Indonesian Journal of Electrical Engineering and Computer Science
Vol. 21, No. 3, March 2021, pp. 1663~1672
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v21.i3.pp1663-1672

Survey on IoT application layer protocols

Harth Ghassan Hamid, Zainab T. Alisa
Department of Electrical Engineering, University of Baghdad, Iraq

Article Info

Article history:

Received Apr 13, 2020
Revised Sep 14, 2020
Accepted Oct 4, 2020

Keywords:

Application protocols
Internet of things
IoT challenges

ABSTRACT

The constant evolution in internet technology has made. The internet of things (IoT) to be the center of research fields in computer engineering. This review paper discusses the choice of an application layer protocol in an IoT system integration so first, the paper briefly defines the potential protocols. After that, it opens up a comparison between these protocols according to how they manage their overhead and messages which affects traffic management and thus starts the discussion. The main contribution of this work is the simplification of comparison between session layer protocols in the benefit of IoT applications exclusively. IoT system Standards and platforms are being improved constantly. IoT enables application devices to connect and coordinate their tacks, such applications like healthcare, smart home, and industrial automation. Several protocols have been discussed to provide effective communication for resource-limited devices. However, their traffic management is still a field for researches, to find the optimal protocol choice for different situations. The review collects the results of other works that experimentally compared application layer protocols in the IoT environment and presents the graphical and tabular compression. Finally, the conclusion summarize the choice in different applications.

Corresponding Author:

Harth Ghassan Hamid
Department of Electrical Engineering
University of Baghdad, Baghdad, Iraq
Email: h.hamid0902@coeng.ubaghdad.edu.iq

1. INTRODUCTION

The Internet of things (IoT) is obtaining high interest in both industry and research fields. IoT transform objects/devices from being passively observering to become smart objects/devices, which are usually limited in resource and have the ability to communicate, compute, and make critical decisions [1]. And the number of sensors pervading our everyday life, in smartphones, cars, and buildings, is rapidly increasing [2]. It has been more than sixteen years since the Internet of Things term has been introduced to the public. But still, no standard IoT architecture has been clearly defined and no common agreement to defining a protocol for all IoT modules [3]. The application running on IoT is responsible for finding other nodes, efficient computing, information analytics, and communication of machine-to-machine (M2M) [4].

The requirement for specialized protocols of communication is to overcome the IoT challenges. Standardization has to process a complete and efficient application protocol stack for these resource-limited devices [5]. IoT systems involves a lot of connected devices (sensors, gateways, servers/brokers) where the data is being sent by the sensors and collected by the gateway then is forwarded to servers/brokers for processing then send it to the client that requested this data. The addition of sensors devices into the network require IP compatible protocols with efficient bandwidth (BW) to work with limited resource hardware [6].

There is a lot of performance impairment in regards to usage, BW and battery lifetime in sensor nodes because there is no optimized application layer protocol. IoT services an important role in a lot of

applications like healthcare, smart homes/cities transportation, automation, and emergency services [7]. IoT application uses application layer protocols like MQTT, CoAP, AMQP, XMPP, and HTTP for transferring requests/responses between nodes like in Figure 1 that shows the IoT stack [8]. All these protocols are considered to be a real-time publish-subscribe IoT protocol. IoT protocols are a subject of a lot of studies in the research community [9]. Developers are using available technologies to optimize and enhance communication in the IoT system to transmit the data from IoT devices to the cloud, like in Figure 2. Protocols are different in the performance as the payload size changes. The need for developing a system that transmits payload efficiently is increasing [10].

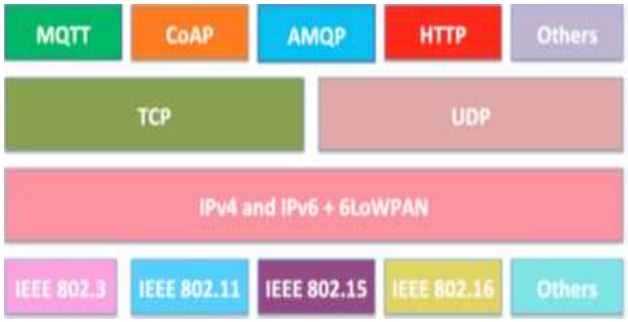

Figure 1. Protocol stack for IoT systems [11]

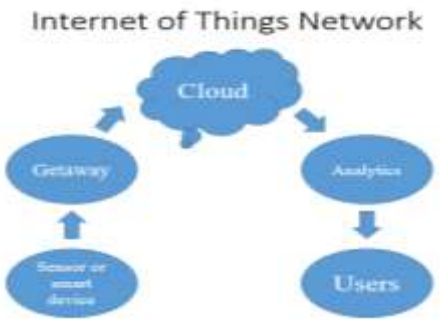

Figure 2. IoT major components

2. RELATED WORKS

A lot of new standardized protocols are being implemented every year, therefor survey papers are continuously being updated to give an insight to the different IoT standardization aspects. This work acts as an updated simplified compression and interpretation of the results from the works in Table 1.

