A Brief Review of Messaging Protocol Standards for Internet of Things (IoT)

Abdullah Ahmed Omar Bahashwan[1] and Selvakumar Manickam[2]

[1]*National Advanced IPv6 Centre (NAv6), Universiti Sains Malaysia (USM), Penang, Malaysia*
[2]*Senior Lecturer and Researcher at National Advanced IPv6 Centre (NAv6), Universiti Sains Malaysia (USM), Penang, Malaysia*
E-mail: bahashwan@student.usm.my; selva@nav6.usm.my; selva@usm.my

Received 19 June 2018; Accepted 21 September 2018;
Publication 12 October 2018

Abstract

As the most recent development in cyberspace technologies, the Internet of Things (IoT) has received substantial research attention. It now occupies a crucial role in advancing society and industry. IoT merges Social Networks and connects devices to allow people interact one another and simplify information sharing. IoT has two main aspects: the internal and the external. Whereas the latter side consists of sensors, actuators, etc. which are physically likely, the former consists of protocols that are highly important. IoT has particular protocols in different layers such as Transport layer, Physical/Link layer and Application layer, which is accountable for messaging and supplying services. This paper explores the foremost widespread IoT data transmission protocols and their main options. The protocols play a vital role in inventing up-to-date IoT projects and devices. This article provides an uncomplicated review of IoT data protocols based on user requirement.

Keywords: Internet of Things (IoT), Applicaton protocol, Data protocols, MQTT, CoAP, Websocket.

Journal of Cyber Security and Mobility, Vol. 8_1, 1–14. *River Publishers*
doi: 10.13052/jcsm2245-1439.811

1 Introduction

Internet of Things (IoT) has gained considerable attention in the past few years. The term was primarily coined by Kevin Ashton in 1999. Due to rapid developments in mobile communication, Wireless Sensor Network (WSN), Radio Frequency Identification (RFID) and cloud computing, communications through IoT devices have become more convenient than it was before. IoT devices perform effectively by co-operating with one another. The world of IoT includes a large variety of devices such as smartphones, personal computers, laptops, tablets, and other hand-held embedded devices [1].

IoT devices are grounded on more efficient sensors and wireless communication methods to interconnect together, and transfer information to the centralized system. Once the data is processed in a localized system, it is then delivered to the intended destinations. With the rapid growth of communication and internet technology, our daily routines are changing to the fictional space of virtual world, in which individuals can work, shop, chat and have their pets and flowers while living in the actual world. However, it is virtually troublesome to move all human activities to the absolutely machine-controlled world.

The IoT has successfully integrated the fictional space with the actual world on the same platform. Its main objective is configuration of smart climate, smart items and smart homes. Currently, the user satisfaction rate of IoT devices are very high. In China alone, for instance, there are about nine billion IoT devices used which is estimated to reach 24 billion by 2020 [2]. The primary goal of IoT is to make a better world for people in future.

Moreover, IoT has its individual protocol stack which is different from the Open Systems Interconnection model and TCP/IP protocol stacks. IoT model protocol stack is such as Application layer (Protocols are COAP, MQTT, AMQP, and Web-sockets), Internet layer (Protocols are "6LowPAN, IPv4/IPv6, and RPL") and Physical/Link layer (protocols are (IEEE 802.11 series) and (BLE, LTE) [3]. The following part of this paper explores Data protocols, and focuses on MQTT, AMQP, CoAp, and web-socket protocol. And finally, Part 3 concludes the study.

2 Data Protocols

This section analyses standards and protocols in message transmission in IoT application layer enforced by a totally different normalization. Web-based

application and IoT application are IP based, and they use TCP or UDP to transport messages. As there are several message distribution functions that are common among various IoT applications, it is crucial that these functions be employed in proper standards by dissimilar applications [4]. Below are the messaging protocols.

