JURNAL INFOTEL
Informatics - Telecommunication - Electronics
Website: http://ejournal.st3telkom.ac.id/index.php/infotel
ISSN: 2085-3688; e-ISSN: 2460-0997

A study of secure communication scheme in MQTT: TLS vs AES cryptography

Favian Dewanta[1,*], Baiq Yuniar Yustiarini[1], Bangun Indrakusumo Radityo Harsritanto[2]
[1]School of Electrical Engineering, Telkom University
[2]Architecture Department, Universitas Diponegoro
[1]Jl. Telekomunikasi, No. 1, Bandung 40257, Indonesia
[2]Jl. Prof. Soedarto, Semarang 50275, Indonesia
[*]Corresponding email: favian@telkomuniversity.ac.id

Received 12 July 2022, Revised 20 September 2022, Accepted 22 September 2022

Abstract — The Internet of Things (IoT) technology help devices to send command and request, exchange messages, and also communicate with an entire level of devices and infrastructures. Several IoT-based systems may communicate at a certain level of latency depending on the urgency and purpose of the communications. Therefore, several protocols with respect to their speed and reliability levels can be chosen for achieving the desired quality of services. As for lightweight and fast communication speed for IoT devices, the MQTT protocol is the most commonly known and recommended in the system. However, the MQTT protocol is not equipped with the appropriate security mechanism. As a consequence, the MQTT messages are easily eavesdropped on and modified by the attackers during communication sessions among devices through several levels of network domains including local, internet, and internal cloud-based network. Considering the well-established security approach and commonly strong cipher system, this research studies the use of the AES cryptography-based communication scheme against the TLS-based communication scheme, which can be used to create end-to-end secure communication channels from the MQTT publishers to the MQTT subscribers. Experimental results show that the TLS-based communication scheme possesses the highest cost in terms of communication delay and network cost among all schemes in the experiment. Eventually, the AES-based MQTT communication scheme is more appropriate for IoT environments because of its communication delay and network cost, which are considered equal to the plaintext-based MQTT communications.

Keywords – lightweight cryptography, MQTT, secure communication

I. Introduction

Internet of Things (IoT) technology requires low computational devices, either CPUs, memories, or storage. Those requirements are caused by having IoT devices to conduct specific tasks in an energy-limited area, such as sensing humidity, temperature, soil moisture in the farm, forest, river, mountain, and so on. Therefore, the components for implementing such mentioned systems are limited by the computational and energy resources and physical dimension with respect to higher functionality and longer device lifetime [1].

Regarding the issue of limited computational resources, Mishra and Kertesz [2] mention several popular communication protocols, which are CoAP, MQTT, HTTP, AMQP, and so on. However, they prefer the MQTT due to its simplicity and easiness of installing that protocol. In addition, the MQTT also provides several QoS for maintaining high-quality communication with respect to packet size and latency.

However, the MQTT is not equipped with a security feature in which the MQTT messages can easily be eavesdropped on and stolen by attackers. Moreover, there are several IoT applications that demand high privacy and confidentiality, such as telemedicine, Smart Home, Smart Mosque, and Industrial Plant. Thus, security issues in the MQTT protocol become a challenge for establishing trusted IoT applications [3].

With respect to the security issues in the MQTT protocol, several researchers and practitioners propose the TLS for providing confidentiality and authenticity [4], [5]. However, even though the TLS is proven to deliver services in a secure and trusted

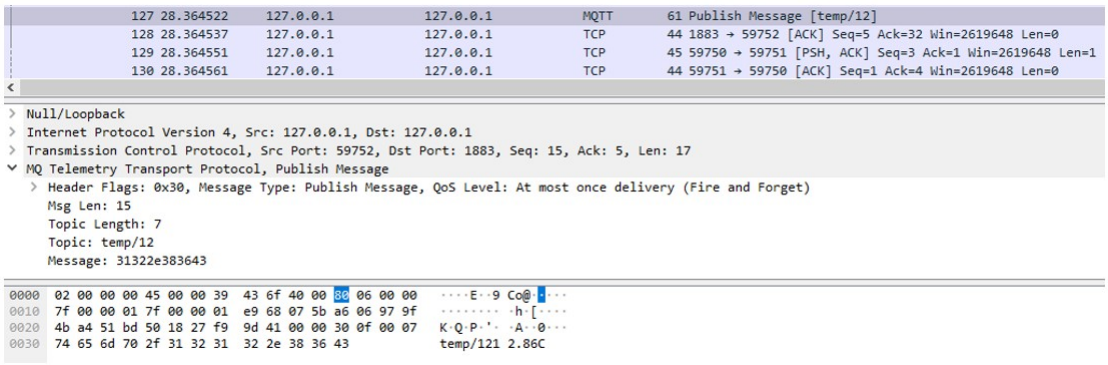

Fig. 1. The message of MQTT protocol from a publisher to the Mosquitto broker located on local host.

manner, such as internet banking and other financial technology platforms, the TLS is considered a heavy and high latency platform for securing IoT devices with low computational resources.

