International Journal of Electrical and Computer Engineering (IJECE)
Vol. 12, No. 6, December 2022, pp. 6293~6302
ISSN: 2088-8708, DOI: 10.11591/ijece.v12i6.pp6293-6302

Message queue telemetry transport and lightweight machine-to-machine comparison based on performance efficiency under various scenarios

Drishti Singh, Ria Singh, Aryan Gupta, Ambika Vishal Pawar
Department of Computer Science and Information Technology, Symbiosis Institute of Technology, Pune, India

Article Info

Article history:

Received Jul 29, 2021
Revised Aug 13, 2022
Accepted Aug 23, 2022

Keywords:

Constrained application protocol
Internet of things
Lightweight machine-to-machine
Message queue telemetry transport

ABSTRACT

Internet of things (IoT) is been advancing over a long period of time in many aspects. For data transfer between IoT devices in a wireless sensor network, various IoT protocols are proposed. Among them, the most widely used are constrained application protocol (CoAP) and message queue telemetry transport (MQTT). Overcoming the limitations of CoAP, lightweight machine-to-machine (LwM2M) framework was designed above CoAP. Recent statistics show that LwM2M and MQTT are the widely used, but LwM2M is still less used than MQTT. Our paper is aimed at comparing both MQTT and LwM2M on the basis of performance efficiency, which will be achieved by sending same file through both protocols to the server. Performance efficiency will be calculated in two scenarios, i) when the client makes a connection with the server i.e., while initial connection and ii) while sending data file to server i.e., while data transfer. Both the protocols will be tested on the number of packets sent and the variability of packet size throughout the session. Experimental results indicated that LwM2M outperformed MQTT in both above scenarios by almost 69%. Therefore, we concluded by stating that LwM2M is best choice over MQTT, but MQTT can still be used in some situations if necessary.

Corresponding Author:

Ambika Vishal Pawar
Department of Computer Science and Information Technology, Symbiosis Institute of Technology
Lavale, Pune, India
Email: ambikap@sitpune.edu.in

1. INTRODUCTION

Internet of things (IoT) clearly suggests connectivity between devices or in lay man terms, devices talking with each other. With increasing technology, human interaction within devices led to decreasing accuracy of the results and even a single mistake can lead to the whole system crashing. Therefore, the need of interactivity i.e., machine-to-machine communication arose. IoT mainly focuses on living with artificial yet intelligent devices to perform daily activities. The best example feasible today is the smart home [1]. Simply, IoT enables things to interact and co-ordinate among themselves thereby reducing human intervention in basic everyday tasks, as shown in Figure 1 [2]. Despite the pandemic in 2020, the IoT devices increased to 11.7 billion devices, which indicates that by 2025, it is expected that there will be more than 30 billion IoT devices connected, and at an average, 4 devices per person.

IoT is not a single unit but a collection of various technologies that work in synchronization. For achieving this, IoT devices consist of sensors, actuators, and processors, which mainly collect data [3]. For processing of this data, a medium of data transfer is required. This gave rise of many IoT protocols which are discussed below. For example, collection and analyzation of data that are available from IoT devices like

sensors provides various types of information such as behavior of data, and behavior of the users. So, in order to build such a system, various and large amounts of IoT devices need to be connected to each other where each device is performing some different tasks that is assigned to it. Hence, communication in a system such as this becomes highly heterogeneous and requires the use of specific mediums through which these devices can communicate with each other with minimal loss in data rates, and packets. This gave rise to many IoT protocols and framework for heterogeneous and constrained devices [4].

