Scientific Research Publishing

Advances in Internet of Things, 2021, 11, 26-41
https://www.scirp.org/journal/ait
ISSN Online: 2161-6825
ISSN Print: 2161-6817

Performance Analysis of the Hybrid MQTT/UMA and Restful IoT Security Model

Omar H. Alhazmi[1], Khalid S. Aloufi[2]

[1]Department of Computer Science, Taibah University, Medina, Saudi Arabia
[2]Department of Computer Engineering, Taibah University, Medina, Saudi Arabia
Email: ohhazmi@taibahu.edu.sa, koufi@taibahu.edu.sa

How to cite this paper: Alhazmi, Q.H. and Aloufi, K.S. (2021) Performance Analysis of the Hybrid MQTT/UMA and Restful IoT Security Model. *Advances in Internet of Things*, **11**, 26-41.
https://doi.org/10.4236/ait.2021.111003

Received: December 5, 2020
Accepted: January 25, 2021
Published: January 28, 2021

Abstract

Internet of Things (IoT) environments are being deployed all over the globe. They have the potential to form solutions to applications, from small scale applications to national and international ones. Therefore, scalability, performance, and security form a triangle of requirements that must be carefully set. Furthermore, IoT applications require higher security standards. A previously proposed IoT application framework with a security embedded structure using the integration between message queue telemetry transport (MQTT) and user-managed access (UMA) is analyzed in this work. The performance analysis of the model is presented. Comparing the model with existing models and different design structures shows that the model presented in this work is promising for a functioning IoT design model with security. The results and analysis showed that the built-in security model had performed better than models with other frameworks, especially with fog implementation.

Keywords

IoT, MQTT (Message Queuing Telemetry Transport), UMA (User-Managed Access), Network Security

1. Introduction

The area of the Internet of Things (IoT) has faced competing requirements of performance and security. This tradeoff has always existed in technology and communication. However, the heterogeneous nature, constrained devices, and multiple overlapping protocols had made it more complicated than the conventional environment.

MQTT protocol is gaining popularity and higher usage rates over the years.

Figure 1 shows a survey done by Cabe [1] showing trends from 2016 to 2018, where 62.61% of the respondents use MQTT, which exceeds the use of HTTP that has reduced to just 54.1%.

1.1. Research Questions

This work aims to examine and analyze the MQTT/UMA Hybrid model proposed by Aloufi and Alhazmi [2]. The research question we would like to investigate is that: Is the hybrid MQTT/UMA model efficient enough and with reasonable cost? If yes, how does it compare to other secure schemes?

In this paper, we briefly explain the model; present a thorough investigation of it; analyze and measure its performance; the model is compared to a Restful model proposed in [3]; the comparison results will be illustrated. Furthermore, the models are compared by considering the attributes of service time, latency, and CPU utilization.

1.2. Literature Review and Related Works

The security of IoT remains a top concern within its deployment remains a major challenge; hence, Zhao suggested that IoT sensors can be integrated into the cloud [4]; Thus, the cloud has become a convenient platform to deploy scalable and efficient IoT [5] [6]. Moreover, the business model can be built around providing Sensing as a Service module (SenaaS) [7]. However, latency issues became a performance drawback of using clouds; fog or edge computing provided proximity complements to the cloud to overcome the latency issues with promising results for an efficient IoT [7] [8] [9].

IoT protocols are the backbone of its systems; MQTT, HTTP and CoAP are prevalent protocols (see Figure 1). MQTT as a standard is presented in [10]; MQTT is a lightweight protocol that uses subscriber/publisher model with high efficiency suitable to IoT environment of low powered constrained devices.

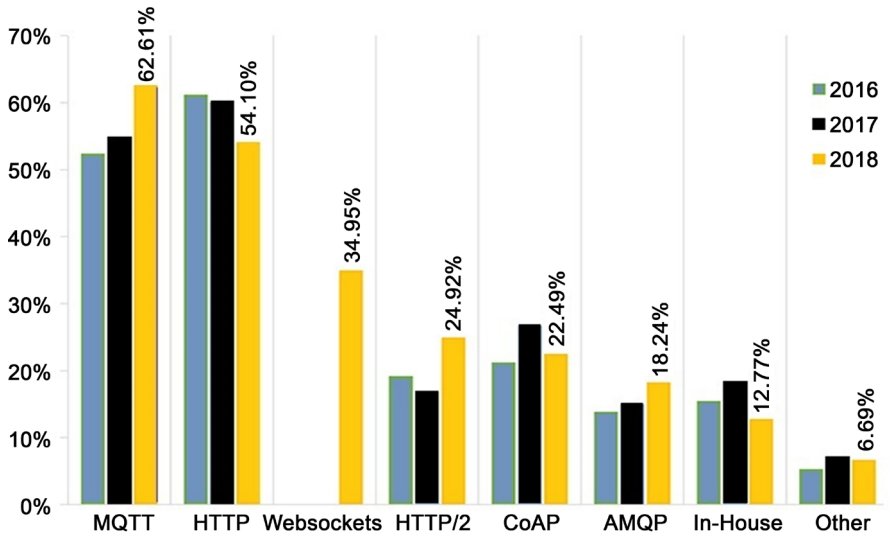

Figure 1. 2016-2018: Changes in Usage of Protocols in IoT environments [1].

Ugalde [11] shows that standard MQTT is not secure enough and needs to have a security framework. Moreover, Mishra and Kertsiz have surveyed the evolution of MQTT and its implementations over the past two decades, with various MQTT implementations; furthermore, they have concluded that MQTT security is evolving and is expected to continue its evolution [12].

Aloufi and Alhazmi suggested an IoT specific deployment of MQTT protocols; in the deployment, the MQTT broker is in the fog part of the cloud to improve latency and over the cloud-fog by placing the broker on the fog [8]. Furthermore, they have proposed an over REST architecture style as the results improved performance and reliability [3].