Several other researchers then propose the use of password-based authentication key exchange (PAKE) as an additional security component of the MQTT [6]–[8]. However, the PAKE methods are considered a partial solution because the data are stored in the MQTT broker in a plaintext condition. Moreover, the use of asymmetric cryptography, point multiplication, and the power of integer techniques are still problematic for IoT platforms due to the high computational and communication cost.

Another approach based on the covert channel mechanism is proposed by Velinov *et al.* [9]. It is proven to be effective to hide secret information by means of direct and indirect covert channels utilizing a modification of topic name, packet identifier, and application message. However, the covert channel approach leads to bandwidth degradation with respect to covert channel placement. Therefore, it can cause high latency and eventually disturb the service of real-time IoT applications. Therefore, the use of covert channel approaches is not favorable in the common situation, unless IoT applications allow slow response and high latency communication.

This paper discusses the issues of a secure communication channel in the MQTT for providing trusted IoT applications. Specifically, this paper does not discuss authentication approaches, but it limits the discussion to the use of AES [10] and TLS scheme [11] because this work is the continuation of our previous work, which discusses authentication approaches between the PAKE model and XOR and hash model [12]. Both the AES and TLS are selected in this research due to the popularity and feasibility of those two approaches for securing communication in any high-layer network protocols, including the MQTT communication protocols. Then, the contributions of our work are as listed below.

1) We explain two MQTT broker platforms (Mosquito and Hivemq) and their weaknesses in section II.
2) We describe our research methodology and experimental setup in section II.
3) Lastly, our experimental results and analyses are delivered in section III.

II. Research Method

There are three discussions in this section, which are the MQTT messaging issues, the comparison of AES and TLS approaches in MQTT communication, and the experimental setup for measuring the performance of both approaches.

A. The MQTT Messaging Issues

The MQTT protocol [2], [9], [13] is one of the IoT messaging platforms based on publish and subscribe operations. The protocol consists of several entities, which are (1) the publisher as a source of the message, (2) the subscriber as a sink of the message, and (3) the broker as a hub among the publisher and the subscriber. In order to set a communication channel between the publisher and the subscriber, those components should publish and subscribe to the same topics in the broker.

There are several scenarios for implementing MQTT-based communication in our IoT projects. One may install the MQTT publisher in the IoT devices acting as sensors by means of several programming languages. In another case, IoT devices acting as actuators may be set as the MQTT subscriber. As for the broker, sometimes it can be installed on cloud or/and edge computing servers. The locations and setup scenarios of those entities, *e.g.*, publisher, subscriber, broker, depending on the design of the IoT systems and sometimes the stakeholders of the IoT projects.

Regarding the MQTT broker platforms, the IoT project developers can utilize several available platforms rather than implementing the broker from the scratch by means of several programming languages. There are several freely available MQTT brokers, which are Mosquitto, HiveMQ, RabbitMQ, EMQ X, and so on. By using those MQTT broker platforms, IoT developers merely need to adjust the topic name and the address

(IP and ports) of those brokers in their IoT devices and cloud servers acting as the publishers and subscribers.

As discussed previously, the MQTT protocol is not equipped with security features. In the basic operation of the MQTT protocol as hosted by the Mosquitto broker, the MQTT messages are sent in the plaintext form as given in Fig. 1 in which a publisher sends an MQTT message containing temperature information '12.86C' with the topic 'temp/12' to the broker. As a consequence, those MQTT communication channels are vulnerable to any attack scenarios, including information theft, man-in-the-middle attack, reply attack, and *so on*.

As for the HiveMQ broker and/or any other enterprise-ready MQTT broker platforms, they are frequently equipped with certain message encoding based on the topic name. As a consequence, their MQTT messages seem to be encrypted as given in Fig. 2. However, if attackers can adjust their subscribers to the same topic name as the targeted communication channel between publishers and subscribers, the attackers can eventually get the plaintext form of the MQTT messages. In Fig. 3, we use the websockets client provided by the HiveMQ platforms to capture our previously encoded MQTT messages in Fig. 2 by means of adjusting the topic name. Thus, this kind of MQTT messaging mode in the HiveMQ broker and any other similar platforms is only effective for protecting the system from random outsider attacks, but not from insider attacks knowing the topic of the MQTT messaging systems.