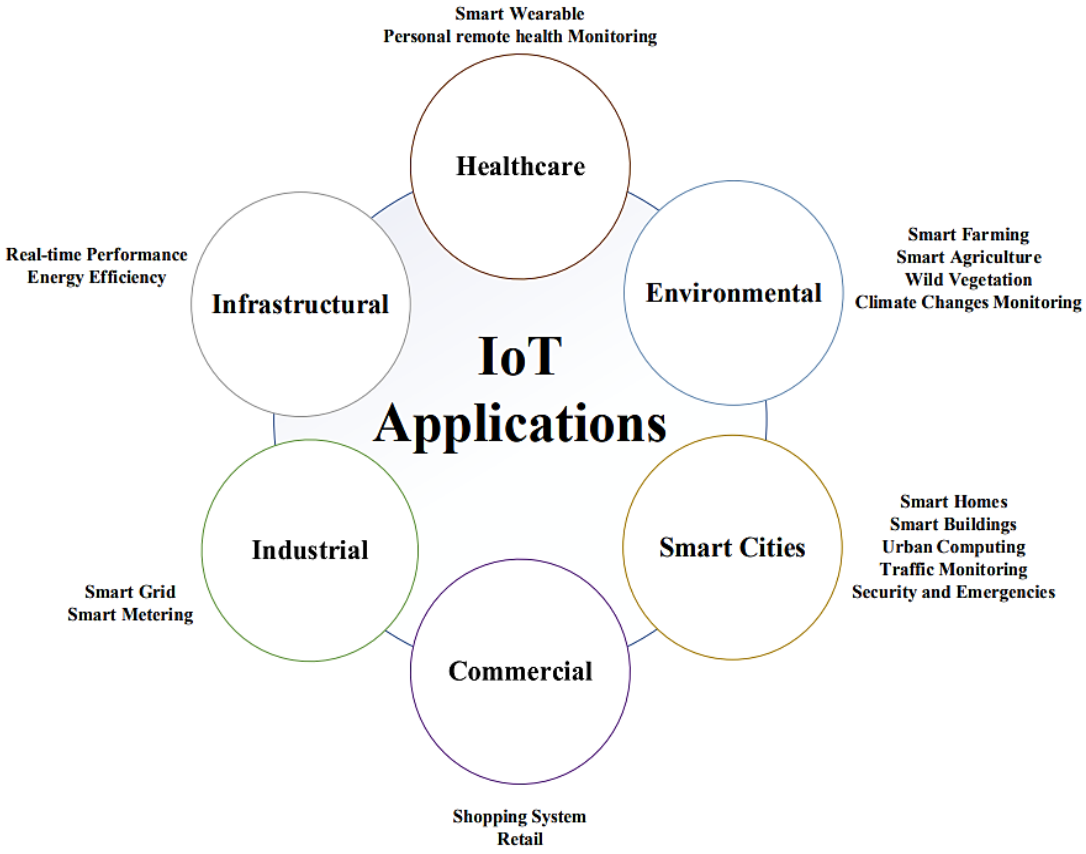

Figure 1. Applications of IoT [5]

The aim of this paper is the compare the performance in terms of packet communication between a protocol and a framework. In order to achieve high quality communication on large network and real time data, protocols and frameworks both should contain appropriate features such as low data loss, high packet transmission time, and low response time [6]. In the view of this purpose, performance of message queue telemetry transport (MQTT) protocol and lightweight machine-to-machine (LwM2M) framework is examined by deploying it on embedded systems. In this study, unlike existing ones, a comparative analysis is done between a protocol and a framework and the results of which provide readers a better understanding of the limitations and strengths of both IoT protocols and frameworks. Section 2 gives an overview of the IoT protocols widely used for communication with IoT devices. Section 3 specifies the method of transfer of packets through both the protocols with the necessary software requirements and appropriate explanation. Section 4 presents the results and discussion for both conditions: while initial connection of client with server and while data transfer. Section 5 concludes the previous results and compares LwM2M and MQTT.

2. BACKGROUND

The very first and important step is to establish the connection between devices. Wireless communication in IOT always takes place over the radio, and various frameworks combined with protocols are used to ensure the same. The IoT protocols such as MQTT, constrained application protocol (CoAP), and advanced messaging queuing protocol (AMQP). enables the communication with the hardware on the client

side without use of internet connection which makes it essentially useful. Different types of protocols based on layers are used to enable effective messaging between devices. From various of the past studies, it has been found that MQTT and CoAP are the most stable protocols that are available [7]. Based on CoAP, LwM2M protocol has been designed. Our paper mainly tries to focus on LwM2M and MQTT protocols in terms of their performance, power, and connection stability.

2.1. MQTT

IoT has seen a great demand and usage in the recent few decades. MQTT was developed by Andy Stanford-Clark of IBM and Arlen Nipper of Arcom in 1999. It was developed for communication among remote devices having unreliable communication, mainly in monitoring oil pipelines using satellite communication. MQTT also provides a three-level layer of quality of service (QoS). This agreement guarantees the message delivery to the receiver taking in consideration the network bandwidth and reliability and what is the priority of the message according to the sender. These features make MQTT a better choice for deploying large scale IoT architecture. In this domain, the features of MQTT are: i) simple architecture; ii) client-server model through publish/subscribe communication; iii) one-to-many communication system; and iv) low bandwidth consumption QoS [8].

The main architecture of MQTT is based on client server model and the main components are: publisher, subscriber, MQTT broker [9]. This is mainly known as publish/subscribe communication, which is an asynchronous communication, where mass communication takes place from publishers to broker to intended subscribers [10]. The publishers generate data according to the constraints and conditions laid out by the IoT system and publish it to the message broker. The broker filters the message and sends it to the intended subscribers. The publishers and subscribers register themselves with the broker through password authentication [11]. For example, in a home automation system, the publishers can be sensors measuring quantities like temperature, humidity and the subscribers will be the smartphone or temperature controller or air conditioner controller, which will receive the data and take decisions accordingly as shown in Figure 2. This model allows communication even between those subscribers and publishers, which are known aware of each other. This allows them to maintain their privacy and communicate effectively via the publicly available broker. It is not necessary for the publishers and subscribers to be available at the same time.