User-Managed Access (UMA) protocol is shown in **Figure 2** [2]; Kantara approved the protocol in 2015 [13]. UMA is based on OAuth access management [14]. Therefore, UMA 2.0 is much more recent than MQTT. OAuth 2.0 scheme that is considered is fully illustrated in [15]. Recently, Aloufi and Alhazmi have proposed a hybrid MQTT/UMA model and this paper extends the work to validate the hybrid model [2].

1.3. Paper Organization

This paper is organized into five sections as follows: in section 2 A background about the MQTT protocol and UMA security model is given 3; then, Section 3 briefly describes the hybrid model and the security Restful model; (4) Sections 4 presents the research methodology; (5) in Section 5, we highlight the results, explain, and discuss main observations about them; and (6) finally, in section 6, we conclude the paper with some remarks and results in summary.

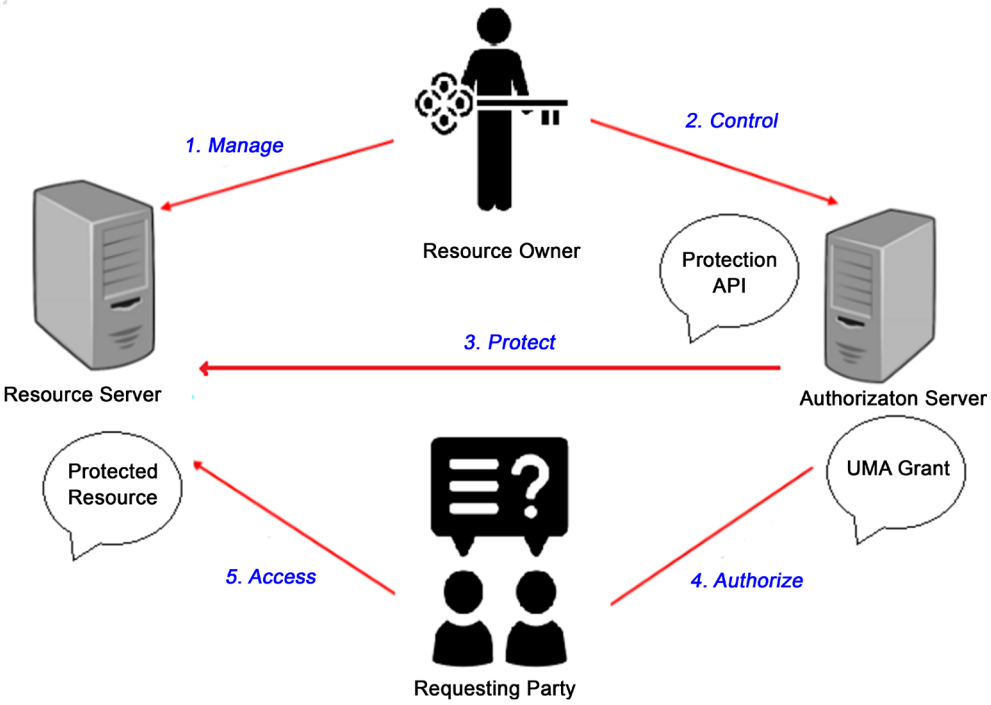

Figure 2. UMA architecure.

2. The MQTT and UMA Background

There are different IoT protocols, such as MQTT, CoAP, HTTP, and others, as shown in Figure 1; all these protocols are application layers protocols (see Figure 3 [16]).

While CoAP is highly competitive to MQTT, due to its mechanisms of exploration and observation, MQTT is probably more popular because it is much simpler to implement [17] [18] [19]. MQTT consists of clients, a broker, and subscriber/publisher relationships. The broker is the moderator, who manages messages and transactions between clients. Clients can either be subscribers, publishers, or both. The payload of messages transmitted contains data with primarily a subject and its value. The publisher sends messages to the broker periodically or when there is an update; upon receiving the messages, the broker sends the messages to a certain subject's subscribers.

As shown in Figure 1 TCP/IP model, MQTT is one of the main IoT protocols because of its reliability in delivery over TCP connections. Nevertheless, other protocols, such as CoAP is transmitted over UDP. Hence, MQTT integration over TCP makes the merger of MQTT with other protocols that share the same stack easier.

OAuth2 is an open protocol which allows secure authorization of application without providing a password [20]. User-Managed Access (UMA) is a profile of OAuth 2.0. UMA defines how different entities communicate together. UMA consists of a resource owner (RO), resource server (RS), authorization server (AS), client and the requesting party (RP). The components work together to secure access to resources using protection API, authorization API, and tokens to access the protected resource.

The resource owner (RO) controls resource access by clients. The client is operated by a requesting party (RP). Resources are hosted by a resource server (RS). An authorization server (AS) has roles defined by the resource owner to

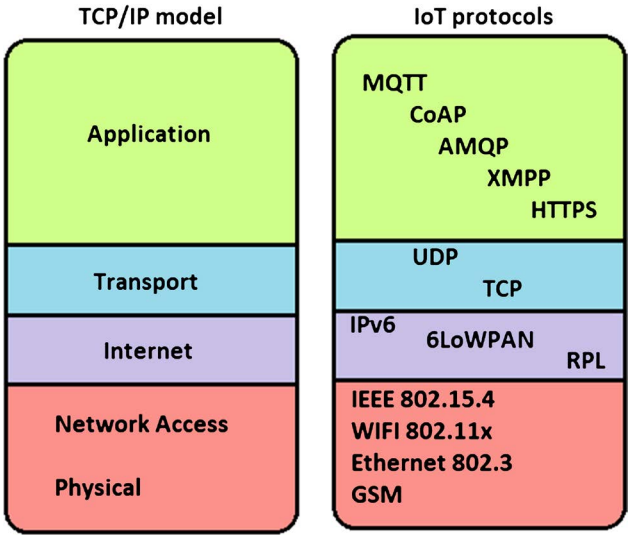

Figure 3. IoT protocols mapped to TCP/IP layers [16].

access resources. For example, a resource owner can grant access to an application for one-time access to a resource following some roles defined by the owner and managed by the authorization server. There are three phases of the UMA of resources, as shown in **Figure 2**, which are protection, authorization, and access [2] [20]. **Figure 2** shows the transactions. The resource owner is managing resources at the resource server. The resource owner controls the authorization server, which provides the resources with protection API. Moreover, the protection API requires a protection API token (PAT) to access a resource. For authentication, the client gets access to a set of resources in the resource server after being authorized by the authorization server (AS). The authorization API uses an authorization API token (AAT) to get a requesting party token (RPT). Finally, the client accesses a resource in the resource server using an RPT.

