Implementation of Simon’s Algorithm in the Encryption Process of Publish-Subscribe Data Sending in the MQTT Protocol using the Raspberry

H Nurwarsito¹, S Yapputra²
Faculty of Computer Science, University of Brawijaya, Malang, Indonesia
¹heru@ub.ac.id, ²samudrayapputra@student.ub.ac.id

Abstract. In the development of IoT there is a higher risk of security because the device can be used as a Botnet and can harm many parties. The use of the MQTT communication protocol on the Internet of Things devices is massive. Otherwise, the MQTT protocol does not have a security side that can secure data exchange in this protocol. In information security, it is explained that data must have a security side in terms of confidentiality, integrity, and availability. One solution to address the problems that have been described is the addition of an encryption algorithm to the MQTT communication protocol so that the data that moved remains safe. The SIMON algorithm is an algorithm released by the National Security Agency (NSA) in 2013 in the selection of the best lightweight cipher at that time. The selection of this algorithm is based on the efficiency, speed, and use of small power in carrying out the process. The encryption method will be implemented at the publisher node where the node will encrypt data first before sending it to the broker node. The decryption method will be implemented on the subscriber node where the node will perform subscribe function and decrypt function to get the plaintext. The SIMON algorithm that is used is SIMON 128/128 because it has been proven that this SIMON type has the fastest time in the encryption and decryption process. Functional testing shows that all nodes in the system can work properly according to their respective functions. On the attack test, two active attack methods are used, namely Ciphertext Only Attack and Known Plaintext Attack. Both attacks were unable to get the plaintext after several attempts. So it can be concluded that the SIMON algorithm has succeeded in securing the system using the MQTT communication protocol.

1. Introduction

Internet of Things is a concept that enables communication between devices via the internet network. In this digital era, the Internet of Things is predicted to develop rapidly, which can reach 50 billion in 2020 [1]. The more development of IoT, the higher risk of security because the device can be used as a Botnet and can harm many parties [2]. The protocols used in IoT general include Hypertext Transfer Protocol (HTTP), Extensible Messaging and Presence Protocol (XMPP), Constrained Application Protocol (CoAP), Advanced Message Queuing Protocol (AMQP), and Message Queuing Telemetry Transport (MQTT) [3].

MQTT is widely used in IoT devices because MQTT can run at low bandwidth and is easy to implement considering the limited resources available on IoT devices [4]. MQTT works by consisting of 3 main nodes, namely the Publisher node, Broker node, and Subscriber node [5]. However, by default, the MQTT protocol still does not have the security that can maintain the confidentiality side of
the data exchange so that it can be read by unauthorized parties[6]. In this research, the SIMON algorithm is used to complement the security side. The SIMON algorithm has 10 different algorithms based on the block size and key size [7]. This research uses SIMON 128/128 which means it has a block size of 128 bits and a key size of 128 bits. The selection of SIMON types is based on the encryption and decryption time trials for each SIMON type with the same plaintext value. The reason for choosing the SIMON algorithm is because it is based on research showing that SIMON has advantages in terms of data encryption speed and a good level of security. This research uses the Raspberry Pi3 tool which is a Publisher node, the Broker node used is the online broker tool HiveMQ, and the laptop is used as a Subscriber node. The selection of HiveMQ is based on its capability in the role of broker and has been proven from the project with BMW industry[8]. The choice of using the Raspberry Pi3 microcontroller tool is based on the speed and ability of the Raspberry Pi which resembles a computer device [9].

The SIMON 128/128 encryption function will be implemented on the Publisher node side and the decryption function will be implemented on the Subscriber node side. So that the exchange that occurs between the Publisher node and the Broker node and the Broker node and the Subscriber node will be in the form of ciphertext. This research is expected to prove that the SIMON algorithm is one of the lightweight block cipher algorithms that are efficient and can work on the MQTT communication protocol.

2. Literature Review
With the popularity of the MQTT protocol, there are still many security aspects that need more attention [10]. The similarity of the research carried out is using the MQTT communication protocol as the test object and making the security parameters the same for testing. The difference between in this research will secure the security gaps that have been found in that research. While that research only attacked without adding a security side to the MQTT communication protocol. In securing data transmission on the system by implementing the ACORN algorithm [11]. This research shows that the time required to perform the encryption process using the ACORN algorithm is 312.26 milliseconds. The similarity in the research being carried out is using a lightweight cipher as a method of securing data in the MQTT communication protocol. The results of the SIMON algorithm can identify local similarities in the preparation of two local sequences that see the similarities between the two sequences [12]. This research shows that SIMON is a lightweight algorithm that can be used not only for security purposes. The similarity with the research being carried out is using the SIMON algorithm in the security method.