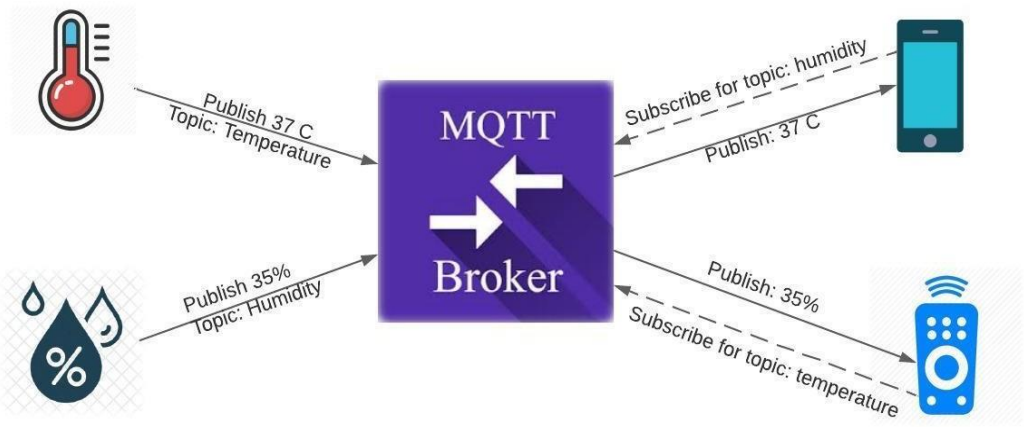

Figure 2. Home automation system using MQTT

MQTT is the most emerging IoT protocol in the present time. This is mainly because hypertext transfer protocol (HTTP) protocol becomes highly unreliable as far as communication between IoT devices are concerned since only 2 devices can involve in communication via HTTP protocol i.e., it does not support multi device communication. Hence there was a major need for a protocol that can achieve the same but utilize minimum resources to enable reliable communication between various devices [12]. Hence MQTT came into existence and was used for machine-to-machine communication because of its simple architecture and low bandwidth consumption. It is built on transmission control protocol/internet protocol (TCP/IP); therefore, it is a secure protocol.

2.2. CoAP

CoAP is another such protocol, which is widely used in IoT systems, is a synchronous request-response application layer protocol designed mainly for constrained networks (micro-controllers) or medium

scaled IoT systems. It is mostly used for Machine-to-machine communication and since it makes use of user datagram protocol (UDP) protocol instead of TCP protocol for data transfer, it has lower overhead [13]. It is the underlying protocol on which the LwM2M framework is designed, which will be discussed further. Basic features of CoAP [14] are: i) involves low overhead and simple model; ii) web protocol which is used in M2M constrained networks; and iii) ability to exchange asynchronous messages.

CoAP is very similar to HTTP, but still features vary between them. CoAP allows the client and the server to involve independently in the communication, which is similar to HTTP [15], [16]. Since, CoAP is built on UDP, it does not have the secure socket layer/transport layer security (SSL/TSL) to provide security. Instead of this, datagram transport layer security (DTLS) is enabled, which provides security for UDP based protocols. Since both MQTT and CoAP are advanced level protocols and provide features to enable communication between IoT devices, it becomes really essential to evaluate them based on their features. A comparative analysis between MQTT and CoAP protocol is shown in Table 1.

Table 1. Difference between CoAP and MQTT [10]

MQTT	CoAP
It is a many-to-many communication protocol for transfer of data between the sender and the receiver through the message broker.	It is a one-to-one communication protocolbetween the sender and the receiver.
MQTT clients create a long-lived TCP connection with the broker, ensuring no problem.	Since, CoAP uses UDP, tunneling has to beused to allow CoAP in such cases [17]
Before, the communication starts, the client/sender has to be well aware of theformat of the message.	CoAP provides the facility of discovering other formats by allowing the communication.

2.3. LwM2M

Since the emergence of IoT, it has experienced countless changes and still is changing by the day. The need to change comes from the need of making IoT more robust and efficient. Also, the advancement of technology and the limited resources has pushed the need for smaller and smaller devices to be deployed, with such small devices to be used as a part of the IoT network and to keep them robust and efficient we run into problems of power delivery and effective communication within the network. For a large-scale communication within a network of thousands of nodes it requires a lot of power which is not possible if we want to keep the device small. So a protocol was needed to be developed that can run on low power while also being able to communicate effectively within a huge network [18]. It also needed to be light enough to get stored inside any tiny device.

So, in 2012, open mobile alliance (OMA) developed the lightweight machine to machine protocol (LwM2M). A protocol designed specifically to reduce data and power consumption. LwM2M wraps around COAP and uses it as an application transfer protocol to establish connection between server and client.

Even though the existence of MQTT and CoAP made the M2M communication safer and more efficient, still, due to massive expansion of IoT in the need of dedicated solutions for proper and innovation deployment, there was a need for quicker, easier and more effective communication consuming less power. Therefore, lightweight M2M protocol was developed by open mobile alliance SpecWorks (OMA) [19]. Lightweight M2M is a communication protocol, mainly for remote device management, designed in order to perform effectively while consuming less power and data, thus, helping in designing remote-constrained devices.

The architecture of lightweight M2M [20], [21] consists mainly of 4 logical interfaces as shown in Figure 3:

- Bootstrapping interface: no pre-configuration of devices will need to be done in thefactory. The devices will be configured according to the requirement.
- Client registration interface: the server is aware of all the specifications and functionality of the client configured with it and also issues software updates,whenever necessary.
- Device management and service enablement interface: The provider has the freedom to access the client objects' instances in order to manage its functionality and change parameters.
- Information reporting interface: similar to MQTT, LmwM2M also follows publish/subscribe model. This allows the user to get timely error reports of devicescurrently not in service and their status also.