The client connects with the RS. It gets the resource; then the client connects to the AS to get a grant of access, and finally, the client can get the resource from the RS. This transaction is updated from the original one since the UMA model is updated, and the client has no direct connection with the RO [1] [21]. The UMA model has different APIs and tokens to be used. Using the protection API, a PAT is issued by the AS to the RS to access the protection API. The RS is then issued using the RPT to protect a specific resource for a specific owner. An AAT is issued by the AS using the Authorization API when the RP contacts the AS with a well-formatted request as recommended by UMA [20].

The RP will use the token to contact the Authorization API again to get a Requesting Party Token (RPT), which is the token that the RP is working to grant based on ATT.

The RO logs in to the RS to share a R. Then, the RS connects with the AS to get a PAT for the R. The AS replies with a PAT with the, RS, and RO information. After that, when the client (RP) requests access to the R at the RS, the client requires the access credentials. For this reason, the client is communicating with the AS to get the RPT and AAT for the specific R. Before the access permission; the AS sets the access permissions rules. Finally, the client gets the R with a valid RPT. The UMA data should be exchanged in JSON format [22] [23].

3. The Compared Models

3.1. MQTT/UMA Hybrid Model

Let us briefly describe the MQTT/UMA hybrid model proposed and explained in detail by Aloufi and Alhazmi [2]. The system is composed of two known systems, which are the UMA and MQTT. The system is built by directly mapping of the MQTT and UMA protocols. The proposed model considers providing the features of both protocols. Therefore, the model extends UMA's area to reach IoT devices that work with MQTT protocols. Thus, the model requires mapping between UMA and MQTT's functionality to work as one system. This section shows the mapping of the function of both protocols, UMA and MQTT.

The AS, RS, RO, client, and RP are mapped with the MQTT broker, MQTT

publisher, and MQTT subscribers. The model, shown in Figure 4, consists of one UMA and two MQTT. MQTT manages topics which are considered an R in UMA. The first MQTT consists of MQTT Broker 1 (MB1), subscriber (S1), and publisher (P1). The second MQTT consists of MQTT Broker 2 (MB2), subscriber (S2), and publisher (P2). The model works by having P1 publish topics from IoT devices. MB1 gets the published topics from P1. After-that, MB1 publishes topics to MB2. S1 is MB2 and P2 is MB1. After the topic is published from P1 to MB1, MB2 gets the published topics from MB1. Finally, MB2 publishes topics to S2, the subscriber in the second MQTT.

To add UMA to the model, MB1 is mapped with the RO because of both similar functionalities since both have resources to share. The S2 is mapped with a RP since S2 subscribes to a topic with the MB2. The S2 is considered an RP because it will require a client to access a resource published by a MB2. The MB2 works as a RS because of the similar functionality of holding the resources access permissions. For the model to be practical and logical for implementations, the AS and client are considered two different standalone fog servers.

One of the main transactions added by the MQTT broker is the publishing of any new topic value. Now the RS has the resource and its value. The resource is available for any successful RP access. However, when a new value of a resource is received, RS must update the resource and publish it to its subscribers. The integration between the functionality of the MB2 and RS is the main contribution of the model.

Figure 4 shows the Hybrid MQTT/UMA model proposed by the authors in [2]. The model consists of the P1 as a publisher, S2/RP as a subscriber, RS/MB2/S1, MB1/P2/RO, client, and AS. As will be shown later, most of the proposed model's transaction messages are between the three main entities, which are S1/MB2/RS, client, and AS. Therefore, to increase the model's performance, RS/MB2/S1, Client and AS are in the fog layer.

The basic MQTT model consists of an MQTT broker, subscribers, and publishers.

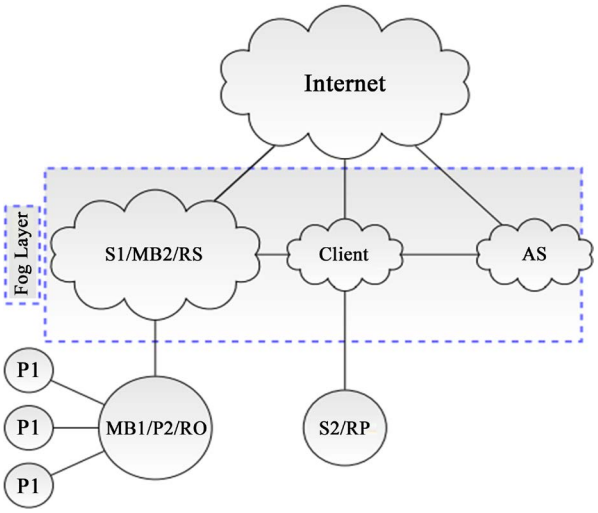

Figure 4. The hybrid MQTT/UMA model.

In the model, two MQTT models are joined to make a new MQTT model. Technically there are no P2 and S1 in the system. P2 and S1 functionality are integrated into MB1 and MB2, consequently. Moreover, S2/RP gets the messages from the fog layer, which is expected to provide more connectivity in availability and connection speed than the first mode and more security enhancement with UMA model.

MB2 is in the fog layer, which is connected directly with MB1. The model can be extended by having the MB2 connected to more than one MB1. Furthermore, each MB1 could be using one or more methods for connectivity to the fog layer.