In this survey, we aim to present compression of the rising protocols to update/extend the work in [11]. The work of [12] discusses MQTT, CoAP, and XMPP only and found that CoAP performs best in low server utilization. In [13], summarize the protocols used for IoT/Fog/Cloud computing and their challenges. This enables the discussion of more standards, in addition to the recently offered in IETF, also the ones that will be used in close future. In [14], summarize the important protocols offered by standards organizations. It also present various IoT challenges discussions. In [15], shows that the protocol efficiency does not change by changing the network setting, from LAN to an IoT network, but only increase the average-RTT.

Table 1. Related works

Author	Title
Nitin Naik	Choice of effective messaging protocols for IoT systems: MQTT, CaAP, AMQP and HTTP [11]
Paridhika Kayal	A Comparison of IoT application layer protocols Through a smart parking implementtation [12]
Jasenka Dizdarević	A survey of Communication protocols for internet of things and related challenges of fog and cloud computing integration [13]
Tara Salman	A survey of protocols and standards for internet of things [14]
Stefan Mijovic	Comparing application layer protocols for the internet of things via experimentation [15]

3. IOT CHALLENGES

IoT application is still hard task to develop despite the quantity of standards available due to number of challenges which will be briefly discussed in the next:

a) Mobility: the IoT system states is dynamic that mean devices are moving freely and their IP addresses and connections are being changed, which add a huge challenge for routing protocols. These protocols need to reconstruct the DODAG (destination-oriented-directed-acyclic-graph) anytime there is movement in the network nodes, which adds a lot of overhead. Also, there is the possibility of service provider changing which added a layer of complexity because of service interruption in real time [16].

b) Scalability, availability, and reliability: scalability is the ability to added new devices/services to the system without the need for complex reconfiguration or lowering performance. In IoT systems this

creates an issue due the huge number of nodes that are supported each with different bandwidth, storage, memory and processing power. Scalability/availability need to be deployed in the IoT framework. Like cloud based IoT systems that offer support for scaling the IoT network by adding more storage and processing power as needed. Which opens up new field of designing smooth IoT framework to meet global needs. Resource availability is another challenge to authentic devices no matter what is their location and time of necessity. Small IoT networks are attached to the global IoT platforms timely to utilize their services and resources. Due to the use of various data transmission channels like satellite communication, some services and availability of resources might be interrupted. Therefore, an independent and reliable data channel is required for uninterrupted availability of services [17].

c) Management, interoperability, and standard issues: although there is several protocols for device management they cannot be applied to all IoT systems, hence it is challenging to present FCAPS faults, configuration, and accounting management for connected devices [18]. Interoperability is the information exchange in hardware feasibility. It arises an issue due to IoT systems different technology and heterogeneous nature. Interoperability levels are semantic, syntactic, technical, and organizational. Several protocols are presented to IoT systems for enhancing interoperability to ensure heterogeneous devices communications, also merging different IoT platforms [16]. These solutions can be virtual networks, overlay based, adapters, gateways based, a service-oriented architecture based, etc.

d) Power, cost and complexity: due to absence of power management technologies in small, resource-restricted devices this creates a challenge for IoT systems. Which is a critical issue in some IoT applications where devices battery are difficult to change. There is solutions like using energy from motion or other sources and transform it to the devices storage. However this method is still weak for satisfying small devices power needs [17]. Reducing cost and complexity is challenge for IoT applications to be public usage, despite the fact that IoT devices (sensors, smart transducers, etc.) it is still high cost when working with full IoT application and the integration of different protocols is very complex [18].

e) Security/privacy issues: due to cyber-attacks risks and threats security is the most IoT challenging issue, because of insufficient authentication/authorization, firmware, software insecurity, week transport layer encryption and web interfaces. This issue is an important parameter in IoT system development confidence. To stop attacks and threats security measures need to be embedded in all IoT architecture layers. Number of protocols are already developed/deployed like secure-socket-layer (SSL) and datagram transport-layer-security (DTLS) which are cryptographic protocols implemented in transport and application layer to solve IoT system security issues. However, it becomes more difficult with some IoT systems (wireless applications for example) that need different approaches for security confirmation which will require the deployment of malicious-detect-actions and self-recovery [16]. While on the other hand piracy issues might prevent users from using IoT systems comfortably. Because of different privacy policies of various IoT devices and service providers, therefor standardized authorization/authentication are needed for secure network communications before transmitting data [19].

f) Quality-of-service (QoS): important aspect of IoT QoS is an evaluation metric to the standard performance, quality, and efficiency in IoT systems (devices and architecture). Important metrics are service-time, availability, reliability, security, cost, and power consumption. An optimal IoT system must satisfy the needed requirements of QoS metrics which are predefined by developer or specified by users. There is number of methods to deploy QoS. However, there is trade off with performance, which will require quality models to overcome it by presenting range of quality factors that is sufficient for IoT system QoS requirements [17].