Considering these issues of MQTT messaging, this paper proposes to use a symmetric cryptography method to protect the MQTT messages end-to-end from the publishers to the subscribers. Eventually, the study of employing AES cryptography here can be viewed as an alternative solution for providing secure communication in the MQTT protocol with respect to the TLS approach in the MQTT communication.

B. The Comparison of AES and TLS Approaches

One may wonder whether the AES and TLS approaches are comparable in this case because the TLS is a complete security system including authentication, key agreement, and encryption-decryption schemes based on message handshakes between two entities [14], [15]. The AES encryption-decryption scheme is certainly also part of the TLS scheme. However, we can argue that the AES and TLS schemes are still comparable in a way that the MQTT communication may select those two schemes based on the specification of the IoT devices. Specifically, the IoT devices with low computational resources may utilize the AES scheme rather than the TLS with the assumption that both MQTT publisher and subscriber already share the same key and encryption-decryption mode, i.e., ECB, CBC, CFB, *etc*.

In order to possess identical keys and encryption-decryption modes, there are two approaches that can be taken, which are fixed pre-shared key (PSK) that is similar to WEP in WiFi security system [16], [17] and authenticated key exchange (AKE) approaches [18], [19]. The fixed PSK approach is the simplest and easiest way, but it is more vulnerable with respect to the AKE approaches due to the use of fixed key and its consequences against several known attacks, such as brute force attacks, replay attacks, *etc*.

Regarding the TLS scheme in the MQTT communication, the MQTT agents (publishers and subscribers) can activate the TLS feature by using a public key certificate (PKC) based on X.509 digital certificate. In some MQTT broker platforms, *e.g.*, in RabbitMQ, HiveMQ, *etc.*, the SSL/TLS feature has been bundled with the MQTT broker and agents. Thus, the TLS can be executed seamlessly by using some codes provided in Python, NodeJS, and so on. However, several MQTT broker platforms do not provide the SSL/TLS feature. In this case, users need to install the OpenSSL application including the digital certificate based on X.509.

In this research, our experiment will compare the performance of MQTT communications utilizing the AES encryption-decryption scheme against the TLS scheme. Considering the TLS feature as mentioned in the previous paragraph, we decide to utilize HiveMQ broker for this experiment. As for the AES cryptography scheme, this experiment uses the AES with the key sizes 128, 192, and 256 bits.

C. MQTT Experimental Setup

The experiment is done by using the network topology as given in Fig. 4. There is an MQTT broker/server connecting an MQTT publisher and an MQTT subscriber through the internet. As for the MQTT server, we use HiveMQ broker in the TLS and non-TLS modes. As for the MQTT publisher and subscriber, we use the Paho-MQTT library based on Python programming language to implement those MQTT agents.

For increasing clarity of the experiment, the illustration of the MQTT message is given in Fig. 5. The MQTT message is encapsulated by headers in TCP, IP, and Ethernet layers respectively. Therefore, the total frame of the MQTT message will be

$$L_f = 54 + x \tag{1}$$

in which variable x represents the length of the MQTT message (bytes).

As for the MQTT message, it is located in the OSI layer 7 and it has a specific format as given in Fig. 5. The MQTT format consists of header flags (1 byte), message length (1-4 bytes), variable length header (0-Y bytes), and message/content (0-Z bytes). The variable

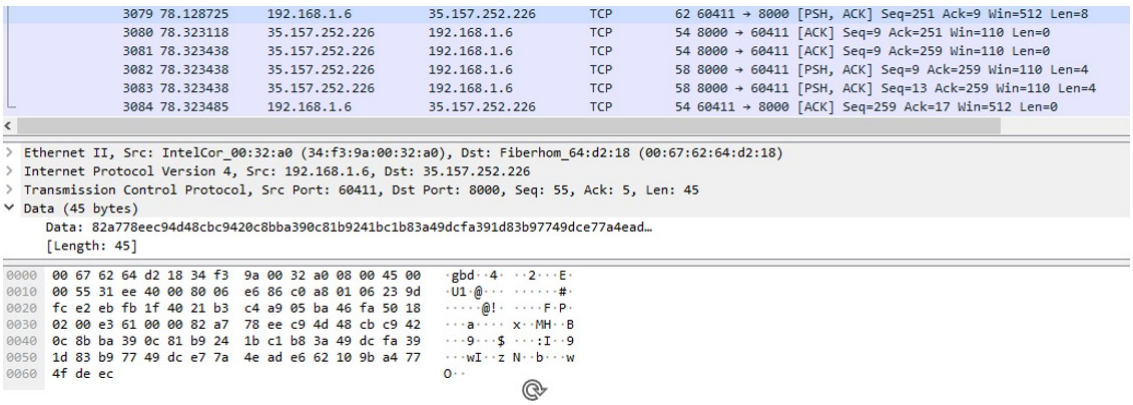

Fig. 2. The message of MQTT protocol from a publisher to the HiveMQ broker.