Some features of LwM2M match with transport layer security, like supporting the DTLS 1.2+ and TLS 1.2+ protocols and also ensures end-to-end application layer security through OSCORE [22]. Earlier, this protocol could work only on UDP but the arrival of LwM2M 1.1 version. It also supports TCP protocol, providing more reliable communication and also helpful in traversing firewall and NAT restrictions.

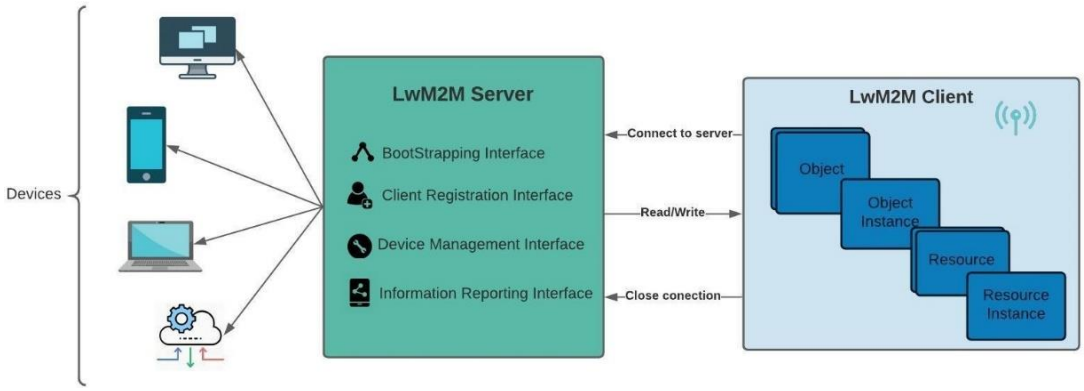

Figure 3. LwM2M server client architecture

3. RESEARCH METHOD

The comparison of LwM2M and MQTT is carried out by deploying the LwM2M and MQTT client on the raspberry pi, which is a microcontroller board. The MQTT and LwM2M clients are deployed in different environments. For LwM2M, AVSystem Coiote IoT device management platform was used for the following features: i) it provides a user-friendly interface for deploying the client and server, which is easy to learn for beginners. We used it for its auto detection feature and easy to learn and ii) it had the feature to easily find errors using its maintenance and anomaly detection system.

Therefore, according to this device management system, AVSystem Anjay Library v.1.16 [23] closed source variant without bootstrap enabled was chosen as the LwM2M client. The client and server objects were created and were complied with support for DTLS. This enables automatic encryption while communication using the security option. AVSystem Anjay Library uses the binary TLV format for transferring data packets while data transmission.

For MQTT client implementation, Amazon web services (AWS) IoT [24] environment was used for the following features: i) Amazon AWS IoT provides an efficient message broker with high throughput. It is highly flexible while two-way communication and is customizable and ii) AWS IoT core provides X.509 certificate-based authentication and token-based authentication. For the same environment, AWS IoT SDK for python was used as MQTT client. AWS IoT core uses the JSON format to transfer data packets while transmission. In the experimentation, the efficiency of LwM2M and MQTT will be accessed by comparing packet transfer rate and quantity. For this, Wireshark [25] tool was used to capture packets from client to server while communication. Following is the table to summarize the above used tools and management systems as shown in Table 2.

Table 2. Software and hardware specifications

Technology	Specification
Raspberry Pi	Raspberry Pi 4
IoT platform for MQTT Client	Amazon AWS IoT
MOTT Client	AWS IoT SDK for python
MQTT Security	TLS with X.509 certificates with AWS IoTendpoint
IoT platform for LwM2M client	AVSystem Coiote IoT device management platform
LwM2M client	AVSystem Anjay Library v.1.16 closed-source variant without bootstrap enabled
LwM2M security	DTLS with certificate-based authenticationand encryption mode.
Router for packet capture	pfSense router upstream

Both the IoT clients were configured to send three-parameter message. Each consisted of a string, an integer and a float, which was encapsulated into a single object over communication. The packet capture rates, and quantity were analyzed in mainly two phases of data communication between client and server.

- Initial connection of client with server: this includes measuring the data transmission during the initial device registration with an IoT platform or after rebooting the device. Here, delivery of packets between the IoT client and the platform was analyzed during the initial. Initial connection refers to the time when the device registers itself with the IoT platform when the device reboots, so the device re-registers itself. For MQTT, the initial connection was made to the AWS MQTT broker. For LwM2M, a non-bootstrap connection was established between the client/device and the AVSystem Coiote LwM2M endpoint. The Packet capture window was 2 minutes.

- Data transfer during active connection: this includes measuring the data transmission for one-time, on-device configuration parameter update, while pushing a single command to a device. For MQTT, the testing was not done on AWS IoT's device model. Instead, the client was configured to listen to specific topic. For LwM2M, the server-initiated update of the counter was tested within the LwM2M object. The packet capture window was 2 minutes.

