hardware in Italy and Brazil, both of which are in rack structures and share resources with other virtual machines.

For precision in time measurements, it was necessary to synchronize the server timestamps. The Network Time Protocol (NTP) was used for this purpose. The two servers have been synchronized to UTC reference by means of GPS receivers with NTP timeserver. There are two GPS receivers one in Italy and the other in Brazil. The NTP time server is connected in the same LAN of the server in order to minimize delay and obtain the best performance. In the current experimental setup, the average residual time offset with respect to UTC is always less than 1 ms, as shown in similar setup described in [16]. The short term stability of the local oscillators in the GPS receivers and in the servers are not relevant due to the short duration of each experimental run (less than 30 s).

The tests with both protocols follow the same steps. Before the tests are performed, the payload size, number of samples, connection ports, quality of service (for MQTT) are set. After the client and server are connected, the client sends a message to the server with a given payload and stores a T1 send time. When the server receives the message, it stores a T2 arrival time and responds with a message that has the same payload and stores a T3 send time. When the client receives the response, it saves an arrival time T4. This process is repeated until the number of samples is reached. When sampling ends, the server sends the stored T2 and T3 times to the client. At the client, a file with the data of all times computed to be analyzed is saved.

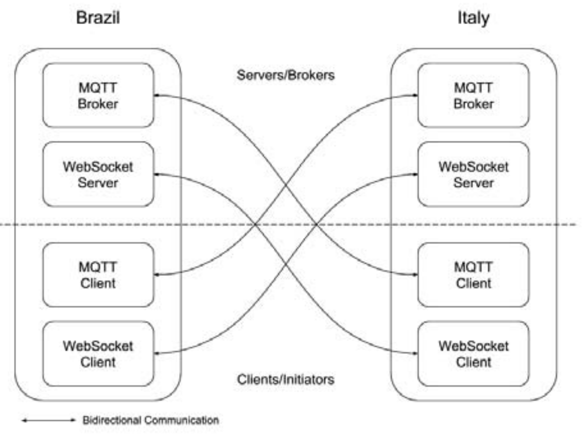

Fig. 3. Bidirectional intercontinental communication using MQTT and WebSocket.

Message exchange has different standards for each of the tested protocols. This implies differences in the server and client architecture. Regarding the MQTT protocol, the client subscribes to the ”/g” topic when connecting to the broker. The client sends messages through the ”/p” topic. The broker sends messages with the same payload to the client in the ”/g” topic. The use of different topics was the strategy chosen to differentiate the data coming from the client or internal communication in the broker used. Using WebSocket protocol, there is no need for specific routes, topics, or events to exchange messages. When the client connects, it sends a message with the predefined payload and the server responds to a message with the same payload. That message exchange occurs for each sample, for statistical reasons.

The payload sizes of messages exchanged between the computers in the tests were 5, 10, 50, 100, 500, and 1000 bytes. A thousand samples with T1, T2, T3 and T4 times were obtained for each payload size using both protocols, MQTT and WebSocket. By obtaining the values of the times, it is possible to establish:

$$T4 - T1 = \text{round-trip time}$$
$$T2 - T1 = \text{one-way}$$
$$T4 - T3 = \text{return time}$$

The algorithms to obtain the times were written in Javascript using the platform Node.js. For the WebSocket protocol, the ’ws’ library was used for both the server and the client. For MQTT was used of the library ’MQTT.js’ for the client and broker ’Mosca’ for the server. For tests, the following configuration is used for MQTT:

- Quality of Service = 0
- Retain = false
- No data persistence
- No authentication

Using WebSocket, the data compression extension (permessage-deflate extension) is enabled on the server. The optional ”bufferutil” and ”utf-8-validate” modules are also enabled to improve performance and compliance with the specification.

## V. Results

The results will be shown separated by protocol, measurement type (one-way, return time or round trip) and initiator (Brazil or Italy). Box plots shown below have no outliers and the orange lines in the graphs are the medians.

### A. *Brazil-Italy-Brazil - WebSocket*

Fig 4 shows the box plots for tests initiated in Brazil, considering round trip times. In other words, the time to the packages go to Italy and go back to Brazil using WebSocket protocol.