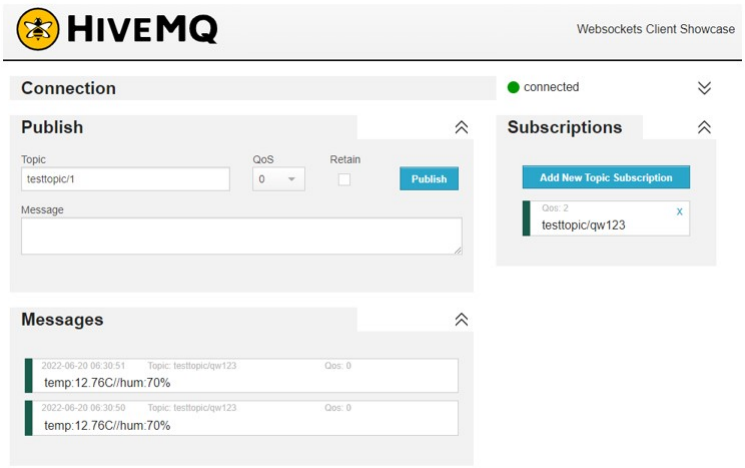

Fig. 3. The captured messages from the encoded MQTT communication channel by adjusting topic name of the HiveMQ websockets client.

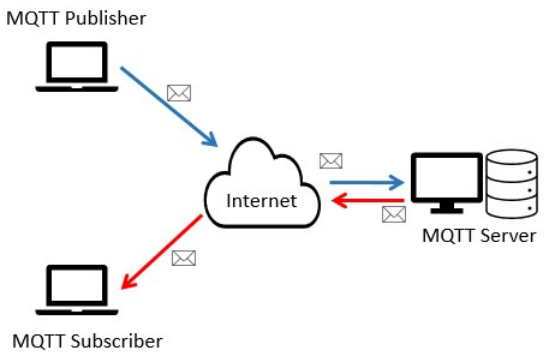

Fig. 4. The MQTT communication system used in this experiment.

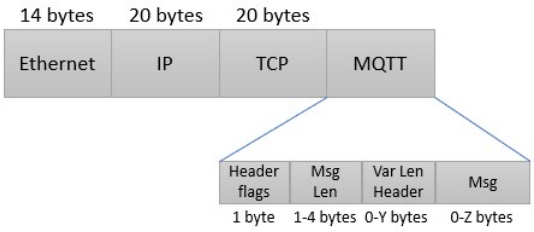

Fig. 5. The MQTT message format in the point of view OSI layer.

length header and message/content may exist in the MQTT message due to the existence of the MQTT control packet type that is described in the header flags. Further information regarding the MQTT control packet type can be checked in the following articles [2], [20].

For measuring the performance of those MQTT communication schemes (AES vs TLS), we evaluate several parameters as described below.

- Communication delay (D_{comm}) is measured by calculating the time difference between MQTT message transmission and reception. For getting the time difference, we put the timestamp in seconds in the MQTT message as given in Fig. 6 by using the Python time() method. Theoretically, the communication delay can be calculated by accumulating all components of delays in the communication process or network. In this case, a component of delays will be transmission delay (T_d) and queueing delay (Q_d) from the publisher to the subscriber. The transmission delay is the delay to transmit frames through network media and the queueing delay is the processing delay in the network devices.

$$D_{comm} = \sigma T_d + \sigma Q_d \quad (2)$$

- Computational delay is measured by calculating the time difference between the beginning and end of executing certain methods. In this case, we compute the computation delay of executing the AES encryption and decryption method. As for