The fog layer provides more performance and security in terms of the high Quality of Service (QoS) required of a large-scale application [24]. One of the expected benefits of using the fog cloud is to decrease the response time between the subscribers and publishers.

The publishers notify the broker of any updates, and the broker notifies the subscribers. The broker also has the log of each publisher in case it is needed by subscribers or for more statistical computations, such as the average value of a specific publisher or timing values, such as the last time a specific publisher updated its value.

3.2. The Restful Model

Restful API is illustrated in **Figure 5**. Restful can be used on a wide range of protocols, including HTTP and MQTT. The authors in [3] have proposed a secure IoT Resources with Access Control Over Restful over HTTP. Furthermore, the Restful implantation model proposed in [3] is shown in **Figure 6**; it has suggested moving access control and data management of MQTT to the fog layer, which has shown significant performance improvements over other existing implementations. Here we compare this Restful implementation with the Hybrid MQTT/UMA model. **Figure 7** shows the flow diagram of the hybrid model

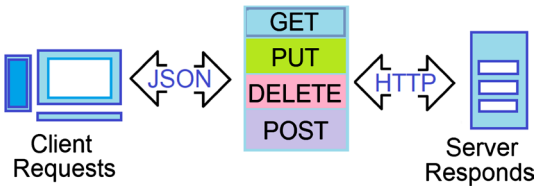

Figure 5. Restful ımplementation.

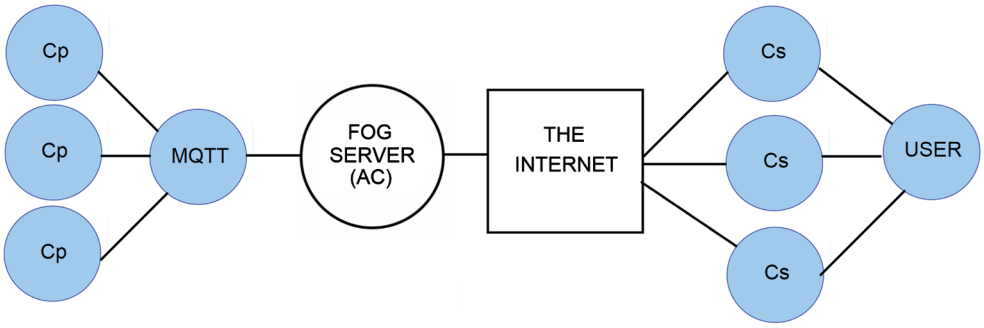

Figure 6. The Restful impmention using fog layer for broker [3].

proposed in [2].

4. The Evaluation Methodology

This section shows the results and discussion of the simulation experiments of the IoT model shown earlier. While P1 is in direct connection to MB1/P2/RO, there is no direct connection between MB1/P2/RO and S2/RP. S2/RP obtains the published topic from S1/MB2/RS, which is well established in security and performance in the fog layer.

MB1/P2/RO is expected to be a private property node in a home or a building at a security level that should not establish connections with general ones, such as RPs. Additionally, S1/MB2/RS is assumed to be connected with MB1/P2/RO with a normal Internet connection.

This is an increase in security without added security overhead for MB1/P2/RO messages. Effectively, the gained security overhead for messages is added to those between S1/MB2/RS and MB1/P2/RO and between S1/MB2/RS and S2/RP.

Figure 4 shows messages exchanged between UMA and MQTT units to provide a reliable IoT system that uses UMA as an added security layer. Most message transactions in the model are between the RS, client, and AS. Therefore, S1/MB2/RS, AS and client are allocated in the fog layer of the model with a high-speed connection with each other to increase the QoS.

The model of the MQTT/UMA hybrid system is evaluated by a simulation. Figure 4 and Table 1 show the simulation parameters and system design. The simulation is configured as shown in Figure 4, while Table 1 shows the transaction time between different connected nodes and the processing time at each node. The model is mutually inclusive between the UMA and the two-broker systems of MB1 and MB2.

Table 1. Time required for processing and transaction.

Parameter	t (ms.)
T_{P1}	10
$T_{MB1/P2/RO}$	100
T_{AS}	13 + 10 + 10
$T_{S1/M\ B2/RS}$	13 + 10 + 10
$T_{S2/RP}$	100
T_{Clinet}	10
$T_{P1 \times MB1/P2/RO}$	4
$T_{MB1/P2/RO \times S1/MB2/RS}$	200
$T_{S1/M\ B2/RS \times AS}$	3
$T_{Client \times S1/M\ B2/RS}$	3
$T_{ClientAS}$	3
$T_{S2/RPClient}$	200