The implementation of the SIMON 32&64 algorithm for QR Code Payment can function to secure data [13]. This research shows that the SIMON algorithm is successful in securing payment data on mobile applications. The similarity to the research being carried out is using the SIMON algorithm as a method of securing encryption and decryption. Meanwhile, the differences between these studies are in the differences in communication protocols and system objects used. Recent research shows that the Lizard algorithm has been successfully implemented in the MQTT communication protocol using the NodeMCU tool with a time of 1.01 seconds [14]. This research shows that the MQTT protocol can be implemented with several algorithms that can secure the data exchange that occurs. The research carried out has a relationship and background support from each of the results of the research that has been discussed previously. The equation in this research is the use of the MQTT protocol as an object that implements the SIMON security algorithm to fulfill the security side of confidentiality and authentication. Then we will briefly explain the basic theory that is related and is the main subject related to this research. The explanation will focus on 3 main theoretical basics, namely the Internet of Things, Message Queuing Telemetry Transport, and the SIMON algorithm.

2.1. Internet of Things (IoT)
Internet of Things (IoT) is a concept to extend the benefits of permanent internet connections in areas such as data exchange, remote management, and data monitoring. IoT is the idea of researchers who want to optimize sensor equipment, radio frequency, wireless sensor networks, and all devices connected to make human work easier. Based on [15] the IoT architecture consists of 5 layers, namely,
Perception Layer, Network Layer, Middleware Layer, Application Layer, and Business Layer where each layer has different functions and purposes. The application of IoT has entered into various fields such as industry, geography, health, defense, and other fields. However, there are several problems faced by the Internet of Things developers, namely the small bandwidth that can be used, the availability of minimal resources, and guaranteed connections so that they are not interrupted. IoT devices in communicating between one device and another use a communication protocol that regulates how data is sent. One of the most widely used communication protocols on IoT devices is Message Queueing Telemetry Transport (MQTT).

2.2. Message Queuing Telemetry Transport (MQTT)
One of the most widely used communication protocols on the Internet of Things devices is Message Queueing Telemetry Transport or better known as MQTT. The reason for the many uses of the MQTT communication protocol is because it can run on low bandwidth, limited resources, and is easy to implement. One of the major companies that use the MQTT protocol is the BMW automotive company. The BMW company trusts their car-sharing application on MQTT with the help of HiveMQ.

Message Queueing Telemetry Transport (MQTT) is a communication protocol that regulates how data is exchanged using the concept of publishing and subscribe. The main concept of the MQTT protocol is divided into 3 nodes, namely the publisher node, subscriber node, and broker node [5].

The publisher node plays a role in sending/publishing data, the broker node plays a role in receiving data and storing it in the database according to the specified topic, and the subscriber node plays a role in subscribing data according to the topic.

2.3. Simon Algorithm
The Simon algorithm is a lightweight algorithm with a block cipher type where each round requires several block bits for encryption and decryption. The SIMON algorithm was released in 2013 by the National Security Agency (NSA) after winning a lightweight cipher contest and was claimed to be the best lightweight encryption algorithm.

The SIMON algorithm requires a key scheduling process to generate a key for each encryption round. The entire SIMON algorithm process requires XOR, AND, and left circular shift arithmetic logic. The SIMON algorithm has 10 types of algorithm types, which are distinguished from the bit block size and key size. The parameters used in the encryption and decryption process of the SIMON algorithm are $2n$, $mn$, $n$, $m$, const seq, and $T$. This research uses SIMON 128/128 which means that the block size is 128 bits and the key size is 128 bits also.

3. Methodology

3.1. General Description of The System
The system used in this research is a simple communication system using the MQTT communication protocol. The system consists of 3 main parts, namely the Publisher node, Broker node, and Subscriber node. Publisher functions to send input data to the communication system which will be sent and stored in the broker. The broker serves as a storage place for all data sent from Publisher and will be retrieved by the Subscriber. The subscriber is the third component in this system which functions as a client that will retrieve data/ subscribe to the Broker. The MQTT protocol requires data transmission consists of a payload packet containing topics and data. The publisher will publish the data along with the topic to the broker and the subscriber will subscribe or retrieve data from the broker according to the selected topic.