2.1 Message Queue Telemetry Transport (MQTT)

The Message Queue Telemetry Transport protocol (MQTT) was first authored by Andy Stanford-Clark and Arlen Nipper in 1999 and became standardized by OASIS in 2013. It connects the network and the device with applications and middleware. The connection types can be classified as, machine-to-server (M2S), server-to-server (S2S), machine-to-machine (M2M) communication designs and routing mechanisms (one-to-many, one-to-one, and many-to-many) [5]. MQTT works on top of TCP/IP protocol. In addition as a machine-to-machine communication (M2M). MQTT protocol is efficient in limited source devices that are used in very "Low-bandwidth and unreliable links". Similarly, MQTT is constructed over TCP protocol as machine-readable text Transfer Protocol (HTTP) [6].

MQTT consists of four major components, and its prime element is a broker. Broker works as the server, but in a different way. It automatically connects many devices with the help of three different level of QoS. The first is Fire and Forget or Fire the message and Forget which means that not all the acknowledgment is received. The second is delivered at minimum once, which indicates that the message is transported at minimum once, and the acknowledgment is received for each delivery. The third is delivered just once, which means that in order to assure message delivery, 4-way handshaking procedure is required to attach to the broker and gain the data. The fourth component is the Topic, which refers to the identity of the information provided by the broker and the receiving devices that take the information within a topic. Moreover, MQTT is used to publish, subscribe, direct and collect the data. It publishes a response to send the data to broker, from which the subscriber is responsible to receive the data [7].

MQTT is lightweight and can be realized with only 2 KB. In other words, it has lower overhead, bandwidth and latency than the TCP/IP network. This protocol helps messages be delivered successfully to their destinations through the intermediate cloud broker. The client of MQTT performs in two standard ports of 1883 and 8883, which are reserved by Internet Assigned Numbers Authority (IANA). Port 1883 represents MQTT traffic over TCP/IP network while port 8883 is used for MQTT traffic encrypted over TLS [9].

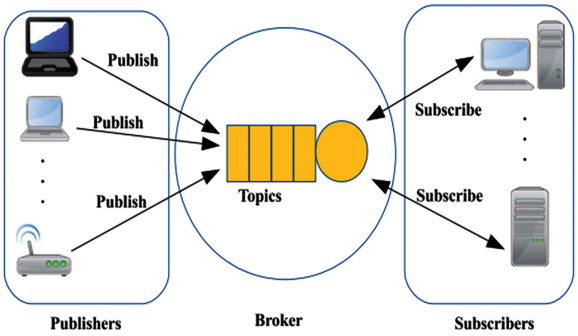

Figure 1 Publisher, Subscribers and Broker [8].

There are two main types of MQTT: MQTT 3.1 and MQTT-SN, known as MQTT-S (v1.2). It is used for wireless communication in low bandwidth atmosphere. Many well-known companies, e.g. Amazon and Facebook, have started to use MQTT in their Web Services [6].

2.2 Constrained Application Protocol

CoAP is a client/server or request and response protocol which meets the requirements of only a constrained node with a low capability in RAM or CPU and a constrained network with lower power (wireless personal area network, WPAN). It is designed by Internet Engineering Task Force, (IETF), which is interested in M2M machines and automation systems. CoAP also supports the publisher/subscribe architecture. The subscriber provides multicast communications, and the publisher sends messages to the subscriber to take action. This situation is not finished in simultaneous mode. The publisher/subscriber architecture is used to support a great size of users by providing better performance than the outdated way. Moreover, CoAP is a web transfer protocol meaning that it is considered to co-operate and work along with HTTP [10].

The most significant features of CoAP area unit are simplicity and reliability. It holds up unicast and multicast requests by taking advantage of UDP, and supplying adroitness to Asynchronous message exchanges. CoAP has only one protocol with dual layers. The first layer is the messaging layer,

and the second one is the request/response layer. The purpose of messaging layer is to accomplish reliability based on UDP while request/response layer aims to operate interactions and communications.