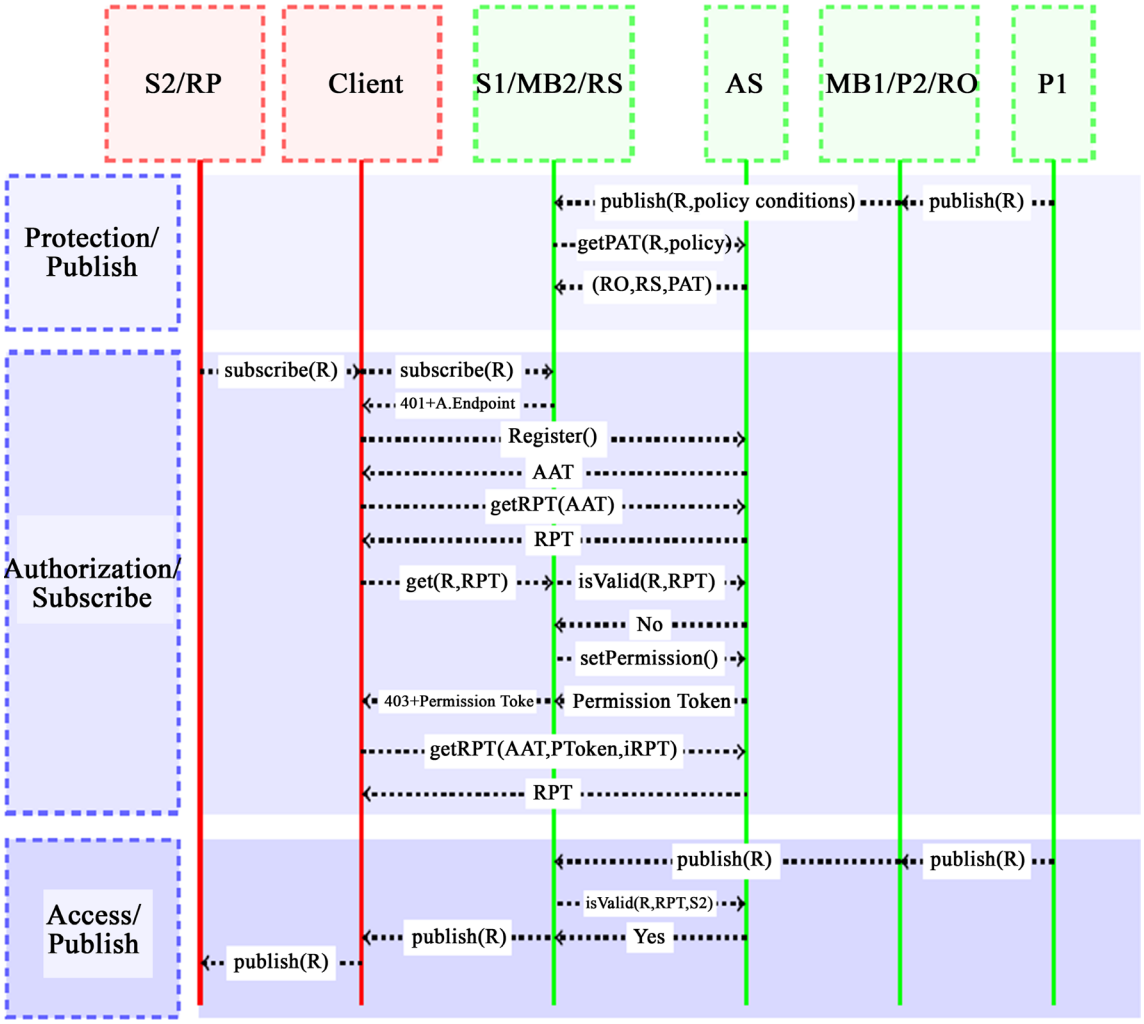

Figure 7. Hybrid MQTT/UMA transaction flow.

The processing time at each node is as follows. IoT devices are assumed to publish data with the Poisson arrival process. The processing time at each IoT device is referred to as $TP1$ and assumed to require 10 ms. The processing times $TMB1 = P2 = RO$ and $TS2 = RP$ are assumed to require 100 ms. Fog servers are assumed to be equipped with powerful nodes, such as Intel XeonW-3275M @ 2.50 GHz. In the fog server, the mean time required to access a database at S1 = MB1 = RS is 13 ms. The mean time required to query the database is approximately 10 ms. The mean processing time at S1 = MB2 = RS, AS, and Client is 10 ms. Therefore, TAS and TS1 = MB2 = RS require 33 ms; for comparison reasons, the simulation follows the specification of Aloufi and Alhazmi [2]. While $TClient$ requires only 10 ms. transmission time that required at each node is as follows. A duration of 200 ms is the required time for $T_{MB1=P2=RO \times S1=MB2=RS}$ and $T_{S2=RP \times Client}$, which are the two connections to the fog layer in the system. There are different online MQTT servers, such as test.mosquitto.org [25]. Based on the measure connection by ping to the MQTT broker at test.mosquitto.org, the average transmission time to servers in the fog layer is approximately 200 ms.

The transmission time between the IoT devices, $P1$ and node MB1 = P2 = RO is represented by $T_{P1\times MB1=P2=RO}$ and requires 4 ms. Node $MB1 = P2 = RO$ has a connection of 10 - 100 Mbps over the 802.11 g protocol or wired connection. Therefore, MB1 = P2 = RO can connect with a high data rate from IoT devices at once. However, the sending rate of each IoT device is much lower, allowing the broker to receive data from several IoT devices. Each IoT device is equipped with ZigBee, using an IEEE 802.15.4 antenna, with a rate of 250 kbps and a maximum message size of 127 bytes, according to ZigBee Specification Osipov [26]. As a result, the transmission time of one message is approximately 4 ms.

The transmission time for the nodes on the fog layer $T_{S1=MB2=RS\times AS}$, $T_{Client\times S1=MB2=RS}$ and $TClient\times AS$ is 3 ms. A ping between two servers in Europe assumed to be servers in the fog layer, requires between 3 ms and 100 ms, depending on the location of the fog servers. In this work, it is assumed that the fog server is in the same area, such as in the same country. Therefore, the assumed time is approximately 3 ms to exchange a message between two fog servers. The tested ping between two fog servers in different areas, such as the US and Europe, is approximately 200 ms.

From the simulation experiments, the arrival rate of data from each of the IoT devices is 127 bytes per second, which is one reading of subject data keeping the data size minimal to avoid fragmentation. Because the fog server is assumed to be in nearby locations, the transmission time is assumed to be 3 ms.

$$\begin{aligned}&1\times T_{P1}+1\times T_{MB1/P2/RO}+2\times T_{AS}+3\times T_{S1/MB2/RS}+1\times T_{P1\times MB1/P2/RO}\\&+1\times T_{MB1/P2/RO\times S1/MB2/RS}+2\times T_{S1/MB2/RS\times AS}=497\text{ ms}\end{aligned}\tag{1}$$

$$\begin{aligned}&10\times T_{AS}+6\times T_{S1/MB2/RS}+10\times T_{Client}+1\times T_{S2/RP}+1\times T_{S2/RP\times Client}\\&+4\times T_{Client\times S1/MB2/RS\times AS}+6\times T_{Client\times AS}=970\text{ ms}\end{aligned}\tag{2}$$

$$\begin{aligned}&1\times T_{P1}+1\times T_{MB1/P2/RO}+2\times T_{AS}+3\times T_{S1/MB2/RS}+1\times T_{Client}+1\times T_{S2/RP}+1\times T_{P1\times MB1/P2/RO}\\&+1\times T_{MB1/P2/RO\times S1/MB2/RS}+2\times T_{S1/MB2/RS\times AS}+1\times T_{client\times S1/MB2/RS}+1+T_{Client\times S2/RP}=601\text{ ms}\end{aligned}\tag{3}$$

Equation (1) shows that 497 ms is required for protection and initial publishing. For authorization and subscribing, Equation (2) shows that 970 ms is required. The time of access and publishing is 601 ms by Equation (3).