SIMON is a simple encryption algorithm because it has uncomplicated arithmetic functions and has been tested by the NSA. SIMON has 10 types of algorithms that are distinguished by the size of the block size and key size consisting of (32, 64), (48, 72), (48, 96), (64, 96), (64, 128), (96, 96), (96, 144), (128, 128), (128, 192), and (128, 256). In this research, the SIMON encryption algorithm will be added to the publisher and subscriber so that the data exchange will fulfill the side of confidentiality.
Figure 1 is an overview of the system that will be carried out in this research. It can be seen that the data input comes from the dummy data entered by the researcher with a text charset type with a maximum of 128 bits. In this research, a microcontroller with the Raspberry Pi version 3 brand was used as a publisher node which is useful for sending data to the broker. It adds an encryption function to the publisher node using the SIMON 128/128 algorithm to convert the original message (plaintext) into a secret message (ciphertext) before it is sent or published to the broker node. Before sending by publisher nodes, the topic of each data must be determined in advance on the publisher node, broker node, and subscriber node. So that each data packet sent already has its representative topic.

In this research, the node broker HiveMQ is a tool that acts as an intermediary between the publisher and the subscriber. Node broker HiveMQ After the data packet that contains the ciphertext and the length of time required for the encryption process has arrived at the broker node, the next step is for the subscriber node to connect to the broker node. The subscriber in this research uses the researcher’s laptop that has been implanted with the program so that it can execute as a subscriber node. After the subscriber node is connected and has been authenticated as an authorized party, the subscriber node can subscribe to the broker node according to a predetermined topic. Furthermore, the broker node sends a data packet containing the ciphertext and encryption time to the subscriber node according to the topic that has been subscribed. When the data packet has been received by the subscriber node, the decryption process will be carried out using the SIMON 128/128 algorithm decryption function to convert ciphertext data into plaintext. After getting the plaintext, the subscriber node will display the plaintext data and the time for the decryption process.

3.2. Publisher Node Design

The publisher nodes in this research use the Raspberry Pi3 microcontroller as a tool. Raspberry will be implanted with the program code as a Publisher node using SSH from the researcher’s laptop to the Raspberry Pi3. The following is a description of the workflow on the Publisher node. When the publisher is run, the researcher will enter the topic and the contents of the message, then the publisher will connect to the broker node.

After the publisher node is connected to the broker, the publisher will perform an encryption function to convert the original message (plaintext) into a message that has been disguised (ciphertext). After the data is converted into ciphertext, the publisher node will send the data to the broker node.

Publisher nodes will implement the SIMON 128/128 algorithm encryption function.

3.3. Broker Node Design

The Node Broker in this research uses HiveMQ software which can act as a broker on MQTT. The use of HiveMQ as a node broker is based on the recognized ability of HiveMQ because it has been used in car-sharing applications at BMW automotive companies.

The following is the workflow of the broker node, namely the Broker node can connect to Publisher and Subscriber nodes. Broker nodes can receive ciphertext data sent by Publisher nodes. Broker nodes can display the contents of the data which consists of ciphertext and the encryption process takes a long time. Broker nodes can send data to Subscriber nodes according to the topics subscribed.
3.4. Subscriber Node Design

The subscriber node in this research uses a laptop which will also implement the SIMON 128/128 algorithm decryption function. The following is the workflow of the subscriber nodes, namely the subscriber nodes that can connect to the Broker nodes. Subscriber nodes subscribe data to Broker nodes according to predetermined topics.

The first step that must be done is to make a connection between the subscriber node and the broker node. After the connection is successful, the subscriber node will subscribe to the topics on the broker node. When the data has been published by the publisher node, the data will automatically enter the broker node and can be subscribed by the subscriber node.

After the subscriber node gets the data obtained from the broker node, the subscriber runs the SIMON algorithm decryption function to convert ciphertext data into plaintext. After obtaining the plaintext value, the subscriber node will display the data along with the amount of time required to perform the decryption process.