There are some additional features that make CoAP a better protocol in relation to HTTP:

- CoAP works over UDP (User datagram protocol) which helps to avoid the cost of TCP handshake before transmission.
- IN terms of the size, CoAP is just a 4-byte header that affords reliable transfers, and there is no reliable transfer till it uses four types of messages [5].
- CoAP supports a number of Conformable Messages and Non-Conformable Messages such as, Acknowledgement Message, Reset Message, Piggybacked Response, Separate Response, and Empty Message. Figures 2, 3, and 4 show the way messages interconnect with one another:

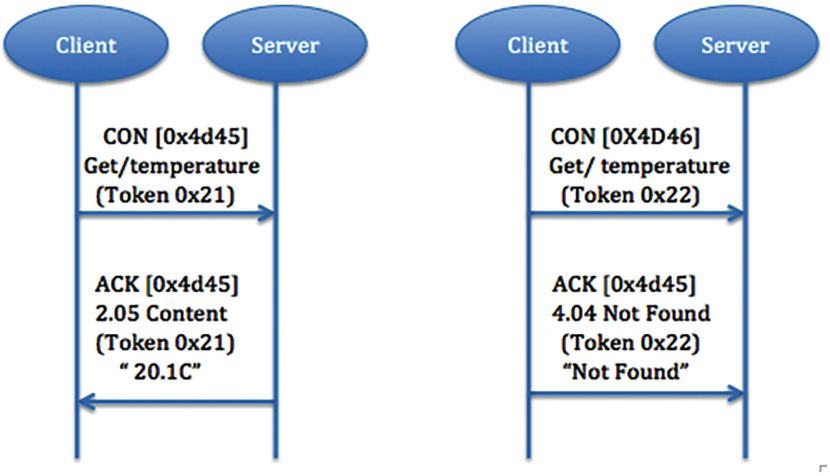

Figure 2 In piggy-backed, successful and failure response results [5].

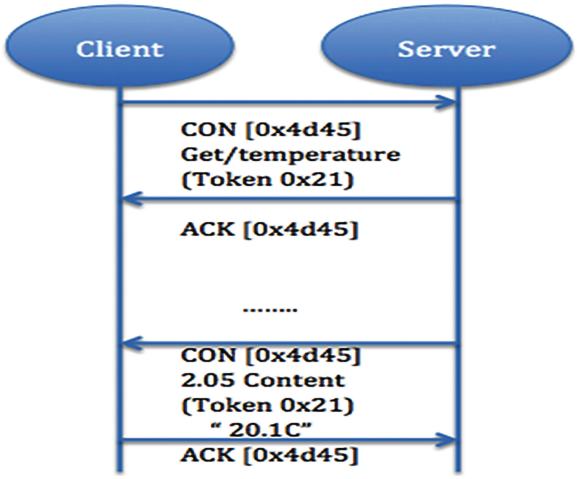

Figure 3 A Get request with an unconnected response [5].

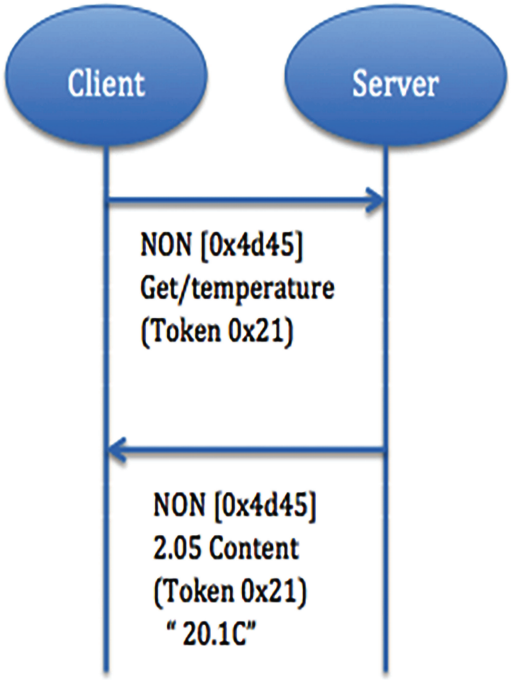

Figure 4 Non-confirmable request and response [5].

2.3 Advanced Message Queuing Protocol (AMQP)

AMQP is an open customary application layer protocol for middleware messaging protocol. It is a uniform protocol by OASIS. Currently, it is broadly used in business and commercial platforms. It is a unique protocol because it supports point to point and publisher/subscriber models, routing and switching. Moreover, it has the ability to do message orientation, queuing, switching reliability and security [10].