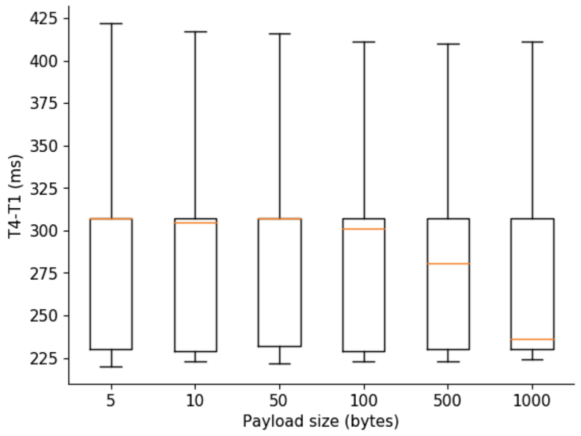

Fig. 4. Brazil-Italy-Brazil - Round Trip

The same data were used to fill out Table I. As can be seen, the means for messages with all payload sizes were between 267 and 287 milliseconds. The mean times are much closer to the minimum time than the maximum time, and all standard deviations are consistently between 40 and 65 milliseconds. In this case, can be assumed that at least three packages with different payload sizes (limited to 1000 bytes) can be sent and received back with particular responses in less than one second. This insight can be used to guide the feasibility of possible applications for industry 4.0.

TABLE I
Brazil-Italy-Brazil - WebSocket - Round Trip

| Payload size | T4 - T1 | | | | |
|---|---|---|---|---|---|
| | Mean | Std. Deviation | Median | Min. | Max. |
| 5 | 283.35 | 53.52 | 307.00 | 220.00 | 1331.00 |
| 10 | 280.02 | 61.34 | 305.00 | 223.00 | 1221.00 |
| 50 | 286.62 | 53.41 | 307.00 | 222.00 | 1026.00 |
| 100 | 279.05 | 64.51 | 301.00 | 223.00 | 1092.00 |
| 500 | 270.93 | 40.85 | 280.50 | 223.00 | 472.00 |
| 1000 | 267.43 | 55.63 | 236.00 | 224.00 | 1349.00 |

Fig. 5 as well as Table II shows the results for one-way time (T2-T1) for connections initiated in Brazil. In this case, can be observed that the payload size had influenced on delivery time. The bigger is the payload, bigger is the time measurements.

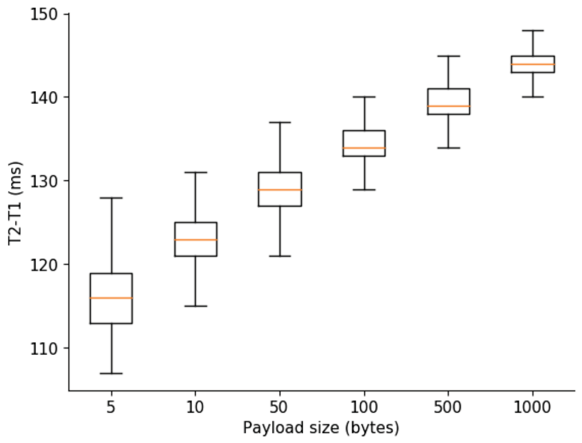

Fig. 5. Brazil-Italy-Brazil - One way

TABLE II
Brazil-Italy-Brazil - WebSocket - One way

| Payload size | T2 - T1 | | | | |
|---|---|---|---|---|---|
| | Mean | Std. Deviation | Median | Min. | Max. |
| 5 | 117.77 | 34.68 | 116.00 | 107.00 | 1178.00 |
| 10 | 124.95 | 34.61 | 123.00 | 114.00 | 1112.00 |
| 50 | 130.54 | 19.30 | 129.00 | 121.00 | 683.00 |
| 100 | 136.27 | 31.51 | 134.00 | 126.00 | 992.00 |
| 500 | 140.29 | 8.22 | 139.00 | 133.00 | 328.00 |
| 1000 | 145.84 | 39.82 | 144.00 | 135.00 | 1260.00 |

### B. *Brazil-Italy-Brazil - MQTT*

Fig. 6 and Table III show the results for MQTT round trip times. When compared to WebSocket we can note many similarities. In the general case, the means are very close to each other, standard deviations are equivalent as well as minimum and maximum times. The differences are minimal and due to their experimental nature.