3.5. Publisher Node Implementation

Publisher nodes use the Raspberry Pi3 tool to carry out publisher functions. The publisher node has an embedded SIMON 128/128 algorithm encryption function so that it can encrypt the input data. In implementing the publisher node on the Raspberry Pi3 device, researchers used SSH to enter the program code to the tool. Figure 5 is shown Raspberry Pi3 looks like.

Figure 2. Sensor Node Implementation

Table I. describe the parameters used in each type of SIMON are determined by the official SIMON cipher journal which contains parameters 2n as the block size, mn as the key size, n as the word size, m as the keywords, const seq is predetermined constant, and T which indicates the number of repetitions (rounds).

In Figure 3 SIMON encryption illustrates the process performed when the encryption functions every round of the SIMON algorithm. Can be seen that the encryption process begins with the input block size which is divided by 2 equal to x and y. Continues with the left circular shift process where the x value will be shifted to the left by 1, 8, and 2 bits. The values of 1, 8, and 2 are the values that have been determined by the SIMON researcher so that they cannot be changed. Then S1 will be “AND” with S8 then XOR with the value of y. Then will be XOR with S2 and the value of the first key which results in a new value as x and for the value y is the previous value of x.
Table 1. SIMON Parameters

| block size (2n) | key size (mn) | word size (n) | key words (m) | const seq | rounds (T) |
|-----------------|---------------|---------------|---------------|-----------|------------|
| 32              | 64            | 16            | 4             | z₀        | 32         |
| 48              | 72            | 24            | 3             | z₀        | 36         |
| 48              | 96            | 24            | 4             | z₁        | 36         |
| 64              | 96            | 32            | 3             | z₂        | 42         |
| 64              | 128           | 32            | 4             | z₃        | 44         |
| 96              | 96            | 48            | 2             | z₂        | 52         |
| 96              | 144           | 48            | 3             | z₃        | 54         |
| 128             | 128           | 64            | 2             | z₂        | 68         |
| 128             | 192           | 64            | 3             | z₃        | 69         |
| 128             | 256           | 64            | 4             | z₄        | 72         |

Process encryption process in the SIMON algorithm requires a key generation process called key scheduling. Key scheduling in the SIMON algorithm is a process to find the value of the key in each round. So that the use of the key will be different for each round of encryption. Key scheduling in the SIMON algorithm is divided into 3 different types according to the type of SIMON algorithm that is used. The types of key scheduling in the SIMON algorithm are m-2, m-3, and m-4 which are keywords in predetermined parameters. The key scheduling process uses XOR logic and reverses the left circular shift. Figure 4 illustrates the m2 process used in this research because it uses SIMON 128/128.

Figure 3. SIMON Encryption Process

Figure 4. M-2 Key Scheduling Process
Figure 4 shows the m2 key scheduling process to find the value of the 3rd to 68th keys. The first key is \( k_i \) (\( k_0 \)) and the second key is \( k_{i+1} \) (\( k_1 \)). The process starts with \( k_1 \), a reverse left circular shift of 3-bits \( (s-3) \) is carried out, then \( [s-3] \) is performed a 1-bit reverse left circular shift \( (s-3-1) \). The next step is to perform XOR logic of the input values \( n \), \( [s-3] \), \( [s-3-1] \), \( c \) (FFF…C), and \( z_2 \). The result of the XOR calculation of all the values above becomes the 3rd key \( (k_2) \). The key scheduling process is carried out until it finds the 68th key \( (k_{67}) \).

3.6. Broker Node Implementation

In this research, using the online broker tool HiveMQ where the broker already has the MQTT function. To connect to the broker, publisher and subscriber have a configuration to connect to HiveMQ website. HiveMQ broker can display data sent from publisher to subscriber on online mode. Figure 5 below shows the display of a broker that has been implemented and receives data from the publisher node.

3.7. Subscriber Node Implementation

The implementation at the subscriber node has several steps, namely, connecting to the broker, receiving messages from the broker, decrypting data, and displaying the plaintext results. The publisher node will be embedded with a decryption function to convert ciphertext to plaintext. The decryption process is the same as the encryption process on SIMON which uses AND logic, XOR logic, and left circular shift logic, only it does start with the input value on the last round. In other words, the decryption process in the SIMON algorithm is an inverse of the encryption process. So that the input from the first round process in the SIMON algorithm decryption process uses the encryption result value in the last round.

