|| Volume 2 || Issue 3 || OCTOBER 2017 || ISSN (Online) 2456-0774 ||
INTERNATIONAL JOURNAL OF ADVANCE SCIENTIFIC RESEARCH
AND ENGINEERING TRENDS

Review on IOT- Based Home Automation

Mohammed Salim Khan[1], Prof. R. A. Auti[2], Prof. B. K. Patil[3]
PG Student, Computer Science & Engineering, Everest College of Engineering & Technology, Aurangabad, India[1]
Head of Department, Computer Science & Engineering, Everest College of Engineering & Technology, Aurangabad, India[2]
Assistant Professor, Computer Science & Engineering, Everest College of Engineering & Technology, Aurangabad, India[3]
salim.khan1686@gmail.com[1], cseroyal7@gmail.com[2], rajeshauti24@gmail.com[3]

Abstract – Internet of things (IOT) technologies is increases as a powerful domain. In which embedded devices and sensors can connect and interchange information over the Internet. IOT technologies can extend a development of communication protocols as well as sensors. Communication protocol is one of basic and important means to lightweight, low power sensor module, bandwidth consumption, battery lifetime and security. In this paper, present an idea for better improvement IOT based Home Automation System (HAS) with the help of communication protocol mechanism for IOT networks. In this paper discussed various technique and protocols for HAS. The proposed Message Queue Telemetry Transport (MQTT) protocol required low power, low bandwidth, battery life, security. The proposed protocol gives surety of message delivery to subscribers and publishers.

Keywords *– CoAP, HAS, IoT, Low Power, Low bandwidth, MQTT, QoS.*

I INTRODUCTION

Nowadays, The Internet of Things (IoT) is one of the most research topics. As user demands for communication between the home and outside world increases, the requirement for IoT technologies for various systems also increased. IoT has spread widely and used in different environments including homes, health care systems, aerospace and various transportations. IoT Technologies [1] have extended the development of communication protocols as well as sensor networks for home automation systems. The Internet of Things (IoT) is a recent communication prototype that visualizes a near future, in which the object of everyday life will be equipped with microcontrollers, transceivers for digital communication that will make them able to communicate with one another and with the clients, becoming an integral part of the Internet. additionally, by enabling easy access and interaction with a wide variety of devices such as, home appliances, surveillance cameras, monitoring sensors, actuators, displays, vehicles, and so on, the IoT will foster the development of a number of applications that make use of the potentially enormous amount and variety of data generated by such objects to provide new services to citizens, companies, and public administrations. This prototype finds application in many different domains, such as home automation, industrial automation, medical aids, mobile healthcare, elderly assistance, intelligent energy management and smart grids, automotive, traffic management, and many others [2].

Recently, home automation system platforms, which collect data from the sensors and devices using wireless technologies, have been developed with IoT technologies [1]. Some of major communication technologies used by today's home automation system include Bluetooth, WI-MAX, and Wireless LAN (Wi-Fi), ZigBee, and Global System for Mobile Communication (GSM) [3]. In addition home application usually requires accurate and concurrent time information in order to transmit their data in timely manner. For example fire alarm and intrusion warnings in home automation systems need their data to be transmitted quickly and reliably as possible. Hence, QoS message delivery between the nodes is so crucial that it can be severely affect the performance of home automation systems.

II LITERATURE SURVEY

In this section, review of the selected literature on Internet of Things (IOT) in different standards, protocols and their usage in different area using different application is mentioned.

Seung-Chul Son [1] paper addressed time synchronization techniques for low power sensor modules. A constrained application protocol (CoAP) was recently standardized for sensor networks by IETF and is becoming widely adopted for home automation systems by ETSI, OMA, and oneM2M. Due to network time protocol (NTP) limited computing resources, it is not applicable to home automation systems. This paper proposes a lightweight time synchronization algorithm for CoAP-based home automation system networks. The proposed scheme comparatively reduces network overhead because it only uses CoAP instead of the additional standards for time synchronization protocols. Another advantage is that it does not require an increase in performance, as experimental results indicate that the proposed scheme has a reasonable synchronization error when compared to NTP for existing distributed systems. Consequently, it is expected that the proposed CoAP-based time synchronization scheme can be extensively applied, not only in home automation systems but also in other applications, such as environmental monitoring, building and plant management, urban monitoring, disaster monitoring, etc.