3.1. For MQTT connection

The AWS IoT device SDK allows developers to access AWS IoT platform by using MQTT protocol. By connecting their devices to AWS IoT, users can securely work with sending and receiving of messages. The SDK is built on top of MQTT python client library. For this experiment, TLS with X.509 certificates with AWS IoT endpoint were selected for connecting with AWS IoT. For X.509 certificate, AWS IoT root CA was downloaded to get access to a private certificate and a private key. For a basic MQTT connection, the script will look like the following:

```
myMQTTClient.connect()
myMQTTClient.publish("myTopic", "myPayload", 0)
myMQTTClient.subscribe("myTopic", 1, customCallback)
myMQTTClient.unsubscribe("myTopic")
myMQTTClient.disconnect()
```

3.1.1. AWS IoT shadow client

A shadow device is basically JSON document that is used to store and retrieve information of the device. It is used to either get or set the state of device regardless of the fact that the device is connected to internet or not. The SDK allows applications to retrieve, update and delete shadow documents in one MQTT connection. Following is the how the script will look like for shadow operation using MQTT connection.

```
myShadowClient.connect()
# Create a device shadow instance using persistent subscription
myDeviceShadow=myShadowClient.createShadowHandlerWithName("Bot", True)
# Shadow operations
myDeviceShadow.shadowGet(customCallback, 5)
myDeviceShadow.shadowUpdate(myJSONPayload, customCallback, 5)
myDeviceShadow.shadowDelete(customCallback, 5)
myDeviceShadow.shadowRegisterDeltaCallback(customCallback)
myDeviceShadow.shadowUnregisterDeltaCallback()
```

Using these two functionalities, connection was established and was experimented based on the messages being sent from client to server.

3.2. For LwM2M connection

For setting up the Coiote IoT Device management server, one has to register at Anjay AVSystem official website. Then by filling the appropriate credentials, the server can be set up. For creating the Anjay client on Raspberry Pi, first the Anjay is initialized using CMake as the build system and then the Anjay object is instantiated. Following is the code snippet to initialize the client object.

```
const configuration CONFIG={
.endpoint_name=argv[1],
.in_buffer_size=4000,
.out_buffer_size=4000,
.msg_cache_size=4000
};

anjay_t*anjay=anjay_new(&CONFIG);
          if (!anjay) {
                    avs_log(tutorial, ERROR, "Could not create Anjay object");
          return -1;
```

Once the Anjay client is run, then if this is successful, a message of "registered successfully" is received every 30 seconds in the log. This means that client and server have been connected and ready for any activity like read or write. Since Anjay does not have any self-contained loops for maintaining incoming packets, therefore they need to be handled manually. The process flow in explained further in reference to Figure 4: i) firstly, all the server socket ports are obtained from the *AVS_LIST* available in AVSystem common library; ii) the expected time of the job is to be calculated and accordingly the *poll()* method is directed to wait for the next incoming request but not for long, iii) the *anjay_serve()* method is called on

incoming packet to manage the messages. If a message is split in many packets (in our case 2 packets), then the incoming packets will be handled by inside libraries. This function basically blocks everything till the intended message is completely handled properly, iv) the *send()* method is used to send the original message for the request, and v) then the scheduler is run for other pending jobs to continue to with their execution.