Figure 6 below illustrates the decryption process of the SIMON algorithm that occurs at the subscriber node. It can be seen that the process used is the opposite of the encryption process.
4. Results And Discussion

In the results and discussion chapter, we will discuss the tests that have been carried out after the implementation of the system has been built. In this research, there is a performance test for functionality and testing for system security. Functionality testing consists of testing the publisher node functionality, testing the functionality of the broker node, and testing the functionality of the subscriber nodes. The system security testing consists of testing Ciphertext Only Attack and Known Plaintext Attack.

4.1. Publisher Node Functionality Testing

This test aims to ensure that the publisher node can perform its tasks, among others, connect to the HiveMQ broker, encrypting data, and send data such as ciphertext to the broker node. Figure 9 shows that the publisher node has successfully connected and sent a message to the broker. The plaintext used is the “satu” text charset and the SIMON algorithm used is SIMON 128/128 with a block size of 128 bits and a key size of 128 bits.
4.2. Broker Node Functionality Testing
This test aims to ensure that the broker node is successful in carrying out its duties such as receiving data from the publisher node and displaying data in the form of ciphertext and the encryption time according to the topic. Figure 10 below shows that the broker node has successfully performed its functions properly. It can be seen that the interface of the broker node UI displays the data packet in the form of ciphertext and the encryption processing time. To determine the topic of the broker node, the researcher initiates the tab to the right of the subscription with the topic name “encryption/Simon”.

![Broker Receives Data from Publisher](image)

Figure 10. Broker Receives Data from Publisher

4.3. Subscriber Node Functionality Testing
This test aims to ensure that the subscriber node has successfully carried out its duties, such as connecting with the broker node, subscribing data according to the topic, decrypting the data in ciphertext form, and displaying plaintext. Figure 11 below shows that the subscriber node has successfully carried out its duties.

![Subscriber Connected, Decrypting, and Showing Data](image)

Figure 11. Subscriber Connected, Decrypting, and Showing Data

4.4. Testing Attack of Ciphertext Only Attack
This test aims to see the level of security that exists in the MQTT protocol system that has been implemented with the SIMON 128/128 algorithm. The Ciphertext Only Attack uses the Crackstation tool with the brute force method of the value to get the plaintext value. Figure 12 shows the attack.
mechanism of Ciphertext Only Attack. The Cipherhext Only Attack mechanism starts from the attacker sniffing packets on the sending side from the publisher node to the broker node. After getting the ciphertext through sniffing, the attacker will run the cracking method using the Crackstation tools.

Figure 13 shows the results of the Ciphertext Only Attack method which is failed to get the plaintext. The failure of this method is because the brute force method on the Crackstation tools does not yet have a database regarding the SIMON algorithm cracking.

**Figure 12. Ciphertext Only Attack Mechanism**

| Method | Type | Result |
|--------|------|--------|
| Ciphertext Only Attack | Based64 String XORed | Found |

4.5. **Testing Attack of Known-Plaintext Attack**

This test aims to see the level of security that exists in the MQTT protocol system that has been implemented with the SIMON 128/128 algorithm. Known Plaintext Attack attacks use the reverse XOR method to obtain plaintext values from the ciphertext data. The researcher conducted 6 attack attempts with 6 different inputs as shown in Table 1 Results of the known-plaintext Attack and Figure 14 shown the known-plaintext Attack program that using Based64 String XORed.
Table 2. Results of a known-plaintext attack

| Testing | Pieces of Plaintext | Ciphertext                  | Plaintext               |
|---------|---------------------|-----------------------------|-------------------------|
| 1       | sa                  | 676AB883836A658             | saäëaâqçé Eë|█              |
|         |                     | C1889EB4E9D773BD0           |                         |
| 2       | du                  | BA7D16DF1DBA435             | du≈│├▓             |
|         |                     | A                           | ¥R╬╕▀æ─              |
| 3       | ti                  | C37F6364382D4568            | ti╘rÅ;≥~ε          |
|         |                     | 59B9CB72649E62DF            | »|d╙ê╒╔              |
| 4       | em                  | B920E0E84E1F0D46            | em¬NÆER             |
|         |                     | CCD58DFCE76556F4            | -toggle>ëç        |
| 5       | li                  | EACF547AC9173562            | li╞O              |
|         |                     | D789F755A8DF751             | Q≤q≥yq=            |
| 6       | en                  | 99111522105D8AC56           | enΘ×"ν              |
|         |                     | E6FA5A51035D3A7             | ΑEY<ξJ/>          |