Olaf Bergmann, Kai T. Hillmann, Stefanie Gerdes [4] in this paper, authors were discussed about basic design concept to interconnect CoAP and the proprietary FS20

protocol for home automation. Key aspects of our approach are the mapping of FS20 device addresses to path segments of CoAP URIs and to map FS20 commands to the four basic CoAP operations for creation, recovery, refresh and cancellation of assets, and the dynamic revelation of new hubs to enroll their abilities with the CoAP benefit. To demonstrate the practicality of this approach, we have built up a multi-protocol passage for a mainstream low-spending plan coordinated access gadget. To be really useable, the CoAP service should contain a proxy to make CoAP-enabled devices that are already present in the household accessible from outside. The CoAP detail accordingly recommends the utilization of DTLS or IPSec to give end to end security between two CoAP endpoints. In our application situation, the CoAP correspondence is ended at the portal on the IAD, and in this manner no safe channel can be set up between a CoAP customer and a FS20 gadget. End to-end security between two CoAP hubs (more often than not a remote associate and the IAD) is accomplished in our situation through a SSH burrow ended at the IAD. Since FS20 like most other heritage protocols does not offer appropriate security components, this is as well as can be expected get right now.

Vasileios Karagiannis [5] this paper present and compare existing IoT application layer protocols as well as protocols that are utilized to connect the things but also end-user applications to the Internet. We highlight IETFs CoAP, IBMs MQTT, HTML 5s Web socket among others, and we argue their suitability for the IoT by considering reliability, security, and energy consumption aspects. In this paper discussed the following list of protocol being used to solve different needs of the communication between machines:

Constrained Application Protocol (CoAP)

The Constrained Application Protocol (CoAP) is a synchronous request/response application layer communication that was composed by the Internet Engineering Task Force (IETF) to target obliged plan of action gadgets. It was outlined by utilizing a subset of the HTTP strategy making it interoperable with HTTP [6]. CoAP keeps running over UDP to keep the general execution lightweight. It utilizes the HTTP charges GET, POST, PUT, and DELETE to give asset situated collaboration in customer server design. CoAP is a demand/reaction convention that uses both synchronous and no concurrent reactions. The purpose behind outlining a UDP-based application layer convention to deal with the assets is to expel the TCP overhead and diminish data transfer capacity prerequisites [7]. Moreover, CoAP underpins unicast and also multicast, instead of TCP, which is by its inclination not multicast-situated. Running on the untrustworthy UDP, CoAP incorporated its own systems for accomplishing dependability. Two bits in the header of every parcel express the kind of message and the required Quality of Service (QoS) level.

There is also a simple Stop-and-Wait retransmission mechanism for confirmable messages and a 16-bit header field in each CoAP packet called Message ID which is unique and used for detecting duplicates. CoAPCHTTP Mapping enables CoAP clients to access resources on HTTP servers through a reverse proxy that translates the HTTP Status codes to the Response codes of CoAP [8]. Despite the fact that CoAP was created for the IoT and for M2M communications, it does not include any built-in security features. The convention that is proposed to secure CoAP exchanges is the Datagram Transport Layer Security (DTLS).

DTLS keeps running over UDP and is the comparable to of TLS for the TCP. It gives validation, information respectability, privacy, programmed key administration, and cryptographic algorithm [9]. Despite the fact that DTLS secures UDP exchanges, it was not intended for the IoT, along these lines its reasonableness can be contended. In the first place, DTLS does not bolster multicast [9], which is a prime favorable position of CoAP contrasted with other application layer conventions. DTLS handshake [10] requires additional packets that increase the network traffic, occupy additional computational resources, and shorten the lifespan of mobile devices that run on batteries, an essential part of the IoT. Being intended for the IoT, CoAP is HTTP-compatible, but CoAP over DTLS might create additional confusion to the HTTP servers due to its diverse packet structure. Other protocols for securing CoAP can be found in the literature including approaches that are still being under research [9] [10].