AMQP is different from MQTT, and has more advantages. It stores the data, and then forwards it. This feature works when the network is troubleshooting that time guarantees reliability. AMQP conforms reliability with the following message-delivery guarantees:

1. At most once: the message is sent only once, whether received or not.
2. At least once: the message is sent on time.
3. Exactly once: the message is sent only once.

In AMQP, the security held with the TLS protocols are above TCP. Many studies have pointed out that AMQP has low bandwidths and achievement rate. According to one study, AMQP can direct an enormous number of messages per second. Furthermore, AMQP environment with 2,000 users from five continents is able to procedure 300 million messages per day [5]. Figure 5 shows that AMQP component, the broker is divided into two parts of exchange and the queue that hold the communication. The exchange receives publishers' messages and gives them to the queue. The queues send the messages and the data to subscribers.

AMQP is widely run in IoT devices which concentrate on message exchange and communication. It supports many languages, and decreases

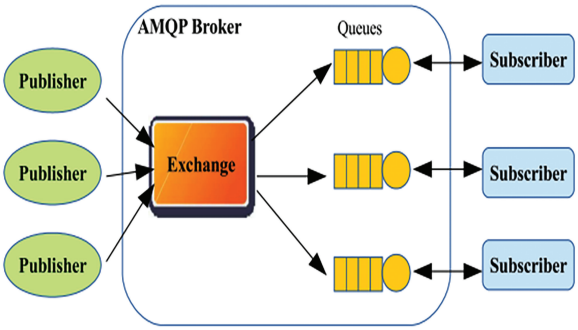

Figure 5 Publish/subscribe of AMQP [8].

communication problems between dissimilar devices. AMQP is a highly flexible protocol that can work in different platforms and environment applications. It supports industrial environment applications, yet it is unsuitable for constrained environments and automation discovery mechanisms.

2.4 Web-Socket

WebSocket protocol was first developed by The World Wide Web Consortium (W3C) and The Web Hypertext Application Technology Working Group (WHATWG), and was fully standardized by the Internet Engineering Task Force (IETF) in 2011. It was established as Part of Hypertext Markup Language five to form unruffle communication channels over Transmission Control Protocol. The Web-Socket protocol session starts without using request/response and publish/subscribe protocols. It is full-duplex communication between a client and a remote server that provides security just like the security model used in a web browser. WebSocket works in TCP, and the applications that use browsers with that do not need to interact and communicate with a remote host [11].

WebSocket diminishes internet communication overhead during actual period of full duplex communications. Its sub-protocols are known as WAMP, which are engaged with publish/subscribe messaging systems. WebSocket only runs over TCP and on no other reliability mechanism [5]. Its sessions are more secure, and because of using TLS/SSL, its message size is 2 byte from the overhead. WebSocket starts the connection. It begins from an HTTP request at the same time that the connection is not built on HTTP. It is layered above the TCP. The connection starts when the client initiates opening handshake by a basic HTTP request. It is also above the special header "Upgrade: WebSocket". The client requests to upgrade the connection to WebSocket protocol. An example of a request to establish a WebSocket connection is illustrated by Wang et al. (2013) as [11]:

```
GET / echo HTTP/1.1
Host: echo.websocket.org
Upgrade: websocket
Connection:  Upgrade
Origin: http://www.websocket.org
Sec-WebSocket-Key: 7+C600xYybOv2zmJ
69RQsw==
Sec-WebSocket-Version: 13
```

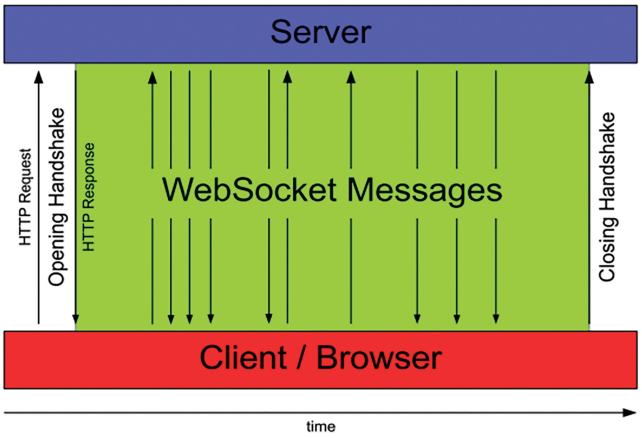

Figure 6 Start WebSocket connection [11].