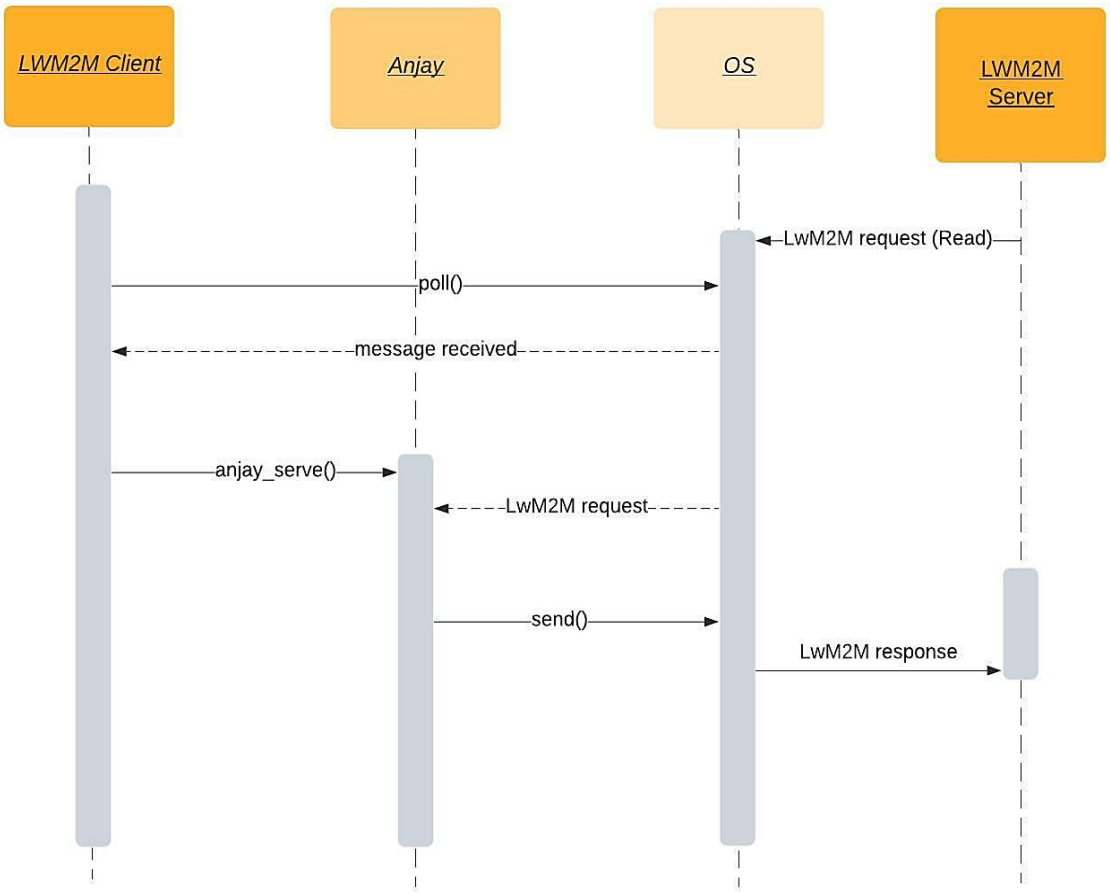

Figure 4. Communication flow between client and server using LWM2M Anjay library

4. RESULTS AND DISCUSSION

4.1. While initial connection of client with server

Table 3 displays the inefficiency of JSON format of MQTT over binary TLV format of LwM2M during registration of client with the IoT platform. LwM2M packets lengths are firstly less variable than MQTT and are around 7% less than average MQTT packet length as shown in Table 4 and Figure 5. Therefore, LwM2M becomes a perfect choice for networks with constrained message transmission unit (MTU) sizes.

4.2. While data transfer during active connection:

From Table 5, it is inferred that LwM2M is more efficient than MQTT in transferring data during single platform-to-device. The test helped in comparing the protocol efficiency and overhead, since each protocol transmits and update of the smallest possible size. From Table 6, LwM2M packet lengths are consistent in size while MQTT packets have varied size, as shown in Figure 6.

Table 3. Comparison and inference of MQTT and LwM2M according to criterions

Criterion	MQTT	LwM2M	Inference drawn
Total Bytes transferred	11560	3489	LwM2M transferred approximately 69.8% less bytes of data than MQTT.
Total Packetstransferred	68	25	LwM2M transferred approximately 63.2% less packets of data than MQTT.

Table 4. Count of packets lying in packet length ranges

Packet Lengths (Bytes)	Packets sent (LwM2M)	Packets Sent (MQTT)
0-19	0	0
20-39	1	15
40-79	6	24
80-159	2	4
160-319	4	2
320-639	1	4
640-1279	0	4

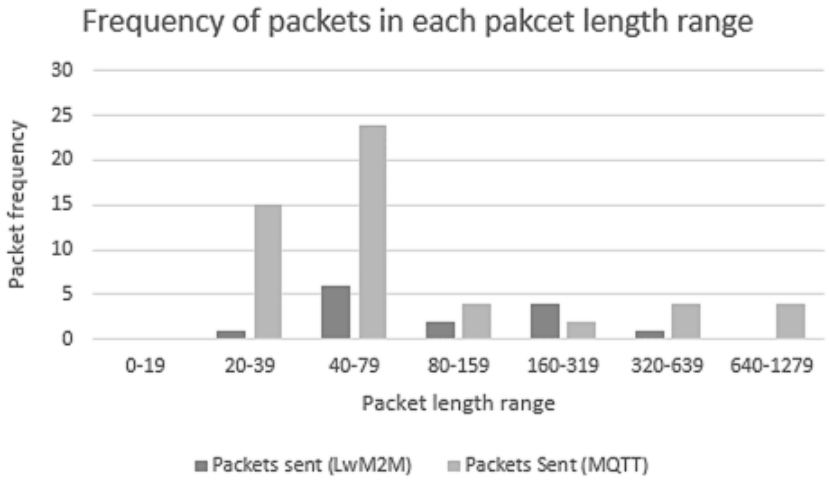

Figure 5. Count of packets lying in packet length ranges

Table 5. Testing observations according to the criterion mentioned

Criterion	MQTT	LwM2M	Inference drawn
Total Bytes transferred	365	215	LwM2M transferred approximately 41% less bytes of data than MQTT.
Total Packets transferred	5	4	LwM2M transferred 1 less packet than MQTT.