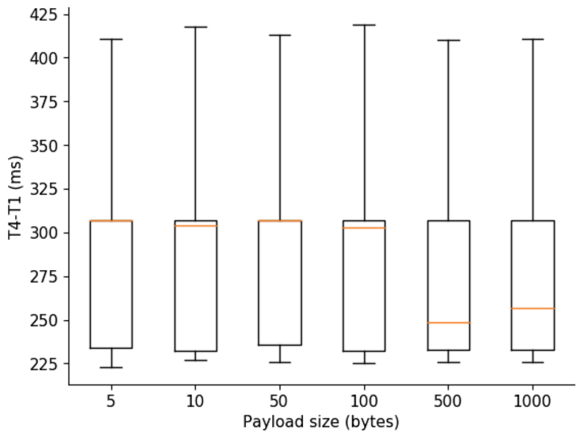

Fig. 6. Brazil-Italy-Brazil - Round Trip

TABLE III
BRAZIL-ITALY-BRAZIL - MQTT - ROUND TRIP

| Payload size | T4 - T1 | | | | |
|---|---|---|---|---|---|
| | Mean | Std. Deviation | Median | Min. | Max. |
| 5 | 284.70 | 52.51 | 307.00 | 223.00 | 1331.00 |
| 10 | 279.78 | 61.62 | 304.00 | 227.00 | 1590.00 |
| 50 | 289.57 | 80.23 | 307.00 | 226.00 | 1757.00 |
| 100 | 282.01 | 67.09 | 303.00 | 225.00 | 1217.00 |
| 500 | 270.77 | 44.65 | 248.50 | 226.00 | 798.00 |
| 1000 | 270.73 | 39.70 | 256.50 | 226.00 | 686.00 |

Fig. 7 and Table IV show the results for one way times ($T2 - T1$).

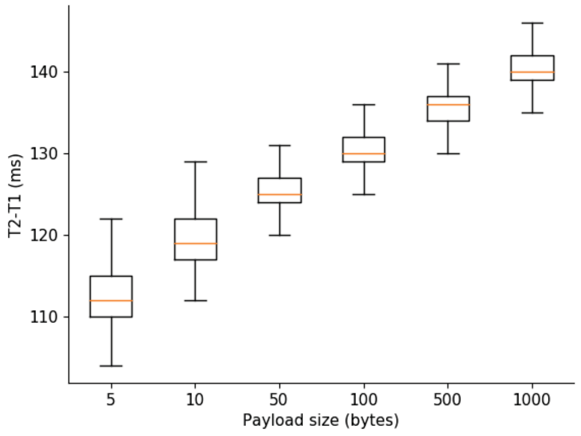

Fig. 7. Brazil-Italy-Brazil - One way

TABLE IV
BRAZIL-ITALY-BRAZIL - MQTT - ONE WAY

| Payload size | T2 - T1 | | | | |
|---|---|---|---|---|---|
| | Mean | Std. Deviation | Median | Min. | Max. |
| 5 | 114.30 | 34.75 | 112.00 | 100.00 | 1178.00 |
| 10 | 119.97 | 5.98 | 119.00 | 112.00 | 237.00 |
| 50 | 126.52 | 6.63 | 125.00 | 118.00 | 206.00 |
| 100 | 133.09 | 43.66 | 130.00 | 120.00 | 1115.00 |
| 500 | 136.79 | 18.57 | 136.00 | 129.00 | 703.00 |
| 1000 | 140.87 | 14.68 | 140.00 | 133.00 | 594.00 |

### C. Italy-Brazil-Italy - WebSocket

Fig. 8 and Table V show the results for WebSocket protocol with connection initiated in Italy. The mean times are all equivalent with the same measurements but opposite way. The biggest difference is on its variations. The times for different payload sizes are almost the same. Standard deviations are also much smaller and minimum and maximum times are more predictable.

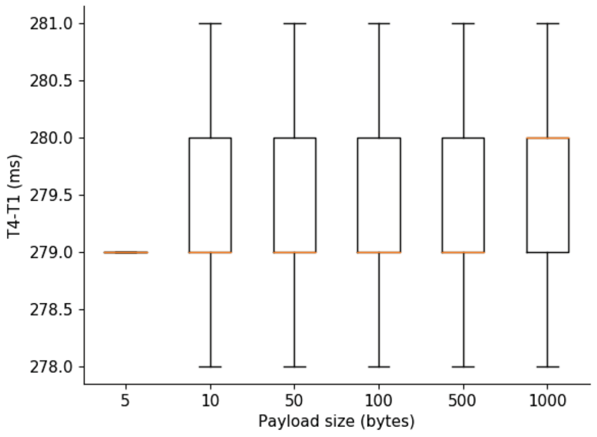

Fig. 8. Italy-Brazil-Italy - Round Trip

These results show a more consistent performance when the packages come from Italy. Consequently, we can associate bigger standard deviations on connections originated in Brazil not to the protocols, but to the network issues. These good results show that consistent and reliable delivering times can be reached and Industry 4.0 applications can make use of them.