g) Traffic/queuing: with the exponential rise of connected devices increases the challenge of scalability that is needed on different levels. There is need for proper collection of protocols and access-technologies to support the needed flexibility in the architecture (dynamically placing of gateways/brokers). Although those same devices add performance impairment. To achieve the required scalability there is need for adaptive load balancing mechanisms development that are dimensioned properly [19]. The IoT traffic pattern model need to be evaluated to achieve scalability with increment of devices numbers. In IoT systems sensors often send data in deterministic periodic manner. And as the sensor network become large in numbers the aggregated traffic will be considered a superposition of deterministic process (poisson process model), which largely simplifies the data arrival process model. However it will added error due to the passion approximation. In discrete-time networks where there is a constraint on which service nodes will be active at any time, the max-weight scheduling algorithm chooses a service policy to grant optimal throughput in the case that each packet visits only a single server service node. A networking scheduler must choose a queuing algorithm, which affects the characteristics of the bigger network, which is an arbiter on a node in the packet-switching

communication network. It manages the sequence of network packets in transmit and receive queues of the network interface controller [13]. The network scheduler decides which network packet to forward next. The network scheduler is related to a queuing system, arranging the network packets temporarily until they are being transmitted. Systems may have one or multiple queues in which case each may hold the packets of one flow, classification, or priority. An automatic process that categorizes network traffic according to various parameters into several traffic classes. Each traffic class can be treated differently to differentiate the service implied for the data generator and endpoint devices. A priority queue can be implemented to improve network performance. In the next sections, we will discuss the different IoT application layer protocols to help the development choice in designing an IoT application [20].

4. IOT APPLICATIONS

Within industrial use cases, computers were introduced over the last decades, to fulfill specific requirements, such as meeting real-time response times or operating reliably in very rough environments. Now with IoT, this role can be improved by Networks of devices, processes, and services constantly exchange data with each other and enable the cooperation for a common task [21]. Such services are:

a) IoT in transportation: IoT devices collect, stores and process faster way of traffic information, which helps in optimizing traffic problems. Future applications may include location-sensitive billing, location-based advertising, and information services such as navigation, points of interest, etc. [22].
b) IoT in education: The usefulness of the internet of things in education was concluded by improving and developing education and the extent of their relevance in universities and its application through the work of smart class using modern techniques in classrooms and using smart Laboratories to conduct experiments better and facilitate tests and use of devices to facilitate student communication with the teacher and other students and scientific material [23].
c) IoT in healthcare: Although IoT-backed smart healthcare technologies can enhance income and enhance the quality of lives, security and safety is a concern as well. Additional measures should be taken to deal with threats and to secure potential information at the ends of both the customer and the developer. Thus, this dynamically increasing industry's vision and long-term achievement lie in the synergy between scientists, healthcare practitioners, and the people [24].
d) IoT in smart homes: a home automation system uses the technology of IoT for the screening and controlling of the electrical and electronic appliances at home from any remote area by essentially utilizing a smartphone. There has been rising interest in a secure framework that must be tried, true and fast in reaction to the ventures and organization [25].

5. APPLICATION LAYER PROTOCOLS

5.1. Hyper-text-transfer protocol (HTTP)

It is a protocol that is responsible for collaboration and distribution of information in the system. HTTP is the world wide web (global network) information communication foundation, which depends on hyper-text-documents that contain hyper-links that provide the user with access to other resources. Communication between clients and servers is done by sending HTTP requests and receiving HTTP Responses. It is the main client/server model used in the current internet web, also the most compatible with current infrastructure used by developers daily [26].

It works as a request/response model in the client/server computing system. Where HTTP-request message is sent by client to the server, which then present the requested resource (HTML files or others) or do a specific function, and sends the client a HTTP-response message, which contains data status about the request with the requested content in the message body [14]. The protocol is intended to enhance the communication of clients/servers by using intermediate network elements. For example High traffic websites use cache-servers to get the content on behalf of upstream servers which improves response [27].

HTTP is created inside the internet protocol suite framework. It is a reliable transport layer protocol that usually uses transmission control protocol (TCP), which offers reliable delivery of huge data and that is a connection advantage if there is no strict limits of latency, but adds a challenge in resource-restricted nodes. Because those nodes constantly sends small amounts of data and with each there is need for TCP connection that creates unnecessary overhead and takes time [28]. However, HTTP is able to use unreliable protocols like the user-datagram protocol (UDP). HTTP is an extensible protocol that is simple to use. The client-server structure, combined with the ability to easily add headers, allows HTTP to advance together with the extended capabilities of the internet web. The message format, Communication between clients and servers is done by requests and responses:

a) HTTP request is sent by client to the web
b) A web server receives the request
c) The server runs an application to process the request
d) HTTP response is sent by the server to the browser
e) Response is received by client

REST guideline has been used with HTTP which helps in developing web services according to architectural style and the interaction between various. There is effort to implement the RESTful-web service in IoT systems because of its success, using HTTP and REST. This mixture is admirable because the nodes can make their information state available easily thanks to ways of standardization (read, update, and delete data). As of QoS, this protocol adds no additional options, it depends on the guaranteed successful delivery provided by using TCP if the connection is not interrupted. The security in this protocol uses TLS to enable secure and encrypted channel that creating the secure HTTP which is called HTTPs [11].