5. Results and Discussion

The system is modeled as an M/M/1 queuing model with an exponential distribution of inter-arrival time and service time. The system is tested with a mean inter-arrival time between 601 ms. and arbitrarily high inter-arrival time in ms. The mean service rate is fixed at $\mu = 1/601$, which is the minimum time required for authorization and publishing. As the experiments show, the overhead created between UMA and MQTT is the main critical overhead by the model. The UMA and MQTT are originally separate models; each model has different features and objectives. In the hybrid model, MQTT and UMA are integrated to provide an extra layer for security IoT devices. The main objective of the hybrid model is to add secure access for users for IoT devices. Any IoT device is not connected to

the world unless connected to a broker or a type of protocol, which allows the IoT device to send and receive data.

However, the accessibility of IoT devices is expected to be quite high; therefore, UMA provides high scalability for IoT devices to publish their data to a range of users or RP. RP could be another IoT device. Currently, with the expected future and developing projects of smart cities and buildings, IoT devices are expected to send a large amount of data (*i.e.*, big data). Additionally, the integration of plug and play is a necessary future for the IoT. In the hybrid MQTT/UMA model, there is a probable tradeoff; in this case, the system administrator can discuss the QoS of any model of expected growing IoT projects.

For MQTT, the model is added to UMA to add publisher and subscriber functionality. The model shown in **Figure 4** has mainly S1 and MB2 added to RS, S2 added to RP, and MB1 and P2 added to RO. The integration between MB2 and RS is performed by considering any newly published data received by MB2 as a request received by RP, which is registered as a subscriber, S2, for a specific topic or resource, R.

The performance analysis of the system considers only the publishing transactions since the subscription transaction is not repetitive; therefore, the hybrid model has a negligible effect on the performance. The same applies to the protection and authorization of transactions of UMA.

For the performance of the system, the time required for publishing a message with MQTT and UMA is respectively 970 + 601 and 497 ms. for the registration, protection, authorization, access, and subscription of a message. However, only 497 ms. is required for subsequent publishing, which is only 24% of the time required for complete transaction. UMA does not repeat the expensive transaction more than the first time. Consequently, the performance gain by UMA is 74% with MQTT for secure publishing.

The main model proposed in Aloufi and Alhazmi [2] is referred to as Hybrid MQTT/UMA model with a Fog Layer (HMUFL).

The hybrid model is evaluated by a comparison with two different models as follows:

Hybrid MQTT/UMA with fog computing (HMUFL) [2].

Hybrid MQTT/UMW without fog (HMULL) [2].

Access Control over Restful Web Services (ACRWS) [3].

$$1\times T_{p} + 1\times T_{s1} \times T_{FS} + 1\times T_{FS\times S} + 3\times T_{p\times MB} + 3\times T_{MB\times FS} = 1065 \text{ ms} \quad (4)$$

For HMUFL, the mean service time is 601, while it is 1389 ms for HMULL. The publishing time for HMULL is computed using Equation (3); however, because there is no fog layer in this model, some parameters require different timing. Therefore, in **Table 1** TS1/MB2/RS * AS, TClient * S1/MB2/RS, 0.2 and TClient * S2/RP changes from 3 ms to 200 ms.

For ACRWS, 1065 ms is required for publishing time, as shown by Equation (4). ACRWS is a model of access control based on web services. In equations, TP, TS, and TFS are the processing times at the IoT device, subscriber, and fog

server, respectively.

The transmission time between the fog server and the subscribers, the publisher and the MQTT broker, and between the MQTT broker and the fog server is referred to by TFS * S, TP * MB, and TMB * FS, respectively. ACRWS is simpler in design than the other two models. It has more transactions for publishing a topic from the publisher to the subscribers.

Figure 8 shows the system utilization. The utilization shows a clearer advantage of using the hybrid model with fog than the restful than the hybrid without fog; here the models compare to each other with various but linear growth rates. However, in Figure 9 the mean waiting number (queue size) and the mean waiting time shows exponential growth in the non-fog hybrid model and in the restful model; however, the hybrid with fog grows in a much slower pace showing clear advantage in performance. Likewise, in Figure 10 almost the same rate is shown but this figure considers the mean waiting time for messages. Moreover, Figure 11 depicts the waiting time and queue size; here, the same number of messages takes longer to process and longer queue sizes in the hybrid without

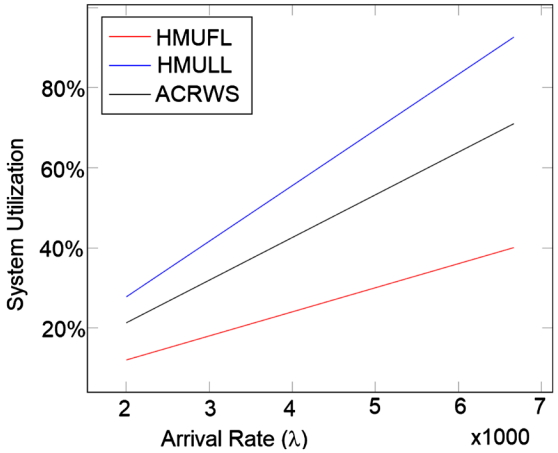

Figure 8. System utilization for different arraival rates.