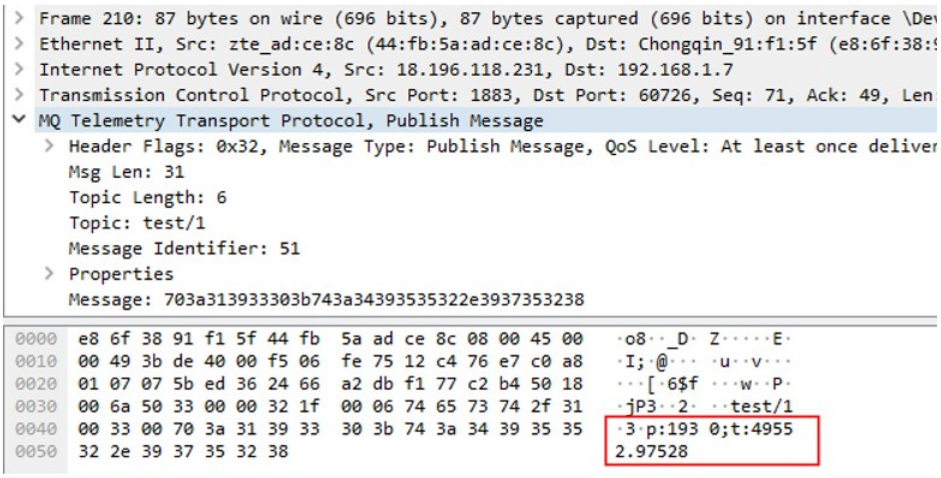

Fig. 6. The use of timestamp in the MQTT message for measuring communication delay from the publisher to the subscriber.

the plaintext and TLS schemes, the computational delay is equal because the MQTT computational process only handles the plaintext data for both plaintext and TLS schemes and the TLS is handled by the process in the lower layer (network kernel).

- Network cost is the number of packets in the network, in maintaining communication between publishers and subscribers. These packets include the MQTT message, TCP signaling packet, TLS signaling packet, *etc*.

Table 1. Parameters of Experiment

No.	Parameter	Value
1	Number of MQTT message samples per scenario	100
2	AES key size	128, 192, and 256 bits
3	MQTT payload size in plaintext	20, 30, and 40 bytes
4	MQTT message inter-departure time	2 and 5 seconds
5	Network bandwidth	Up to 10 Mbps

The experimental parameters are described in Table 1. As for the number of MQTT message samples, we use 100 samples to anticipate the unpredictability of the network traffic due to using public network infrastructure with up to 10 Mbps bandwidth. The AES key size is set to be 128, 192, and 256 bits to the given plaintext payloads which are 20, 30, and 40 bytes. At last, the MQTT message inter-arrival is set to be 2 seconds for anticipating long delays due to the high network traffic and HiveMQ server occupation.

III. Result

Before discussing the performance evaluation of both the TLS and AES, we need to show the captured packet by both schemes to demonstrate that both schemes provide confidentiality and secrecy for exchanged MQTT messages. Fig. 7 shows the captured TLS traffic using the Wireshark application. In that figure, the type of the packet above the TLS layer including its content is invisible to the attacker. As for the AES scheme in Fig. 8, the type of the packet, in which the MQTT packet, can be identified clearly by the Wireshark. However, the content of the message is covered by the AES encryption so that the content shown in Fig. 8 seems like random characters. Those two figures eventually prove that the TLS and AES schemes can secure the MQTT message content.

The experimental result for communication delay is shown in Fig. 9. There are three scenarios of payload sizes (20, 30, and 40 bytes) and MQTT communication schemes (Plaintext, AES, and TLS data transmissions). The communication delays vary from 0.31 to 0.392 seconds and there is no certain pattern that differentiates the communication delays for all payload sizes and communication schemes, except for the TLS scheme which shows slightly higher communication delay with respect to the plaintext and AES communication scheme. These communication delays are also supported by the frame size results, with respect to payload size, captured by Wireshark as shown in Fig. 10. It is clear in that figure that the frame size of TLS communication schemes is the highest one among those schemes.

However, we can see that these results generally emphasize that the significant component contributing to communication delays of those 3 payload sizes and communication schemes among MQTT publishers, broker, and subscribers are the queueing delay. The transmission delay generated from the frame size affects the communication delays, but the contribution of the transmission delay is not significant with respect to queueing delay due to using a public network with random characteristics and user behaviors. Therefore, the communication delay in this experiment is mostly driven by the network quality and there is no significant correlation between the payload size and communication delay.

Regarding the computational delay, Fig. 11 shows that AES-256 tends to have the most efficient computational delay with respect to AES-128 and AES-192. However, those delays are not significant due