5.2. Message-queuing-telemetry-transport protocol (MQTT)

It is a lightweight (M2M) connectivity from open standards of OASIS/ISO, a publish/subscribe protocol that manages the transportation of messages between nodes, which is an alternative to the traditional client/server protocols, where a client connects directly with an endpoint [11, 14]. It uses TCP-IP to run over; although all protocols that provides lossless, bi-directional, and ordered can work with MQTT. It enables the users and applications to connect at one end, also at the other end communications, and network. It was created to reach remote locations where the bandwidth of the network is limited. It is suited for IoT communications thanks to its small headers and simplicity [29]. It contains three main components: Broker and several nodes (publishers, subscribers) as we can see in Figure 3. The role of MQTT broker is a server which collects data from publisher nodes and forward them to subscriber nodes. MQTT nodes (clients) are any devices that uses the MQTT library and able to connect with the broker in the network. For a client to receive messages about a specific topic, it has to subscribe to it. Also it is possible that number of clients all subscribe to the same topic and all receive data about it whenever available from the broker [30].

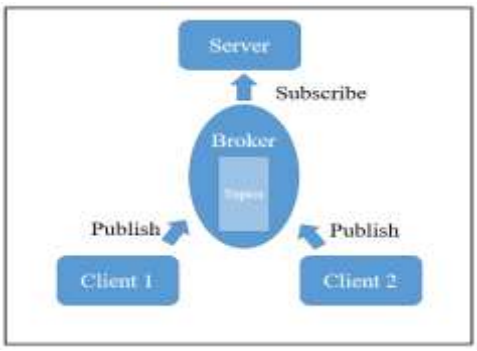

Figure 3. MQTT aarchitecture

The MQTT connection is between broker and two clients/nodes where the center component is the broker that receives messages from publisher nodes and according to topic filtering deliver those messages to subscriber nodes. It is needed to install the MQTT-broker library in the device for it to act as a broker, as for client nodes they require installation of MQTT-client libraries. The publisher client adds a labeled topic to the broker. One MQTT publishes the message to a set of topics [31]. This data will be sent/published to the broker that can store it in its database temporally [32]. The subscribers who are subscribed to this topic will send messages to check for updates [28]. The MQTT-control messages are minimal and small as of 2 B of information. But it can hold up to 256 MB when needed. The messages used in MQTT are different in types like the ones used to control/disconnect a node from a broker, from forwarding data, data receipt acknowledgment, to monitor the connection. Message types are connected, disconnect and publish. Sent data supports SSL/TLS encryption [33]. MQTT uses different modules for message exchange known as quality of service (QoS) profiles:
QoS0: once at most: the data/message is sent once and no additional delivery acknowledgment.
QoS1: once at least: unless delivery acknowledgment is received, data/message will keep being resent.
QoS2: once exactly: both nodes sender/receiver will start in two level handshake to grant that one copy of the data/message is received only.

5.3. Constrained-application protocol (CoAp)

It is an application layer protocol specialized for constrained devices, it enables the communication of nodes to Internet, such as wireless sensor network nodes. It is used in transferring data from/to clients/servers over the Internet. It is intended to be used between nodes on the same level of constrained network (low power, loss networks, etc.), nodes on different constrained networks and lastly between constrained nodes and general Internet devices. Its design is intended for (M2M) applications where it has very low overhead, and simplicity. CoAP can work with most devices that support user datagram protocol (UDP). From an architecture view, the end nodes (like sensors) will get CoAP server added to them [34].

The controller should have the CoAP client installed, where it will manage several end nodes. CoAP functions as a sort of HTTP for restricted devices, enabling equipment such as sensors or actuators to communicate on the IoT. CoAP connection example steps in Figure 4. The protocol is designed for reliability in restricted bandwidth and high congestion through its low network overhead and low power consumption. CoAP also supports networks with billions of nodes. For security, default DTLS parameters are chosen as an equivalent of 3072 bit RSA keys [35]. CoAP is a client-server IoT protocol that follows request/response method like what happens in HTTP. CoAP is intended to solve the problems of REST so it will give IoT applications the ability to use RESTful services without violating their restrictions. It uses UDP, with a lightweight mechanism which enables reliability [32].