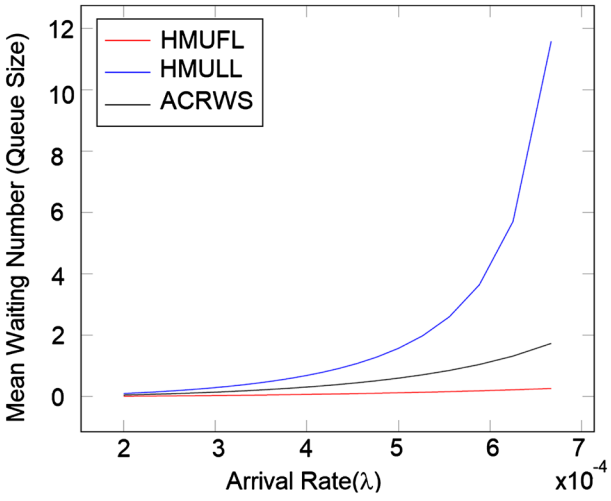

Figure 9. Mean waiting number (queue size) vs. arrival rate.

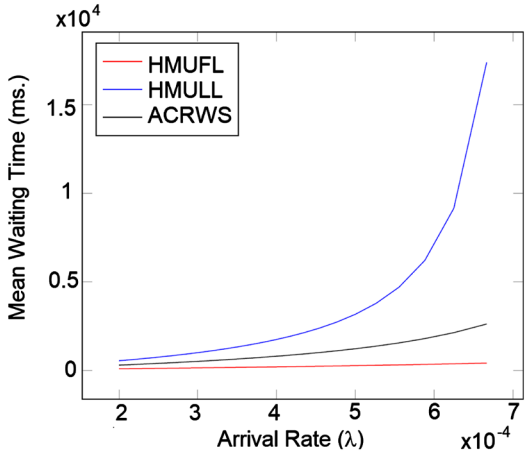

Figure 10. Mean waiting time in milliseconds vs. arrival rate.

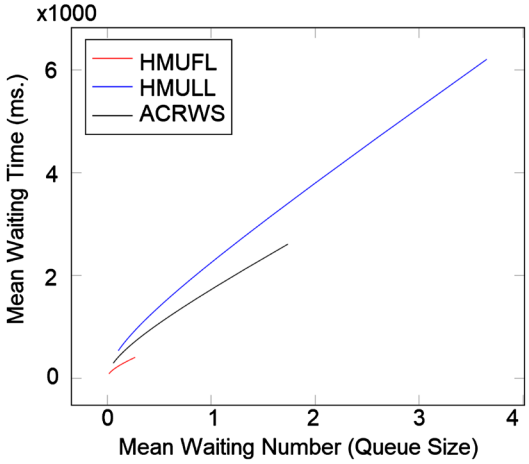

Figure 11. Mean waiting time vs. mean waiting number.

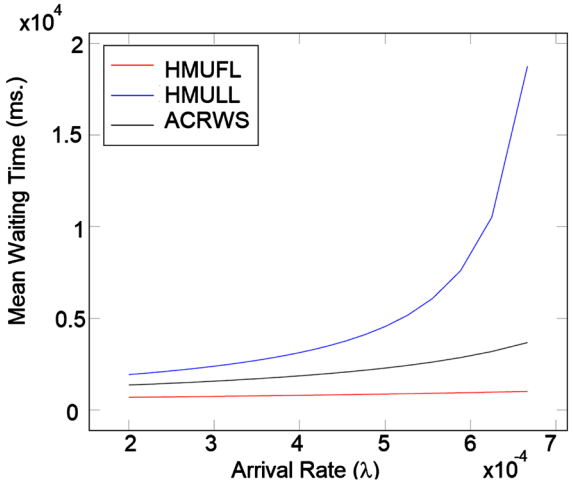

Figure 12. Proortion of time the system is idle.

fog while the ACRWS is in the middle and the hybrid with cloud the time and the queue sizes are much smaller. The proportion of time the server is idle is shown in Figure 12.

Overall performance comparisons show that the hybrid model shows better performance when deployed over fog. The performance comparison shows that while there is a complexity of the access control mechanism design between models has created a gap in queues size and waiting time.

6. Conclusions

In conclusion, IoT applications are promising for future implementation in different domains, such as smart cities or SenaaS model of business.

When an IoT application works, the number of devices, as well as the size of the data, grows exponentially. Security is required to secure access to data sources. Data needs to be protected without compromising performance, necessitating a security layer. In this work, the UMA security protocol is used to add security with the well-known IoT application protocol MQTT.

There are different IoT application protocols; however, MQTT is the most popular for its implementation simplicity. This work has shown how to integrate MQTT and UMA in one fast-performing and secure model. The integration is maintained by mapping the functionality of both protocols. Performance comparisons have shown that there is a performance gain for different functionalities, mainly the publishing function. Publishing is the primary function of any IoT device. The model has successfully connected remote IoT devices to the publishing network without compromising security. The data are published and accessed without accessing the source of the data. Thus, UMA is a promising model for IoT security.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Benjamin, C. (2018) Key Trends from the IoT Developer Survey 2018. https://blog.benjamin-cabe.com/2018/04/17/key-trends-iot-developer-survey-2018

[2] Aloufi, K.S. and Alhazmi, O.H. (2020) A Hybrid MQTT/UMA Internet of Things Model. *Communications and Network*, **12**, 155-173. https://doi.org/10.4236/cn.2020.124008

[3] Aloufi, K.S. and Alhazmi, O.H. (2020) Secure IoT Resources with Access Control Over Restful Web Services. *Jordanian Journal of Electrical Engineering*, **6**, 63-77. https://doi.org/10.5455/jjee.204-1581015531

[4] Zhao, F. (2010) Sensors Meet the Cloud: Planetary-Scale Distributed Sensing and Decision Making. 2010 9*th IEEE International Conference on Cognitive Informatics*, Beijing, 7-9 July 2004, 998-998. https://doi.org/10.1109/COGINF.2010.5599715