Extensible Messaging and Presence Protocol (XMPP)

This protocol was designed for chatting and message exchanging. It was standardized by the IETF over a decade ago, thus being a well-proven protocol that has been used widely all over the Internet. Be that as it may, being an old protocol, it falls short to provide the required services for some of the new arising data applications. For this reason, last year, Google stopped supporting the XMPP standard due to the lack of worldwide support. Be that as it may, lately XMPP has re-gained a lot of attention as a communication protocol suitable for the IoT. XMPP runs over TCP and provides publish/subscribe (asynchronous) and also request/ response (synchronous) messaging systems. It is designed for near real-time communications and thus, it supports small message footprint and low latency message exchange [11]. XMPP is extensible and allows the specification of XMPP Extension Protocols (XEP) that increases its functionality. XMPP has TLS/SSL security built in the core of the specification. Be that as it may, it does not provide QoS options that make it impractical for M2M communications. Only the inherited mechanisms of TCP ensure reliability. XMPP supports the publish/subscribe architecture that is more suitable for the IoT in contrast to CoAPs request/response approach. Furthermore, it is an already established protocol that is supported all over the Internet as a plus with regard to the relatively new MQTT [12]. However, XMPP uses XML messages (eXtensible Markup Language) that create additional overhead due to

unnecessary tags and require XML parsing that need additional computational ability which increases power consumption.

Restful Services

The Representational State Transfer (REST) isn't generally a protocol yet a structural style. It was first introduced by Roy Fielding in 2000 [13], and it is being widely used ever since. REST uses the HTTP methods GET, POST, PUT, and DELETE to provide a resource-oriented messaging system where all actions can be performed simply by using the synchronous request/response HTTP commands. It utilizes the inherent acknowledge header of HTTP to indicate the format of the data that it contains. The content type can be XML or JSON (JavaScript Object Notation) and depends on the HTTP server and its configuration. REST is as of now an imperative piece of the IoT because it is supported by all the commercial M2M cloud platforms. Moreover it can be implemented in Smartphone a tablet applications easily because it only requires an HTTP library which is available for all the Operative Systems (OS) distributions. The features of HTTP can be completely utilized in the REST architecture including cashing, authentication, and content type negotiation [14]. RESTful services use the secure and reliable HTTP which is the proven worldwide Internet language. It can make use of TLS/SSL for security. Be that as it may, today most commercial M2M platforms do not support HTTPS requests. Instead, they provide unique authentication keys that need to be in the header of each request to achieve some level of security.

Despite the fact that REST is already used widely in commercial M2M platforms, it is unlikely that it will become a dominant protocol due to not being easily implementable. It uses HTTP which means no compatibility with constrained-communication devices. This leaves its use for final applications. Given the current tendency for applications running on Smartphone, tablets and pads, the additional overhead associated to request/response protocols affect battery usage, as it also does the continuous polling or long polling for values especially when there are no new updates and the overhead becomes useless. Issues that can be avoided if publish/subscribe protocol are used such as MQTT or XMPP. CoAP on the other hand, which is the lightweight version of REST, bears the same disadvantages of the request/response architecture.

Advanced Message Queuing Protocol (AMQP)

The Advanced Message Queuing Protocol (AMQP) is a protocol that emerged from the monetary business. It can utilize different transport protocols but it assumes an underlying reliable transport protocol such as TCP. AMQP provides asynchronous publish/subscribe communication with messaging. Its main advantage is its store-and-forward feature that ensures reliability even after network disruptions [15]. Security is handled with the use of the TLS/SSL protocols over TCP. Recent research has shown that AMQP has low success rate at low bandwidths, but it increases as bandwidth increases [15]. Another study shows that comparing AMQP with the aforementioned REST, AMQP can send a larger amount of messages per second [16]. Additionally, it has been reported that an AMQP environment with 2,000 users spread across five continents can process 300 million messages per day [16]. Besides, JPMorgan which is an American banking and financial services company uses AMQP to send 1 billion messages per day.

Web Socket

The Web socket protocol was created as a major aspect of the HTML 5 activity to encourage communication channels over TCP. Web socket is neither a request/response nor a Publish/subscribe protocol. In Web socket a client initializes a handshake with a server to establish a Web socket session. The handshake itself is like HTTP with the goal that web servers can deal with Web socket sessions and in addition HTTP connection through a similar port. Notwithstanding, what comes after the handshake does not conform to the HTTP rules. Truth is told, during a session, the HTTP headers are removed and clients and servers can exchange messages in an asynchronous full-duplex connection.

The session can be terminated when it is no longer needed from either the server or the client side. Web socket was made to diminish the Internet correspondence overhead while giving ongoing full-duplex communications. There is also a Web socket sub-protocol called Web socket Application Messaging Protocol (WAMP) that provides publish/subscribe messaging systems Web socket keeps running over the solid TCP and executes no unwavering quality components by its own. If necessary, the sessions can be secured utilizing the Web socket over TLS/SSL. During the session, Web socket messages have only 2 bytes of overhead. As reported by relevant studies [17], the HTTP polling (in REST) repeats header information when the data transmission rate increases, thus increasing latency. Web socket is evaluated to provide a three-to-one reduction in latency against the half-duplex HTTP polling. Web socket is not outlined for resource constrained devices as the previous protocols and its client/server based architecture does not suit IoT applications. Be that as it may, it is designed for real-time communication, it is secure, it minimizes overhead and with the use of WAMP it can provide efficient messaging systems. Hence, it can compete any other protocol running over TCP.