A simple binary format is how CoAP messages are encoded. Simple generation of which offers saving them without needing extra RAM in the constrained nodes themselves. It contain four types of messaging: confirmable and non-confirmable which represent the unreliable and reliable transmissions, respectively. Piggyback and separate is used for client/server communication, in which the server response is sent immediately after receiving the message, within the acknowledgment message or not respectively [14].

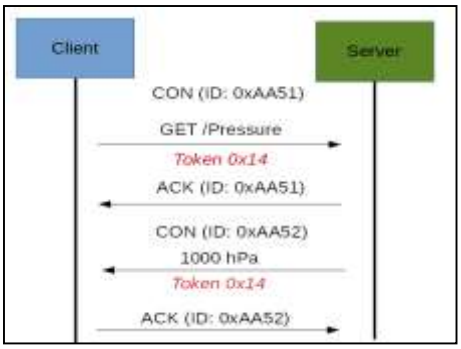

Figure 4. CoAP cconnection steps

5.4. Advanced-message-queuing protocol (AMQP)

It is an OASIS open standard binary wire level message oriented middleware application layer protocol. It is a replacement for existing propriety messaging middleware. Its features are queuing, orientation, routing of messages also reliability and security (SASL/TLS). AMQP supports a wide variety of messaging applications and communication patterns efficiently. In AMQP implementations of different service providers is interoperable because it highly depends on the messaging provider and client. Because AMQP is wire level protocol the data format description is forwarded as a stream of bytes throughout the network. Therefore, tools that can manage messages will confirm the ability of data format interoperation to other tools regardless of the language of implementation [36].

The header of AMQP is a delivery related annotations standard set of which can be indicated or requested for a message, the header includes priority, time to live, and durability. The structure of message is a standard optional list of properties of specific applications (user id, message id, creation time, replay, subject, coordination id, etc.), and a body (application data according to AMQP). AMQP is defined for messaging abilities, it enables interoperability with intermediaries messaging (like Brokers, Bridges, and others) in large rich networks, and also it is used in simple systems of peer-to-peer. Although the framework covers the basic behaviors, it allows for extensions so it can be further standardized. Like MQTT AMQP uses publish/subscribe TCP architecture. But it differs that the broker is subdivided into two components: queues and exchange as we can see in Figure 5. The exchange part is the one responsible for collecting and redirecting of publisher messages according to set of coded roles. Queues act as topics that subscribers connect to get the data whenever available [37].

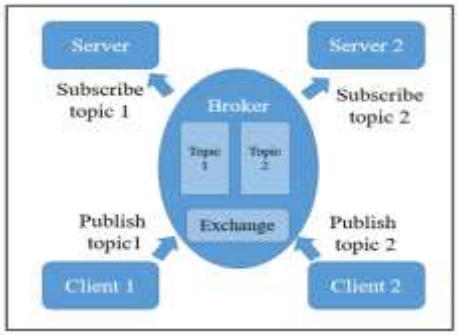

Figure 5. AMQP Architecture

5.5. Extensible-messaging protocol (XMPP)

It is a XML based communication protocol built for message-oriented middleware, which supports a wide range of applications like presence and collaboration of instant messaging. It provides close to real-time exchange of extensible and structured data between any amounts of network nodes. It is designed to be extensible and instant messaging between applications. The protocol is used for streaming XML elements over a network to exchange messages and presence data in close to real-time. Also, it supports publish-subscribe systems running over TCP, like VoIP signaling, video, file transfer, IoT applications such as social services and smart grid [38].

The definition of XMPP is an open standard that uses an open systems approach for development, by which offers the ability that anyone can implement an XMPP service and interoperate it with other organizations 'implementations. Due to XMPP being an open protocol, design of implementations uses any software license and as many service as needed, client and library implementations are available for free. It functions like the HTTP GET/POST method. XMPP's strengths are decentralization, Open standards, Security, Flexibility. XMPP weaknesses are Text-based communication, no Quality of Service, In-band binary data transfer is limited [39]. XMPP provides:

a) Send and receive messages with other users.
b) Check and share presence status
c) Manage subscriptions to and from other users.
d) Manage contact list
e) Block communications (receive messages, sharing presence status, etc.) to specific users.

XMPP connection example in Figure 6. There is an open research to modify a better XMPP for IoT. A publish/subscribe lightweight schema was implemented for resource-constrained applications, which improves the current version of the protocol.