[5] Biswas, A.R. and Giaffreda, R. (2014) IoT and Cloud Convergence: Opportunities and Challenges. 2014 *IEEE World Forum on Internet of Things*, Seoul, 6-8 March 2014, 375-376. https://doi.org/10.1109/WF-IoT.2014.6803194

[6] Zeydan, E., Bastug, E., Bennis M., Kader, M.A., Karatepe, I.A. and Debba, M. (2016) Big Data Caching for Networking: Moving from Cloud to Edge. *IEEE Communica-*

tions Magazine, **54**, 36-42. https://doi.org/10.1109/MCOM.2016.7565185

[7] Alam, S., Chowdhury, M.M.R. and Noll, J. (2010) SenaaS: An Event-Driven Sensor Virtualization Approach for Internet of Things Cloud. *IEEE International Conference on Networked Embedded Systems for Enterprise Applications*, Suzhou, 25-26 November 2010, 1-6. https://doi.org/10.1109/NESEA.2010.5678060

[8] Alhazmi, O.H. and Aloufi, K.S. (2019) Fog-Based Internet of Things: A Security Scheme. In 2019 2*nd International Conference on Computer Applications Information Security*, Riyadh, 1-3 May 2019, 1-6. https://doi.org/10.1109/CAIS.2019.8769506

[9] Sharma, S.K. and Wang, X. (2017) Live Data Analytics with Collaborative Edge and Cloud Processing in Wireless IoT Networks. *Access IEEE*, **5**, 4621-4635. https://doi.org/10.1109/ACCESS.2017.2682640

[10] International Organization for Standardization (2016) ISO/IEC 20922: 2016 Information Technology—Message Queuing Telemetry Transport (MQTT) v3.1.1. https://www.iso.org/standard/69466.html

[11] Ugalde, D.S. (2018) Security analysis for MQTT in the Internet of Things. Master Thesis, KTH Royal Institute of Technology (School of Electrical Engineering and Computer Science), Sweden. https://www.diva-portal.org/smash/get/diva2:1295431/FULLTEXT01.pdf

[12] Mishra, B. and Kertesz, A. (2020) The Use of MQTT in M2M and IoT Systems: A Survey. *IEEE Access*, **8**, 201071-201086. https://doi.org/10.1109/ACCESS.2020.3035849

[13] Machulak, M. and Richer, J. (2015) User Managed Access, 2015. https://docs.kantarainitiative.org/uma/wg/rec-oauth-uma-grant-2.0.html

[14] Internet Engineering Task Force (2016) The OAuth 2.0 Authorization Framework. RFC 6749. https://tools.ietf.org/html/rfc6749

[15] Dewni Weeraman (2017) User Managed Access—UMA 2.0. https://medium.com/@dewni.matheesha/user-managed-access-uma-2-0-bcecb1d535b3

[16] Gerber, A. and Romeo, J. (2017) Connecting All the Things in The Internet of Things: A Guide to Selecting Network Technologies to Solve Your Iot Networking Challenges. https://developer.ibm.com/technologies/iot/articles/iot-lp101-connectivity-network-protocols/

[17] Aloufi, K.S. (2019) 6LoWPAN Stack Model Configuration for IoT Streaming Data Transmission over CoAP. *International Journal of Communication Networks and Information Security* (*IJCNIS*), **11**, 304-311. https://www.ijcnis.org/index.php/ijcnis/article/view/4034/358

[18] Yokotani, T. and Sasaki, Y. (2016) Transfer Protocols of Tiny Data Blocks in IoT And Their Performance Evaluation. 2016 *IEEE* 3*rd World Forum on Internet of Things*, Reston, 12-14 December 2016, 54-57. https://doi.org/10.1109/WF-IoT.2016.7845442

[19] Yokotani, T. and Sasaki, Y. (2016) Comparison with HTTP and MQTT on Required Network Resources for IoT. 2016 *International Conference on Control, Electronics, Renewable Energy and Communications*, Bandung, 13-15 September 2016, 1-6. https://doi.org/10.1109/ICCEREC.2016.7814989

[20] Hardjono, T., Maler, E., Machulak, M. and Catalano, D., (2013) User-Managed Access (UMA) Profile of OAuth 2.0. Internet-Draft Draft-Hardjono-Oauth-Umacore-13. https://tools.ietf.org/html/draft-hardjono-oauth-umacore-13

[21] Jones, M. and Hardt, D. (2012) The OAuth 2.0 Authorization Framework: Bearer Token Usage. RFC 6750. https://tools.ietf.org/html/rfc6750 https://doi.org/10.17487/rfc6750

[22] Siriwardena, P. (2014) OAuth 2.0. In: Siriwardena, P., Eds., *Advanced API Security*, Apress, Berkeley, 91-132. https://doi.org/10.1007/978-1-4302-6817-8_7

[23] Maler, E., Machulak, M., Richer, J. and Hardjono, T. (2019) User-Managed Access (UMA) 2.0 Grant for OAuth 2.0 Authorization: Draft-Maler-Oauth-Umagrant-00. https://tools.ietf.org/html/draft-maler-oauth-umagrant-00

[24] Al-khafajiy, M., Baker, T., Waraich, A., Al-Jumeily, D. and Hussain, A. (2018) IoT-Fog Optimal Workload via Fog Offloading. 2018 *IEEE/ACM International Conference on Utility and Cloud Computing Companion*, Zurich, 17-20 December 2018, 359-364. https://doi.org/10.1109/UCC-Companion.2018.00081

[25] Jaloudi, S. (2019) MQTT for IoT-based Applications in Smart Cities. *Palestinian Journal of Technology and Applied Sciences*, No. 2, 1-13.

[26] Osipov, M. (2008) Home Automation with ZigBee. 2008 *International Conference on Next Generation Wired/Wireless Networking*, 3-5 September 2008, St. Petersburg, 263-270. https://doi.org/10.1007/978-3-540-85500-2_26