Message Queue Telemetry Transport (MQTT)

Message Queue Telemetry Transport (MQTT) was delivered by IBM and targets lightweight M2M communications. It is an asynchronous publish/subscribe protocol that runs on top of the TCP stack. Publish/subscribe protocols meet better the IoT requirements than request/ response since clients do not have to request updates thus, the network bandwidth is decreasing and the need for using computational resources is dropping. In MQTT there is a

broker (server) [18] that contains topics. Each client can be a publisher that sends information to the broker at a specific topic or/and a subscriber that receives automatic messages every time there is a new update in a topic he is subscribed. The MQTT protocol is outlined to use bandwidth and battery usage sparingly, which is why, for example, it is currently used by Facebook Messenger. Even though MQTT runs on TCP, it is designed to have low overhead compared to other TCP-based application layer protocols [19]. Besides, the publish/subscribe architecture that it used, is more suitable for the IoT than request/response of CoAP, for example, because messages do need to be responded. This means lower network bandwidth and less message processing that actually extends the lifetime of battery-run devices.

To make sure security, MQTT brokers may require username/password authentication which is handled by TLS/SSL (Secure Sockets Layer), i.e., the same security protocols that ensure privacy for HTTP transactions all over the Internet. By comparing MQTT with the aforementioned CoAP, it is possible to see that the UDP-based CoAP has lower overhead than the TCP-based MQTT. Though, due to the lack of TCPs retransmission mechanisms, packet loss is more likely to happen when using CoAP. According to a recent research study [19], MQTT experiences lower delays that CoAP for low packet losses, but CoAP generates less extra traffic for ensuring reliability. Though, results can vary depending on the network conditions. Additionally packet loss and delays depend on the QoS of the messages. In both protocols, packet loss degrades and delays increase when the QoS level is higher.

Jamie M. Robinson, Jeremy G. Frey [20] in this paper discussed about MQTT protocol and message broker. The use of a message broker based approach gave a significant head-start in the implementation of the laboratory monitoring solution. The MQTT message broker gives message transmission reliability, the ability to distribute messages to a range of clients, and the ability to filter the message stream based on client requests. The availability of standard libraries eases the implementation of MQTT clients. By automatically collecting and distributing data, additional metadata can be provided to the experimental report that would otherwise have been missed.

III CONCLUSION

We studied that Internet of things (IOT) is a wide network having various application area for making home smart. Each application of IOT can be implemented using different standards and different protocols. We have gone through various standards and protocols and way they are used for specific application, but each one has some drawback for its technological differences. MQTT is publish/subscribe, extremely simple and lightweight messaging protocol, designed for constrained devices and low-bandwidth, high-latency or unreliable networks. The design principles also minimize network bandwidth. It is an asynchronous publish/subscribe protocol on top of the TCP stack. Publish/subscribe protocols meet better the IoT requirements than request/response since clients do not have to request updates thus, the network bandwidth is decreases and the need for using computational resources is dropping. Every client can be a publisher that sends information to the broker at a specific topic or/and a subscriber that receives automatic messages every time there is a new update in a topic he is subscribed. Hence the paper comes to a conclusion that MQTT can be a reliable and most suitable protocol for IOT application areas that can publish real time message directly on the smart phones.

ACKNOWLEDGEMENT

I would like to thank my guide, Prof. B. K. Patil, Prof. R. A. Auti, Head of Department and Staff of Computer Science Engineering Department for their guidance. Also a heartily thank to EES COE & T, Aurangabad for valuable inputs and directions for shaping this project.

REFERENCES

[1] Seung-Chul Son, Nak-Woo Kim, Byung-Tak Lee, Chae Ho Cho, and Jo Woon Chong "A Time Synchronization Technique for CoAP-based Home Automation Systems" IEEE Transactions on Consumer Electronics, Vol. 62, No. 1, February 2016.

[2] P. Bellavista, G. Cardone, A. Corradi, and L. Foschini, Convergence of MANET and WSN in IoT urban scenarios, IEEE Sens. J., vol. 13, no. 10,pp. 3558–3567, Oct. 2013.