TABLE V
ITALY-BRAZIL-ITALY - WEBSOCKET - ROUND TRIP

| Payload size | T4 - T1 | | | | |
|---|---|---|---|---|---|
| | Mean | Std. Deviation | Median | Min. | Max. |
| 5 | 280.02 | 15.84 | 279.00 | 278.00 | 775.00 |
| 10 | 280.16 | 3.89 | 279.00 | 278.00 | 335.00 |
| 50 | 280.15 | 4.98 | 279.00 | 278.00 | 337.00 |
| 100 | 279.39 | 1.98 | 279.00 | 278.00 | 321.00 |
| 500 | 280.13 | 3.16 | 279.00 | 278.00 | 312.00 |
| 1000 | 279.93 | 1.44 | 280.00 | 278.00 | 307.00 |

Fig. 9 and Table II show the results for one way ($T2-T1$) delivery times using WebSockets. As the round trip times, these ones are very close to each other, showing only small variations as consequence of consistent connections.

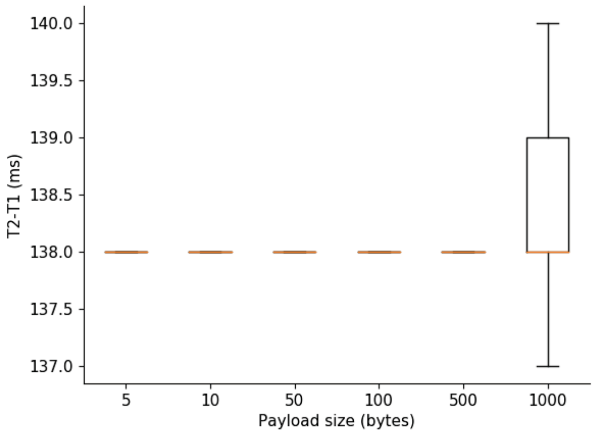

Fig. 9. Italy-Brazil-Italy - One way

TABLE VI
ITALY-BRAZIL-ITALY - WEBSOCKET - ONE WAY

| Payload size | T2 - T1 | | | | |
|---|---|---|---|---|---|
| | Mean | Std. Deviation | Median | Min. | Max. |
| 5 | 138.24 | 2.03 | 138.00 | 137.00 | 167.00 |
| 10 | 138.67 | 3.46 | 138.00 | 137.00 | 194.00 |
| 50 | 138.73 | 4.58 | 138.00 | 137.00 | 196.00 |
| 100 | 138.11 | 1.76 | 138.00 | 137.00 | 180.00 |
| 500 | 138.15 | 0.43 | 138.00 | 137.00 | 140.00 |
| 1000 | 138.30 | 0.49 | 138.00 | 137.00 | 141.00 |

### D. Italy-Brazil-Italy - MQTT

Fig. 10 and Table VII show the result times for MQTT messages initiated in Italy. As well as the other measurements initiated in Italy, these times are very close to each other, showing the consistency of the network and protocol. Besides that, the times are very close to WebSocket times and considering the average values, they are very close to round-trip times when connections are initiated in Brazil. This shows a consistent result for both protocols in both ways.

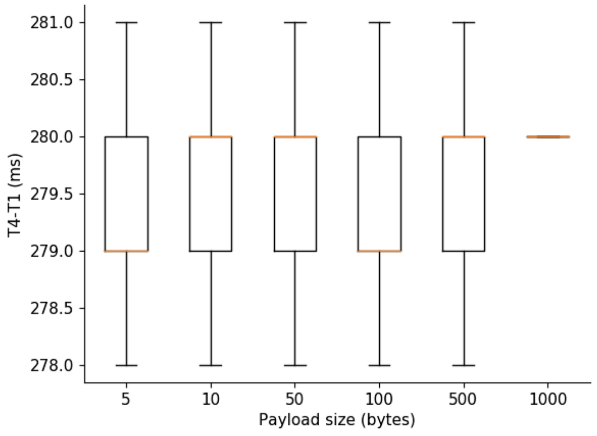

Fig. 10. Italy-Brazil-Italy - Round Trip

TABLE VII
ITALY-BRAZIL-ITALY - MQTT - ROUND TRIP

| Payload size | T4 - T1 | | | | |
|---|---|---|---|---|---|
| | Mean | Std. Deviation | Median | Min. | Max. |
| 5 | 279.86 | 2.63 | 279.00 | 278.00 | 325.00 |
| 10 | 282.33 | 12.46 | 280.00 | 278.00 | 460.00 |
| 50 | 280.35 | 15.83 | 280.00 | 278.00 | 773.00 |
| 100 | 280.16 | 15.64 | 279.00 | 278.00 | 773.00 |
| 500 | 280.90 | 15.64 | 280.00 | 278.00 | 763.00 |
| 1000 | 280.33 | 1.82 | 280.00 | 279.00 | 311.00 |

## VI. CONCLUSION

In the context of industry 4.0, a small network latency is required for applications that require real-time and cloud data monitoring and analysis.