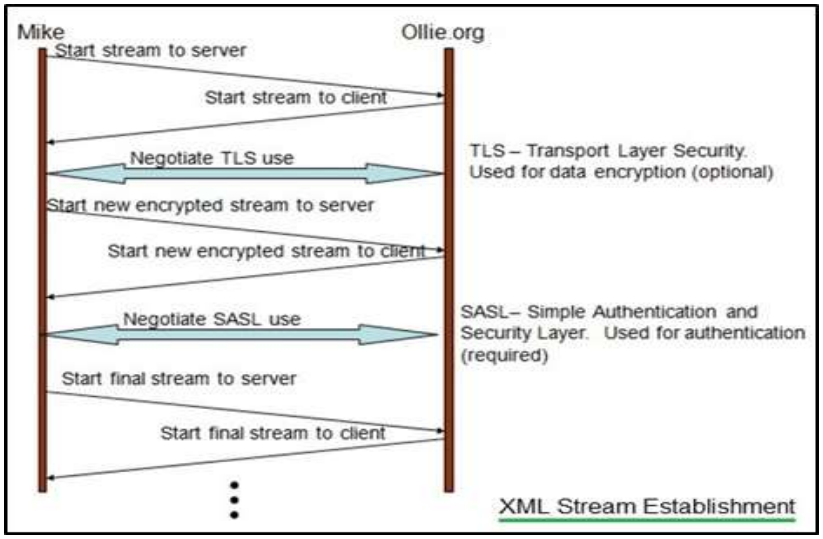

Figure 6. XMPP Connection messages example

6. COMPARISON

Several application layer protocols for IoT have been studied and discussed. The choice among these protocols depends on the needs of specific application. For example, if an application has been implemented with XML, and able to accept overhead in its headers, XMPP could be the best option to choose among session layer protocols. Else wise, if the application is overhead and power-sensitive, then choosing MQTT would be the better option [40, 41]. Due to low overhead and power consumption MQTT is the popular one in IoT. However, that will require an additional broker implementation.

CoAP is the best and most suitable if the application needs REST functionality [42]. Comparison summary of the discussed protocols is presented in Table 2. Figure 7 shows the comparison based on overhead and message size. The illustration shows that XMPP has the highest message size and overhead, while other protocols has lower with CoAP the lowest message size and overhead. MQTT, AMQP, XMPP and HTTP uses all TCP connection overheads for connection establishment and closing. However, MQTT is lightweight there for it has the least header size of 2-byte per message but its requirement of TCP connection will add size the overall overhead, so the whole message size will increase [43, 44]. CoAP uses UDP which does not increase connection overheads as it works on basis of fire and forget. This will reduce the overall overhead considerably (message size).

Although AMQP is also a binary lightweight protocol, it has increased overhead and message size because of its support for security, reliability, provisioning, and interoperability. Finally, XMPP is the most heavyweight protocol. It needs higher overhead and message size among all as it was not originally designed for the IoT [45]. As an example of suitable use of the different application protocols discussed previously Figure 8 shows an IoT network making use of the different protocols in one system [46].

Table 2. Application layer protocol comparison

Criteria	HTTP	MQTT	XMPP	AMQP	CoAP
Year	1997	1999	2000	2003	2010
Architecture	Client/Server	Client/Broker	Client/server	Client/Broker or Client/Server	Client/Server orClient/Broker
Abstraction	Request/Response	Publish/Subscribe	Publish/Subscribe or Request/Response	Publish/Subscribe or Request/Response	Request/Response orPublish/Subscribe
Header Size	Undefined	2 Byte	Undefined	8 Byte	4 Byte
Message Size	Large and Undefined (depends on the web server or the programming technology)	Small and Undefined (up to 256 MB maximum size)	Large and Undefined (depends on the web server or the programming technology	Negotiable and Undefined	Small and Undefined (normally small to fit in single IP datagram)
Semantics/ Methods	Get, Post, Head, Put, Patch, Options, Connect, Delete	Connect, Disconnected, Publish, Subscribe, Unsubscribe, Close	Chat, error, group chat, headline, normal	Consume, Deliver, Publish, Get, Select, Ask, Delete, Nack, Recover, Reject, Open, Close	Get, Post, Put, Delete
Cache and Proxy Support	Yes	Partial	Yes	Yes	Yes
Quality of Service (QoS)/ Reliability	Limited (via Transport Protocol - TCP)	QoS 0 - At most once (Fire-and-Forget), QoS 1 - At least once, QoS 2 - Exactly once	None (could be done by extension)	Settle Format (similar to At most once) or Unsettle Format (similar to At least once)	Confirmable Message (similar to At most once) or Non-confirmable Message (similar to At least once)
Standards	IETF and W3C	OASIS, Eclipse Foundations	IETF, XSF Bylaws and XEP-0001	OASIS, ISO/IEC	IETF, Eclipse Foundation
Transport protocol	TCP	TCP (MQTT-SN can use UDP)	TCP	TCP, SCTP	UDP, SCTP
Security	TLS/SASL	TLS/SSL	TLS/SASL OMEMO	TLS/SSL, IPSec, SASL	DTLS, IPSec
Default Port	80/ 443 (TLS/SSL)	1883/ 8883 (TLS/SSL)	5223 (TLS/SSL)	5671 (TLS/SSL), 5672	5683 (UDP Port)/ 5684(DLTS)
Encoding Format	Text	Binary	Text	Binary	Binary