The server responds to and accepts the handshake and the client is allowed to handle the HTTP on the WebSocket port. The connection remains till the acknowledgment exchange begins. It happens on both sides, and in case of any disconnection, they receive a numeric cod that holds the cause of disconnection. When there is a connection, the server and the client are able to send messages simultaneously. Unlike HTTP, if any changes happen in the server, it sends the changed information directly to the client. As shown in Figure 6, when the connection is made, both sides can share messages anytime with any requests or responses. When the server accepts and agrees to advance the connection, it sends a response like the one below which is taken from Wang et al. (2013) [5, 11].

```
HTTP/1.1 101 Switching Protocols
Upgrade: websocket
Connection:  Upgrade
Sec-WebSocket-Accept: fYoqiH14DgI+5 ylEMw
M2sOLzOi0
Date: Wed, 12 Jun 2013 01:33:37 GMT
Server: Kaazing Gateway
```

When the server does not match with WebSocket, they still can understand the handshake because of the HTTP header and reply to it with a negative response.

The messages hold UTF-8 encoded text or binary information. The header of the message is designed to be short to avoid overhead and decrease bandwidth consumption by breaking up a message to several frames and sending them serially. The strategy is beneficial for big messages. WebSocket connection can be coded or Decoded using Transport Layer Security (TLS) precisely in the same way as HTTP connections. They answer to the uniform resource identifier (URL), address the same ports of http/https (80/443), and make the firewalls and the proxies easy to handle [11].

Furthermore, WebSocket interface in web applications is Application Programming Interface (API), which is standardized by W3C in September 2013 as it was recommended by Hickson in 2012. API handles the following four events:

- on open: connection opened
- on message: message received
- on error: error occurred
- on close: connection closed

API is simple to use, and it can hold and support full functionality. Whereas API protocol supports extra functions such as sending and handling frames separately, it is an optional in the standardized WebSocket API [4, 11].

3 Conclusion & Future Work

This article provides valuable information on security features of data transmission protocols. The protocols are from the application layer and are widely used to handle the communication. The most important elements in executing IoT on dissimilar applications are: Security, Privacy and Certainty. As IoT is developing every day, its security has to be advanced for the sake of reliable data transference through billions of networks. In this article, I focused on data transmission protocols such as MQTT, CAOP, AMQP and WebSocket that are used in different IoT applications. Table 1 shows an overview of their key differences from one another. Though nearly all of them support a diverse range of applications, they have different performance and security features that can be used to improve the IoT work because based on fundamental network circumstances, the person can decide which protocol to use.

Table 1 Summary of data protocols [4, 5, 7, 12]

Protocols Futures						
Data Protocols	Security	Architecture	Protocols	QOS	Message Size	Dependability
MQTT	TLS & SSL	Publish & Subscribe	TCP	Yes	2	Dependable
CAOP	DTLS	Request & Response	UDP	Yes	4	Non-dependable
AMOP	TLS & SSL	Publish & Subscribe	TCP	Yes	8	Dependable Highly
WEBSOCKET	TLS & SSL	Client & Server Publish & Subscribe	TCP	No	2	-

References

[1] Razzaq, M. A., Gill, S. H., Qureshi, M. A., and Ullah, S. (2017). Security Issues in the Internet of Things (IoT): A Comprehensive Study. *Int. J. Adv. Comput. Sci. Appl. (IJACSA)*, 8(6), 383–388.

[2] Zhongchao, W., Ligang, H., Baojun, T., Wensi, W., and Jinhui, W. (2017). Design and verification of a novel IoT node protocol. In *13th IEEE International Conference on Electronic Measurement & Instruments (ICEMI)*, 2017 (pp. 201–205). IEEE.