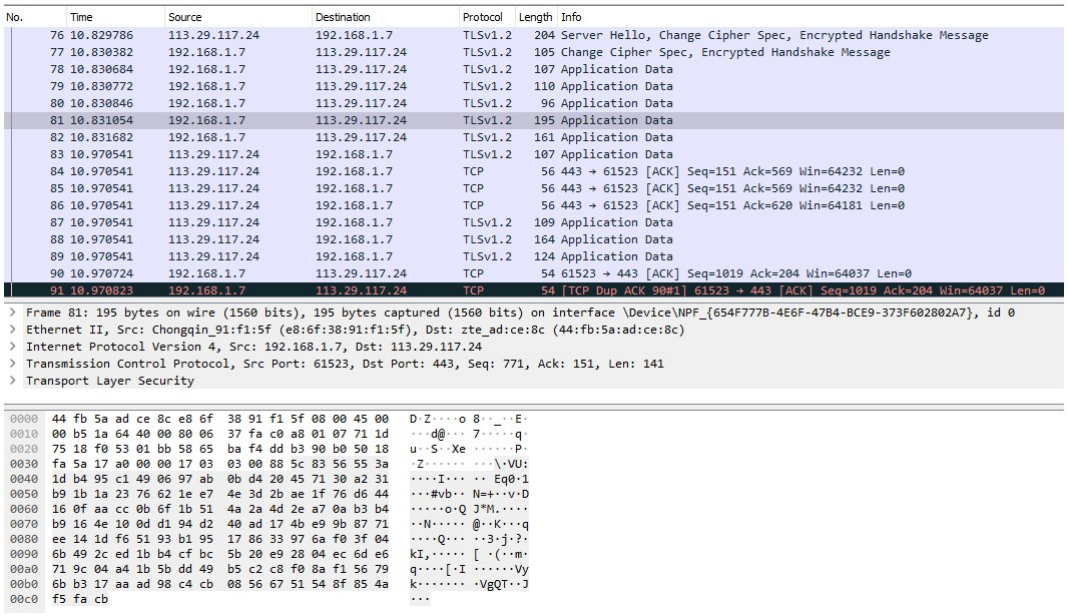

Fig. 7. The MQTT traffic protected by the TLS scheme.

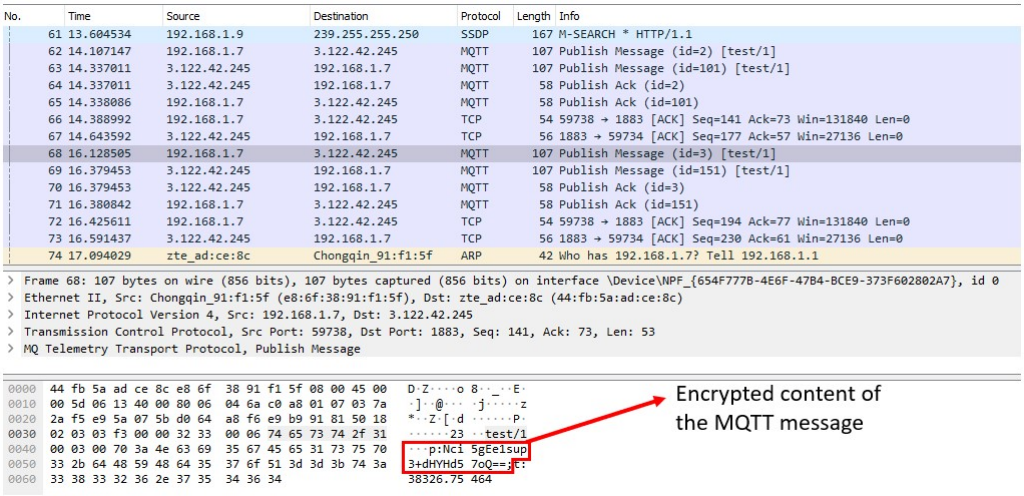

Fig. 8. The MQTT traffic protected by the AES scheme.

to the tiny differences of around 1 – 2 milliseconds. If those computational delays are accumulated with the communication delays, then there is no significant change in the total communication and computational delays. As for plaintext and TLS schemes, there is no additional computation delay because the MQTT communication process implemented in Python programming language basically handles plaintext data for both plaintext and TLS communication schemes. In addition, the TLS feature in this MQTT communication scheme is executed by the lower layer process (network kernel).

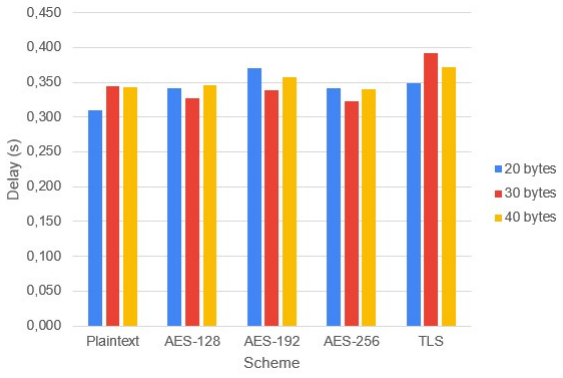

Fig. 9. Communication delays for plaintext, AES, and TLS schemes.

The results of network cost from 1 to 30 sessions for those three communication schemes are presented in Fig. 12. The network cost of the TLS scheme is the highest one among all schemes in all sessions conducted in this experiment. These results are obvious because, the TLS scheme should conduct a TLS handshake including certificate exchange and authenticated key exchange among MQTT publisher, broker, and subscriber prior to having MQTT communications. As a result, the TLS communication scheme generates a greater number of IP packets for securing the MQTT communications.