Table 6. Packets transferred for each range of packet length by LwM2M and MQTT

Packet Lengths (Bytes)	Packets sent (LwM2M)	Packets sent (MQTT)
0-19	0	0
20-39	0	0
40-79	0	1
80-159	1	2
160-319	0	0
320-639	0	0
640-1279	0	0
0-19	0	0

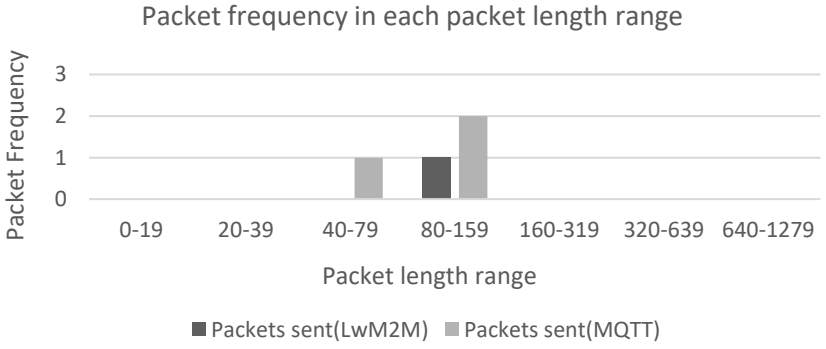

Figure 6. Packets transferred for each range of packet length by LwM2M and MQTT

5. CONCLUSION

In this paper, we presented a comparison of LwM2M and MQTT, which are both popular and widely used communication protocol or frameworks and many times, even used interchangeably. We used two scenarios, when a client registers with the IoT platform and when the client actively sends data packets to the server. By sending same data object, we analyzed the number of data packets transferred and the number of packets sent for each range of packet lengths.

As the results clearly state that LwM2M outperformed MQTT in both scenarios. Since, LwM2M sends less packets for the same data object than MQTT, this states that LwM2M requires less resources (for eg, and power consumption), which makes it faster and more efficient than the latter. Also, LwM2M provides more flexibility and granular control over the exact type and content of messages between the devices and the IoT platform.

From the results it is clear that, LwM2M packet lengths are consistent in size while MQTT packets have varied size and LWM2M can transfer a single file in less packets as compared to MQTT. Therefore, it is more efficient, reliable and fast to use LwM2M for communication between IoT devices than MQTT. Since, in this experiment, only three-parameter message was considered for data transmission, we can try the efficiency of LwM2M and MQTT on large data, since the results can be different for the above to find out which one works best in large traffic. Also, more platforms available can be used to deploy LwM2M clients and MQTT clients, which can also alter the results due to environment change.

ACKNOWLEDGEMENT

We would like express our sincere gratitude to our college for providing funds for this publication and for their constant support.

REFERENCES

[1] P. Govikadhasan and B. Gayathri, "Internet of things (IoT)-survey on concepts, challenges, technologies, security, enterprise integration and applications," *International Journal of Electrical, Electronics and Computer Science Engineering*, pp. 35–37, 2018.

[2] S. Vashi, J. Ram, J. Modi, S. Verma, and C. Prakash, "Internet of things (IoT): A vision, architectural elements, and security issues," in *2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*, Feb. 2017, pp. 492–496, doi: 10.1109/I-SMAC.2017.8058399.

[3] P. Sethi and S. R. Sarangi, "Internet of things: architectures, protocols, and applications," *Journal of Electrical and Computer Engineering*, pp. 1–25, 2017, doi: 10.1155/2017/9324035.

[4] M. A. Iqbal, S. Hussain, H. Xing, and M. Imran, *Enabling the internet of things*. Wiley, 2021, doi: 10.1002/9781119701460.

[5] R. Hassan, F. Qamar, M. K. Hasan, A. H. M. Aman, and A. S. Ahmed, "Internet of things and its applications: a comprehensive survey," *Symmetry*, vol. 12, no. 10, Oct. 2020, doi: 10.3390/sym12101674.

[6] S. Sinche *et al.*, "A survey of IoT management protocols and frameworks," *IEEE Communications Surveys and Tutorials*, vol. 22, no. 2, pp. 1168–1190, 2020, doi: 10.1109/COMST.2019.2943087.

[7] T. M. Tukade and R. M. Banakar, "Data transfer protocols in IoT-an overview," *International Journal of Pure and Applied Mathematics*, vol. 118, no. 16, pp. 121–138, Jan. 2018.

[8] D. Silva, L. I. Carvalho, J. Soares, and R. C. Sofia, "A performance analysis of internet of things networking protocols: evaluating MQTT, CoAP, OPC UA," *Applied Sciences*, vol. 11, no. 11, May 2021, doi: 10.3390/app11114879.

[9] B. Mishra and A. Kertesz, "The use of MQTT in M2M and IoT systems: a survey," *IEEE Access*, vol. 8, pp. 201071–201086, 2020, doi: 10.1109/ACCESS.2020.3035849.

[10] R. A. Al-Qassab and M. I. Aal-Nouman, "Performance evaluation of CoAP and MQTT_SN protocols," in *International Conference on Emerging Technology Trends in Internet of Things and Computing*, 2022, pp. 223–236, doi: 10.3390/app9050848.

[11] S. Pal, S. Ghosh, and S. Bhattacharya, "Study and implementation of environment monitoring system based on MQTT," *Environmental and Earth Sciences Research Journal*, vol. 4, no. 1, pp. 23–28, Mar. 2017, doi: 10.18280/eesrj.040105.

[12] D. Dinculeană and X. Cheng, "Vulnerabilities and limitations of MQTT protocol used between IoT devices," *Applied Sciences*, vol. 9, no. 5, Feb. 2019, doi: 10.3390/app9050848.