[3] Teymourzadeh, Rozita, et al. "Smart GSM Based Home Automation System" Systems, Process & Control (ICSPC), 2013 IEEE Conference on. IEEE, 2013.

[4] O. Bergmann, K. T. Hillmann, and S. Gerdes, "A CoAP-Gateway for Smart Homes," in Proc. International Conference on Computing, Networking and Communications, Maui, USA, pp. 446 – 450, Jan. 2012.

[5] "A Survey on Application Layer Protocols for the Internet of Things" Vasileios Karagiannis1, Periklis Chatzimisios1, Francisco Vazquez-Gallego2, Jesus Alonso-Zarate2, Transaction on IoT and Cloud Computing 2015.

[6] Angelo P. Castellani, Mattia Gheda, Nicola Bui, Michele Rossi, Michele Zorzi, Web Services for the Internet of Things through CoAP and EXI, IEEE International Conference on Communications Workshops (ICC), 5-9 June 2011, pp. 1-6.

[7] Sye Loong Keoh, Sandeep S. Kumar, Hannes Tschofenig, Securing the Internet of Things: A Standardization Perspective, Internet of Things Journal IEEE (Volume: 1, Issue: 3), June 2014, pp. 265-275.

[8] Maria Rita Palattella, Nicola Accettura, Xavier Vilajosana, Thomas Watteyne, Luigi Alfredo Grieco, Gennaro Boggia, Mischa Dohler, Standardized Protocol Stack for the 8 Internet of (Important) Things, Communications Surveys & Tutorials

IEEE 15(3), 2013, pp. 1389-1406.
[9] Thamer A. Alghamdi, Aboubaker Lasebae, Mahdi Aiash, Security Analysis of the Constrained Application Protocol in the Internet of Things, Second International Conference on Future Generation Communication Technology (FGCT), 12-14 Nov.2013, pp. 163-168.
[10] Shahid Raza, Hossein Shafagh, Kasun Hewage, Ren Hummen, Thiemo Voigt, Lithe: Lightweight Secure CoAP for the Internet of Things, Sensors Journal, IEEE 13(10), Oct. 2013, pp. 3711-3720.
[11] Sven Bendel, Thomas pringer, Daniel Schuster, Alexander Schill, Ralf Ackermann, Michael Ameling, A Service Infrastructure for the Internet of Things based on XMPP, IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops), 18-22 March 2013, pp. 385-388.
[12] Michael Kirsche, Ronny Klauck, Unify to Bridge Gaps: Bringing XMPP into the Internet of Things, IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops), 19-23 March 2012, pp. 455-458.
[13] Roy Thomas Fielding, Architectural Styles and the Design of Network-based Software Architectures, PhD thesis, University of California, Irvine, USA, 2000.
[14] Bipin Upadhyaya, Ying Zou, Hua Xiao, Joanna Ng, Alex Lau, Migration of SOAP based.
[15] Frank T. Johnsen, Trude H. Bloebaum, Morten Avlesen, Skage Spjelkavik, Bjorn Vik, Evaluation of Transport Protocols for Web Services, Military Communications and Information Systems Conference (MCC), 7-9 Oct. 2013, pp. 1-6.
[16] Joel L. Fernandes, Ivo C. Lopes, Joel J. P. C. Rodrigues, Sana Ullah Performance Evaluation of RESTful Web Services and AMQP Protocol, Fifth International Conference on Ubiquitous and Future Networks (ICUFN), 2-5 July 2013, pp. 810-815.
[17] Victoria Pimentel, Bradford G. Nickerson, Communicating and Displaying Real-Time.
[18] Shinho Lee, Hyeonwoo Kim, Dong-kweon Hong, Hongtaek Ju, Correlation Analysis of MQTT Loss and Delay According to QoS Level, International Conference on Information Networking (ICOIN), 28-30 Jan. 2013, pp. 714-717.
[19] Dinesh Thangavel, Xiaoping Ma, Alvin Valera, Hwee-Xian Tan, Colin Keng-Yan Tan, Performance Evaluation of MQTT and CoAP via a Common Middleware, IEEE Ninth International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), 21-24 April 2014, pp. 1-6.
[20] Jamie M, Robinson; Jeremy G, Frey; Andy J, Stanford-Clark; Andrew D, Reynolds; Bharat V, Bedi; ―Sensor Networks and Grid Middleware for Laboratory Monitoring‖ School of chemistry, IBM UK Laboratories.