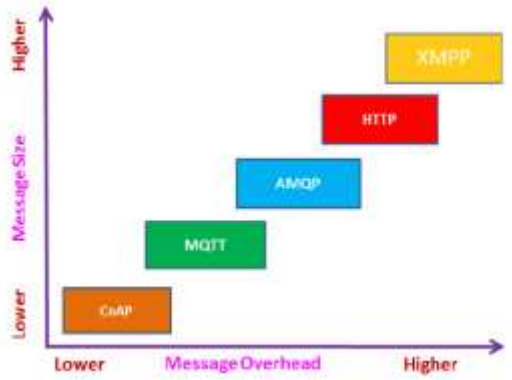

Figure 7. Message size vs message overhead

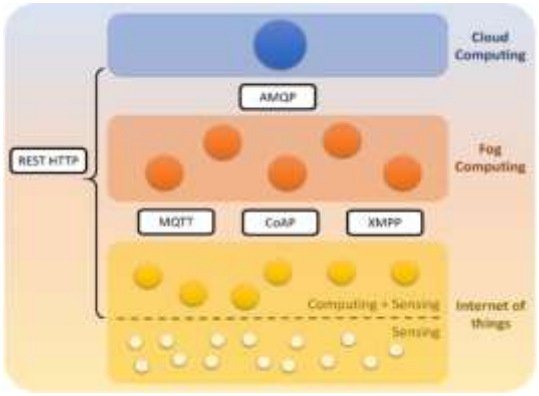

Figure 8. Application protocols in IoT networks [12]

7. CONCLUSION

The main goal of the current study was to provide a comprehensive survey of IoT application layer protocols to help developers by giving an insight of IoT solutions and alternatives for applications that's also standardized by IEFT, IEEE, and others. This study has found that IoT systems take into account different requirements. There is no one protocol alone will satisfy the development of entire communication in the system, beginning from resource-restricted devices to the cloud to the server nodes. The following is a summary of the conclusions. RESTful HTTP is suitable in cloud computing systems because there is no issue of battery consumption or constrained communication, but the standard is still evolving and has challenges of short-term interoperability. MQTT has proved to have best performance in IoT system due to its simple configurations and stability. And between those message queue protocols CoAP performs best in connecting them. XMPP is best used in applications that supports multi-threading, because of lower server utilization. Also, adds horizontal scalability which is missing in CoAP and MQTT as they are a single point of failure. The current findings add to a growing body of literature on IoT system design and application protocols used in it. Developer team needs a lot of training when developing application protocols other than HTTP. Due to the overhead they bring features, like privacy and security, need to be analyzed more, to find more optimal solutions, which creates challenges and also open exciting opportunities for novel architectures that combine cloud computing and IoT systems to enhance performance of future applications.

REFERENCES

[1] J. Huang, "Managing the Internet of Things," *The Institution of Engineering and Technology*, 2016.
[2] A. Zanella, et al., "Internet of Things for Smart Cities," *IEEE Internet of Things Journal*, vol. 1, no. 1, 2014.
[3] S. Cirani, "Internet of Things," *JohnWiley & Sons Ltd*, 2019.
[4] J. A. Cedillo, "Internet Prospective Study," in *Bulletin of Electrical Engineering and Informatics (BEEI)*, 2017.
[5] O. Hahm, "Operating systems for low-end devices in the internet of things: A survey," *IEEE Internet of Things Journal*, 2016.
[6] R. van Kranenburg, "The internet of things," *The insittutr of Network Cultures*, 2008.
[7] S. Mann, "The Wireless Application Protocol (WAP)," *John Wiley & Sons, Inc.*, 2000.
[8] A. Al-Fuqaha, et al., "Internet of things: A survey on enabling technologies, protocols and applications," in *IEEE Communications Surveys Tutorials*, vol. 17, no. 4, pp. 2347-2376, 2015.
[9] P. Sethi and S. R. Sarangi, "Internet of things: Architecture, protocols, and applications," 2017.
[10] A. Foster, "Messaging Technologies for the Industrial Internet and the Internet of Things," 2014.
[11] N. Naik, "Choice of Effective Messaging Protocols for IoT Systems: MQTT, CoAP, AMQP and HTTP," *IEEE International Systems Engineering Symposium (ISSE)*, 2017.
[12] P. Kayal, "A Comparison of IoT application layer protocols in a smart parking implementation," 2017.
[13] J. Dizdarevic, "A Survey of Communication Protocols for Internet of Things and Related Challenges of Fog and Cloud Computing Integration," 2019.
[14] T. Salman and R. Jain, "A Survey of Protocols and Standards for Internet of Things," *Advanced Computing and Communications*, 2017.
[15] S. Mijovic, "Comparing Application Layer Protocols for the Internet of Things via Experimentation," *IEEE 2nd International Forum on Research and Technologies*, 2016.
[16] W. Shang, et al., "Challenges in iot networking via tcp/ip architecture," 2016.
[17] H. Baba, et al., "Problems in and among industries for the prompt realization of IoT and safety considerations," *IETF Draft*, 2020. Available: https://datatracker.ietf.org/doc/draft-baba-iot-problems/.
[18] N. Fernando and S. W. Loke, "Opportunistic Fog for IoT: Challenges and Opportunities," 2019.