[13] L. Coetzee, D. Oosthuizen and B. Mkhize, "An Analysis of CoAP as Transport in an Internet of Things Environment," *2018 IST-Africa Week Conference (IST-Africa)*, 2018, pp. 1-7.

[14] M. Joshi and B. Pal Kaur, "CoAP protocol for constrained networks," *International Journal of Wireless and Microwave Technologies*, vol. 5, no. 6, pp. 1–10, Nov. 2015, doi: 10.5815/ijwmt.2015.06.01.

[15] M. Martí, C. Garcia-Rubio, and C. Campo, "Performance evaluation of CoAP and MQTT_SN in an IoT environment," in *13th International Conference on Ubiquitous Computing and Ambient Intelligence UCAmI 2019*, Nov. 2019, vol. 31, no. 1, doi: 10.3390/proceedings2019031049.

[16] C. Bayılmış, M. A. Ebleme, Ü. Çavuşoğlu, K. Küçük, and A. Sevin, "A survey on communication protocols and performance evaluations for Internet of Things," *Digital Communications and Networks*, pp. 334–340, Mar. 2022, doi: 10.1016/j.dcan.2022.03.013.

[17] C. Gündoğan, P. Kietzmann, M. Lenders, H. Petersen, T. C. Schmidt, and M. Wählisch, "NDN, CoAP, and MQTT," in *Proceedings of the 5th ACM Conference on Information-Centric Networking*, Sep. 2018, pp. 159–171, doi: 10.1145/3267955.3267967.

[18] B. Sudharsan, J. G. Breslin, and M. I. Ali, "Adaptive strategy to improve the quality of communication for IoT edge devices," in *2020 IEEE 6th World Forum on Internet of Things (WF-IoT)*, Jun. 2020, pp. 1–6, doi: 10.1109/WF-IoT48130.2020.9221276.

[19] M. Pappalardo, G. Tanganelli, and E. Mingozzi, "Enhanced support of LWM2M in low power and lossy networks," in *2020 IEEE International Conference on Smart Computing (SMARTCOMP)*, Sep. 2020, pp. 344–349, doi:

10.1109/SMARTCOMP50058.2020.00075.
[20] D. Tracey and C. Sreenan, "OMA LwM2M in a holistic architecture for the internet of things," in *2017 IEEE 14th International Conference on Networking, Sensing and Control (ICNSC)*, May 2017, pp. 198–203, doi: 10.1109/ICNSC.2017.8000091.
[21] C. Martín, I. Alba, J. Trillo, E. Soler, B. Rubio, and M. Díaz, "Providing reliability and auditability to the IoT LwM2M protocol through Blockchain," *arXiv:2008.06694v1*, Aug. 2020
[22] Y. Kuwabara, T. Yokotani, and H. Mukai, "Hardware emulation of IoT devices and verification of application behavior," in *2017 23rd Asia-Pacific Conference on Communications (APCC)*, Dec. 2017, pp. 1–6, doi: 10.23919/APCC.2017.8304040.
[23] A. A. Corici, M. Corici, E. Troudt, B. Riemer, and T. Magedanz, "Framework for trustful handover of M2M devices between security domains," in *2020 23rd Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN)*, Feb. 2020, pp. 102–109, doi: 10.1109/ICIN48450.2020.9059457.
[24] D.-H. Kang *et al.*, "Room temperature control and fire alarm/suppression IoT service using MQTT on AWS," in *2017 International Conference on Platform Technology and Service (PlatCon)*, Feb. 2017, pp. 1–5, doi: 10.1109/PlatCon.2017.7883724.
[25] V. Jain, "Getting familiar with wireshark," in *Wireshark Fundamentals*, Berkeley, CA: Apress, 2022, pp. 35–78, doi: 10.1007/978-1-4842-8002-7_2.

BIOGRAPHIES OF AUTHORS

Drishti Singh BTech Information technology Student at Symbiosis Institute of Technology, Lavale, Pune, India and working as Intern in IT department of a finance-based firm. She is highly compassionate about her new job as well as researching for new ideas mostly in the areas of autonomous vehicles and IoT. She can be contacted at email: drishti.singh.btech2018@sitpune.edu.in.

Ria Singh BTech Information technology Student at Symbiosis Institute of Technology, Lavale, Pune, India. Currently, she is completing an internship from a telecommunication company. Her research interests include Internet of things (IoT), Software defined networking (SDN), Network function virtualization (NFV) and 5G networks. She can be contacted at email: ria.singh.btech2018@sitpune.edu.in.

Aryan Gupta BTech Information technology Student at Symbiosis Institute of Technology, Lavale, Pune, India. He is doing an internship at a multinational finance firm, as data analyst. His main interests are artificial intelligence, robotics and IoT. He can be contacted at email: aryan.gupta.btech2018@sitpune.edu.in.

Ambika Vishal Pawar PhD. Computer Engineering and Associate Professor at Symbiosis Institute of Technology, Pune is highly passionate about fields of data mining, cloud computing and artificial intelligence. Her latest research work includes H-Prop and H-Prop-News: Computational Propaganda Datasets in Hindi. She can be contacted at email: ambikadshelke@gmail.com.