Table II shows the process when running Known Plaintext Attack attacks with plaintext input “sa” and ciphertext input “676AB883836A658C1889EB4E9D77 3BDO”. In the first experiment, it can be seen that the results obtained from the Known Plaintext Attack method failed to find the correct plaintext. The plaintext generated from the first KPA attack experiment got the plaintext result in the form of “saäëaâqçé Eë|█”. There is a “sa” in front of it because it follows the plaintext input at the beginning of the KPA cracking process with the reverse XOR process.

5. Conclusion

From the results of the research that has been done, the following conclusions can be drawn:

- Each node has functioned properly according to the functional testing that has been carried out and the SIMON 128/128 algorithm has been added to the encryption at the publisher node and decryption at the subscriber node so that the message sent is in the form of ciphertext.
- Encryption and decryption with plaintext values of “satu” are 1.009ms and 0.9ms. The SIMON algorithm has been 100 percent successful in securing data exchange that occurs on systems that are attacked by the Ciphertext Only Attack and Known Plaintext Attack methods. The system has succeeded in getting the confidentiality side by adding the SIMON algorithm.
6. Future Works
The suggestions can be applied so that the development of this research is carried out. It is necessary to add sensors as input data so that it is more diverse and the processed data is more in line with the concept of the Internet of Things. The two attacks carried out during the test failed to find the original plaintext, the researcher hopes that a method that can crack the SIMON algorithm can be found or developed.

7. References
[1] Evans D, 2011, The Internet of Things: how the next evolution of the internet is changing everything.
[2] Wanting L, Jin J and Jong-Hyouk L, 2017, Analysis of Botnet Domain Names for IoT Cybersecurity, IEEE.
[3] Naik N, 2017, Choice of Effective Messaging Protocols for IoT Systems: MQTT, CoAP, AMQP, and HTTP, Defense School of Communication and Information Systems, IEEE.
[4] Thangwei D, Ma X, Valera A, Hwee-Tian T and TAN C K, 2014, Performance Evaluation of MQTT and CoAP via a Common Middleware, IEEE.
[5] Oasis, 2014, MQTT Version 3.1.1. OASIS Standard.
[6] Wei-Tsung S, Wei-Cheng C and Chao-Chun C, An Extensible and Transparent Thing-to-Thing Security Enhancement for MQTT Protocol in IoT Environment.
[7] Beaulieu R, Treatman-Clark S, Shors D, Weeks B, Smith J and Wingers L, 2013, The SIMON and SPECK Families of Lightweight Block Ciphers, Cryptology Eprint Archive.
[8] Kashyap M, Sharma V and Gupta N, 2018, Taking MQTT and NodeMcu to IOT: Communication in Internet of Things, International Conference on Computing Intelligence and Data Science (ICCIDS 2018), PCS 132, 1611-1618.
[9] Hakim M and Putra Y, 2013, Pemanfaatan Mini PC Raspberry Pi Sebagai Pengontrol Jarak Jauh Berbasis WEB Pada Rumah, Jurusan Teknik Komputer, UNIKOM.
[10] Andy S, Rahardjo B and Hanindhitio B, 2017, Attack scenarios and security analysis of MQTT communication protocol in IoT system, EECSI, 1-6.
[11] Hananto M, Kusyanti A and Primananda R, Implementasi Algoritme Acorn untuk Pengamanan Data pada Protokol MQTT menggunakan Perangkat Wemos ESP8266, Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer
[12] Ramadanu S, 2019, Perancangan Aplikasi Pencarian File Teks dengan Menggunakan Algoritme Colussi Dan Algoritme SIMON, JURIKOM, 24-2962.
[13] Asta N, Kusyanti A and Amron K, 2019, Implementasi Algoritme SIMON untuk enkripsi dan dekripsi berbasis QR code, Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer, 10721-10728.
[14] Dismantoro R, Kusyanti A and Data M, 2018, Implementasi Algoritme Lizard untuk Pengamanan Protokol MQTT pada Perangkat NodeMCU, Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer.
[15] Anatao L, Pinto R, Reis J and Goncalves G, 2018, Requirements for Testing and Validating the Industrial Internet of Things, IEEE.