[3] Alkuhlani, A., and Thorat, S. B. (2015). Internet of Things (IOT) standards, protocols and security issues. *Int. J. Adv. Res. Comput. Commun. Eng.*, 4(11), 491–495.

[4] Thangavel, D., Ma, X., Valera, A., Tan, H. X., and Tan, C. K. Y. (2014). Performance evaluation of MQTT and CoAP via a common middleware. In *IEEE 9th International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)*, (pp. 1–6). IEEE.

[5] Karagiannis, V., Chatzimisios, P., Vazquez-Gallego, F., and Alonso-Zarate, J. (2015). A survey on application layer protocols for the internet of things. Transaction on IoT and Cloud Computing, 3(1), 11–17.

[6] Yassein, M. B., Shatnawi, M. Q., Aljwarneh, S., and Al-Hatmi, R. (2017). Internet of Things: Survey and open issues of MQTT protocol. In *International Conference on Engineering & MIS (ICEMIS)*, (pp. 1–6). IEEE.

[7] Kaur, J., and Kaur, K. (2017). Internet of Things: a review on technologies, architecture, challenges, applications, future trends. *Int. J. Comput. Netw. Inf. Sec.*, 9(4), 57.

[8] Al-Fuqaha, A., Guizani, M., Mohammadi, M., Aledhari, M., and Ayyash, M. (2015). Internet of things: A survey on enabling technologies, protocols, and applications. *IEEE Communications Surveys & Tutorials*, 17(4), 2347–2376.

[9] Pooja, S., Uday, D. V., Nagesh, U. B., and Talekar, S. G. (2017). Application of MQTT protocol for real time weather monitoring and precision farming. In *International Conference on Electrical, Electronics, Communication, Computer, and Optimization Techniques (ICEECCOT)*, (pp. 1–6). IEEE.

[10] Yassein, M. B., Shatnawi, M. Q., and Al-Zoubi, D. (2016). "Application layer protocols for the Internet of Things: A survey," In *International Conference on Engineering & MIS (ICEMIS).*

[11] Srinivasan, L., Scharnagl, J., and Schilling, K. (2013). Analysis of websockets as the new age protocol for remote robot tele-operation. *IFAC Proceedings Volumes,* 46(1), 83–88.

[12] Swamy, S. N., Jadhav, D., and Kulkarni, N. (2017). Security threats in the application layer in IOT applications. In *International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*, (pp. 477–480). IEEE.

Biographies

Abdullah Ahmed Bahashwan received his bachelor degree in Computer Application from Osmania university Hyderabad India, in 2012. Currently, he doing master in science Internet Engineer at National Advanced IPv6 Centre (NAv6), Universiti Sains Malaysia (USM). His research interest includes Computer Networks, Internet Communication Protocols (IPv6), Network Security, and Internet of Thing.

Selvakumar Manickam is a senior lecturer and researcher at National Advanced IPv6 Centre (NAv6), Universiti Sains Malaysia. He has authored and co-authored almost 130 articles in journals, conference proceedings and book reviews. He has graduated 7 PhDs and completed many Masters and undergraduate thesis/dissertation. He has given several key note speeches as well as dozens of invited lectures and workshops at conferences, international universities and for industry. He has given talks on Internet Security, Internet of Things, Industry 4.0, IPv6, Green ICT and Open Source technologies at various organizations and seminars. Currently, he is involved a number of key clusters under National Advanced IPv6 Centre, namely Internet Security, Cloud Computing and Internet of Things (IoT), with focus on conducting scientific research in these respective areas. As of date, Dr. Selva has garnered funding in the form of grants and industrial projects worth almost RM2 million to date. He is also involved as committee member or representative at various technical forums locally and globally. He also lectures in various Computer Science and IT courses which includes development of new courseware in tandem with current technology trend. Apart from that, as senior trainer, he is actively involved in conducting professional courses on Cybersecurity, Cloud Computing, SDN, IPv6, and other networking-related technologies. He also codes in Android, Java, .net, C, PHP and Python in relation to the R&D work and projects.