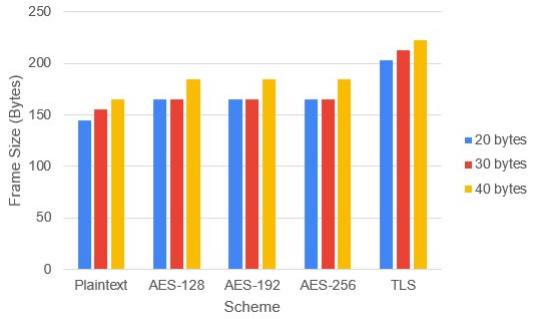

Fig. 10. Frame sizes for all communication schemes.

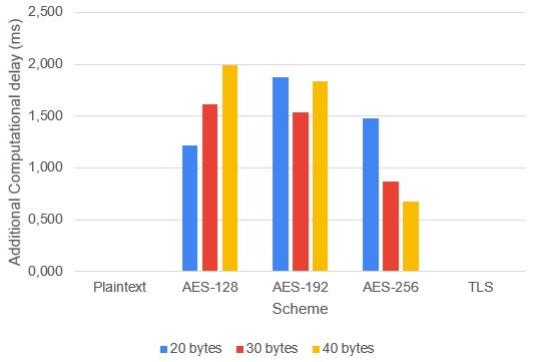

Fig. 11. Additional computational delays for three communication schemes and three payload sizes.

Eventually, based on the results in Fig. 7 to Fig. 12, the TLS communication scheme is proven to be secure, but it has the highest cost among all schemes in terms of communication delay and the number of packets in the network. Instead of using the TLS scheme, the AES cryptography-based communication scheme is preferable for devices with low computational resources considering the communication delay and network cost, which are not significantly differ with respect to the plaintext-based communication scheme in the MQTT communication. However, the use of AES-cryptography in the MQTT communication should be followed by session key usage by means of authenticated key exchange or session key scheduling so that the attackers cannot crack the security of the MQTT communication easily.

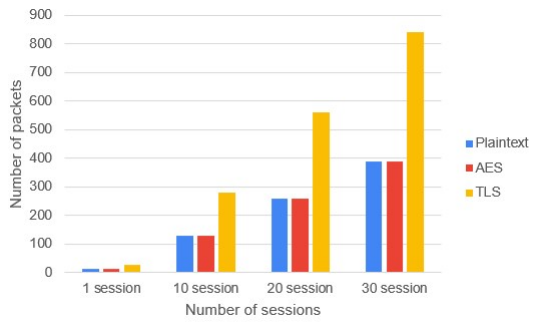

Fig. 12. Network cost for Plaintext, AES, and TLS communication schemes.

IV. Discussion

Confidentiality of the MQTT messaging is a crucial aspect of hosting reliable and secure services in the IoT. One of the discussions in hardening the security in the MQTT is utilizing the TLS approach, which is proven to be the most secure and reliable support system for internet and mobile banking transactions. However, the use of the TLS approach in IoT devices is problematic because the TLS may degrade the QoS of IoT devices with low computational resources.

Instead of using the TLS, the widely implemented AES cryptography is more appropriate for IoT devices with low computational resources. This claim is valid because it is supported by the experimental results in the previous section, which demonstrate the excellent performance of the AES approach with respect to the TLS approach in terms of network cost, frame size, and communication delay.

In addition to the AES approach, the use of lightweight-AKE methods is encouraged for increasing the level of security by dynamically changing secret keys. If by any chance the previous or current session key is successfully guessed by attackers, then the following session key is still considered to be safe within the limited time of the session key usage because attackers need some number of times to correctly guess the key. Furthermore, as long as the session time is shorter than the attackers' computation time for guessing the correct key, then the confidentiality of the information can be guaranteed during the MQTT communications.

V. Conclusion

The MQTT protocol is commonly known as the protocol for supporting devices-to-devices communications in the Internet of Things (IoT) environment. However, the MQTT protocol is not protected with a certain security mechanism except for using the TLS in the lower layer. As a consequence, MQTT communication becomes slower and burdensome for the network due to generating a greater number of IP packets. This research investigates the use of the AES cryptography-based communication scheme against the TLS-based communication scheme for providing end-to-end secure communication channels from the MQTT publishers to the MQTT subscribers. Experimental results show that the TLS-based communication scheme possesses the highest cost in terms of communication delay and network cost among all schemes in the experiment. Eventually, the AES-based MQTT communication scheme is more beneficial for the MQTT communication scheme because its communication delay and network cost are considered equal to the basic MQTT communication scheme that sends data in plaintext mode.