[19] J. Granjal, et al., "Security for the internet of things: A survey of existing protocols and open research Issues," in *IEEE Communications Surveys Tutorials*, 2015.
[20] Z. Sheng, et al., "A survey on the IETF protocol suite for the internet of things: standards, challenges, and opportunities," in *IEEE Wireless Communications*, 2013.
[21] G. Fortino, "Internet of Things AtoZ," *John Wiley &Sons, Inc.*, 2018.
[22] H. V. Chand, "Survey on the Role of IoT in Intelligent Transportation System," 2018.
[23] R. S. Abd-Ali, "A survey: the role of the internet of things in the development of education," in *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, 2020.
[24] C. R. Srinivasan, "An IoT based SMART patient health monitoring system," in *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, 2020.
[25] M. H. Ali, "IoT based security system and intelligent home automation multi monitoring and control systems," in *International Journal of Robotics and Automation (IJRA)*, 2019.
[26] D. Gourley, "HTTP: The Definitive Guide," *O'Reilly Media, Inc.*, 2002.
[27] V. Karagiannis, et al., "A survey on application layer protocols for the internet of things," in *Transaction on IoT and Cloud Computing*, 2015.
[28] C. C. Goh and E. Kanagaraj, "IV-AQMS: HTTP and MQTT Protocol from a Realistic Testbed," *IEEE International Conference on Sensors and Nanotechnology*, 2019.
[29] OASIS, "MQTT Version 5," 2019. Available: https://docs.oasis-open.org/mqtt/mqtt/v5.0/mqtt-v5.0.html.
[30] "MQTT, the lightweight publish/subscribe messaging transport protocol," Available: http://mqtt.org/.
[31] G. C. Hillar, "MQTT Essentials - A Lightweight IoT Protocol," *Packt Publishing Ltd.*, 2017.
[32] D. Thangavel, et al., "Performance evaluation of MQTT and CoAP via a common middleware," in *IEEE Ninth International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)*, 2014.
[33] M. Singh, et al., "Secure mqtt for internet of things (iot)," in *Fifth International Conference on Communication Systems and Network Technologies (CSNT)*, 2015.
[34] "The Constrained Application Protocol (CoAP)." Available: https://tools.ietf.org/html/rfc7252.
[35] Z. Shelby, et al., "The Constrained Application Protocol (CoAP)," *IETF RFC 7252*, 2014. Available: http://www.ietf.org/rfc/rfc7252.txt.
[36] OASIS, "Oasis advanced message queuing protocol (amqp) version 1.0," 2012. Available: http://docs.oasisopen.org/amqp/core/v1.0/os/amqp-core-complete-v1.0-os.pdf.
[37] J. E. Luzuriaga, et al., "Testing AMQP protocol on unstable and mobile networks," in *Internet and Distributed Computing Systems*, 2014.
[38] P. Saint-Andre, "XMPP: The Definitive Guide," *O'Reilly Media, Inc.*, 2009.
[39] P. Saint-Andre, "Extensible Messaging and Presence Protocol (XMPP): Core," *IETF RFC 6120*, 2011. Available: https://tools.ietf.org/html/rfc6120.
[40] J. E. Luzuriaga, et al., "Handling mobility in iot applications using the mqtt protocol," in *2015 Internet Technologies and Applications (ITA)*, 2015.
[41] S. Bandyopadhyay, "Lightweight internet protocols for web enablement of sensors using constrained gateway devices," in *2013 International Conference on Computing, Networking and Communications (ICNC)*, 2013.
[42] N. De Caro, et al., "Comparison of two lightweight protocols for smartphone-based sensing," in *2013 IEEE 20th Symposium on Communications and Vehicular Technology in the Benelux (SCVT)*, 2013.
[43] D. Thangavel, et al., "Performance evaluation of MQTT and CoAP via a common middleware," in *2014 IEEE Ninth International Conference on Intelligent Sensors, Sensor Networks and Information Processing*, 2014.
[44] J. E. Luzuriaga, et al., "A comparative evaluation of AMQP and MQTT protocols over unstable and mobile networks," in *12th Annual IEEE Consumer Communications and Networking Conference*, 2015.
[45] S. S. N. M. Khoi, et al., "Irehmo: An efficient IoT-based remote health monitoring system for smart regions," 2015.
[46] J. Stansberry, "MQTT and CoAP: Underlying protocols for the IoT," 2015. Available: http://electronicdesign.com/iot/mqtt-and-coap-underlying-protocols-iot.