This paper presents an experiment for latency measurement in a possible application architecture for industry 4.0, considering network layer application protocols used in ICT IoT. In addition, it is considered an intercontinental distribution scenario.

The mean values of round-trip time using both protocols and for the Brazil-Italy-Brazil and Italy-Brazil-Italy paths are concentrated below 300 milliseconds. For just one direction, the time values are concentrated below 150 milliseconds. It is important to emphasize that the mean latency values are similar considering each payload size but differ for each analyzed path due to variations in the network. It is not possible to assign these variations to the use of one protocol or other with the results obtained, requiring further analysis.

For future works, the same measurements can be made considering many server variables as CPU and memory usage, and additionally, use limited computational IoT devices as clients instead full capable computers, for a more realistic scenario.

## REFERENCES

[1] Alasdair Gilchrist. 2016. Industry 4.0: The Industrial Internet of Things (1st ed.). Apress, Berkely, CA, USA.
[2] Yang Lu. Industry 4.0: A survey on technologies, applications and open research issues. Journal of Industrial Information Integration. Volume 6, 2017, Pages 1-10.
[3] Zurawski, R. (2017). Industrial Communication Technology Handbook. City: CRC Pr I Llc.
[4] M. R. Palattella et al., "Internet of Things in the 5G Era: Enablers, Architecture, and Business Models," in IEEE Journal on Selected Areas in Communications, vol. 34, no. 3, pp. 510-527, March 2016.
[5] M. Iglesias-Urkia, A. Orive, M. Barcelo, A. Moran, J. Bilbao and A. Urbieta, "Towards a lightweight protocol for Industry 4.0: An implementation based benchmark," 2017 IEEE International Workshop of Electronics, Control, Measurement, Signals and their Application to Mechatronics (ECMSM), Donostia-San Sebastian, 2017, pp. 1-6.
[6] Shekhada D., Stiller M., Salvi A. (2018) A Comparison of Current Web Protocols for Usage in Cloud based Automation Systems. In: Jasperneite J., Lohweg V. (eds) Kommunikation und Bildverarbeitung in der Automation. Technologien fr die intelligente Automation (Technologies for Intelligent Automation). Springer Vieweg, Berlin, Heidelberg
[7] M. Wollschlaeger, T. Sauter and J. Jasperneite, "The Future of Industrial Communication: Automation Networks in the Era of the Internet of Things and Industry 4.0," in IEEE Industrial Electronics Magazine, vol. 11, no. 1, pp. 17-27, March 2017.
[8] Erik Hofmann, Marco Rsch. Industry 4.0 and the current status as well as future prospects on logistics. Computers in Industry. Volume 89, 2017. Pages 23-34.
[9] Jay Lee, Behrad Bagheri, Hung-An Kao. A Cyber-Physical Systems architecture for Industry 4.0-based manufacturing systems. Manufacturing Letters. Volume 3, 2015, Pages 18-23.
[10] FETTE, Ian.(2011, Dec.) *The websocket protocol*.[Online]. Available: https://tools.ietf.org/html/rfc6455.
[11] LUBBERS, Peter; GRECO, Frank.*HTML5 WebSocket: A Quantum Leap in Scalability for the Web*.[Online]. Available: http://www.websocket.org/quantum.html.
[12] *Websocket: BROWSER APIS AND PROTOCOLS, CHAPTER 17*.[Online]. Available:https://hpbn.co/websocket/.
[13] MQTT Version 3.1.1. Edited by Andrew Banks and Rahul Gupta. (2014, Oct.). OASIS Standard.[Online]. Available: http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html. Latest version: http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/mqtt-v3.1.1.html.
[14] Hillar, G.C. MQTT Essentials - A Lightweight IoT Protocol. [S.L.]: Packt Publishing, 2017, 280 p.
[15] BOYD, B. et al. Building Real-time Mobile Solutions with MQTT and IBM MessageSight. [S.L.]: IBM Redbooks, 2014. 264 p.
[16] Brandao, D., Ferrari, P., Rocha, M., Sisinni, E. "Evaluation of communication latency in industrial IoT applications", 2017 IEEE International Workshop on Measurement and Networking (M&N), pp 1-6, 2017