ACKNOWLEDGEMENT

This paper is part of the research project collaboration between School of Electrical Engineering Telkom University and Department of Architecture Engineering Diponegoro University about Secure Communication in Smart Mosque in which funded through matching fund program.

REFERENCES

[1] C. Perera, Y. Qin, J. C. Estrella, S. Reiff-Marganiec, and A. V. Vasilakos, "Fog computing for sustainable smart cities: A survey," *ACM Comput. Surv.*, vol. 50, 2017.

[2] B. Mishra and A. Kertesz, "The use of MQTT in M2M and IoT systems: A survey," *IEEE Access*, vol. 8, pp. 201071–201086, 2020.

[3] S. Seneviratne, Y. Hu, T. Nguyen, G. Lan, S. Khalifa, K. Thilakarathna, M. Hassan, and A. Seneviratne, "A survey of wearable devices and challenges," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 2573–2620, 2017.

[4] A. A. Wardana and R. S. Perdana, "Access control on internet of things based on publish/subscribe using authentication server and secure protocol," in *2018 10th International Conference on Information Technology and Electrical Engineering (ICITEE)*, 2018.

[5] G. Vrettos, E. Logaras, and E. Kalligeros, "Towards standardization of MQTT-alert-based sensor networks: Protocol structures formalization and low-end node security," in *2018 IEEE 13th International Symposium on Industrial Embedded Systems (SIES)*, 2018.

[6] S. Shin and K. Kobara, "Efficient augmented password-only authentication and key exchange for IKEv2," RFC Editor, 2012.

[7] S. Shin, K. Kobara, C.-C. Chuang, and W. Huang, "A security framework for MQTT," in *2016 IEEE Conference on Communications and Network Security (CNS)*, 2016.

[8] M. Calabretta, R. Pecori, M. Vecchio, and L. Veltri, "MQTT-Auth: a token-based solutionto endow MQTT with authentication and authorization capabilities," *Journal of Communications Software and Systems*, vol. 14, pp. 320–331, 2018.

[9] A. Velinov, A. Mileva, S. Wendzel, and W. Mazurczyk, "Covert channels in the MQTT-based internet of things," *IEEE Access*, vol. 7, pp. 161899–161915, 2019.

[10] U. Arom-oon, "An AES cryptosystem for small scale network," in *2017 Third Asian Conference on Defence Technology (ACDT)*, 2017.

[11] M. O'Neill, S. Ruoti, K. Seamons, and D. Zappala, "TLS inspection: How often and who cares?," *IEEE Internet Computing*, vol. 21, no. 3, pp. 22–29, 2017.

[12] F. Dewanta, "Secure microservices deployment for fog computing services in a remote office," in *2020 3rd International Conference on Information and Communications Technology (ICOIACT)*, 2020.

[13] X. Liu, T. Zhang, N. Hu, P. Zhang, and Y. Zhang, "The method of internet of things access and network communication based on MQTT," *Computer Communications*, vol. 153, pp. 169–176, 2020.

[14] A. Liu, A. Alqazzaz, H. Ming, and B. Dharmalingam, "Iotverif: Automatic verification of SSL/TLS certificate for IoT applications," *IEEE Access*, vol. 9, pp. 27038–27050, 2021.

[15] J. Li, R. Chen, J. Su, X. Huang, and X. Wang, "ME-TLS: Middlebox-enhanced TLS for internet-of-things devices," *IEEE Internet of Things Journal*, vol. 7, pp. 1216–1229, 2020.

[16] A. Badholia, V. Verma, and S. K. Kashyap, "Wep, Wap and Wap2 wireless network security protocol: A compact algorithm: (wireless network security protocol)," in *2019 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS)*, 2019.

[17] S. Jain, S. Pruthi, V. Yadav, and K. Sharma, "Penetration testing of wireless encryption protocols," in *2022 6th International Conference on Computing Methodologies and Communication (ICCMC)*, 2022.

[18] J.-D. Wu, Y.-M. Tseng and S.-S. Huang, "An identity-based authenticated key exchange protocol resilient to continuous key leakage," *IEEE Systems Journal*, vol. 13, pp. 3968–3979, 2019.

[19] S. Akleylek and K. Seyhan, "A probably secure Bi-GISIS based modified AKE scheme with reusable keys," *IEEE Access*, vol. 8, pp. 26210–26222, 2020.

[20] R. J. Coppen, "MQTT Version 3.1.1," 29 October 2014. [Online]. Available: https://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html. [Accessed 11 August 2022].