Architectural Design of Token based Authentication of MQTT Protocol in Constrained IoT Device

Adhitya Bhawiyuga, Mahendra Data, Andri Warda
Faculty of Computer Science
University of Brawijaya
Malang, Republic of Indonesia
Email: bhawiyuga@ub.ac.id, mahendra.data@ub.ac.id, andri.warda@gmail.com

***Abstract*—An effective and secure authentication mechanism is one of the important part in implementation of communication protocol in a Internet of Things (IoT) based system. As one of the popular messaging protocol in IoT world, Message Queue Telemetry Transport (MQTT) offers a basic authentication using username and password. However, this authentication method might possibly have a problem in term of security and scalability. In this paper, we propose the design and implementation of token based authentication of MQTT protocol in constrained devices. The proposed design consists of four components : publisher, subscriber, MQTT broker and token authentication server. Publisher/subscriber first sends its username password to authentication server to get the token. Notice that, the token generating process is only performed at following conditions : 1) when token has not been generated yet and 2) when token has been expired. Once publisher get a valid token, it will store that token in its local storage and use it for further authentication. From usability and performance testing result, the proposed system can perform the authentication of valid and expired token in relatively acceptable time.**

I. Introduction

Recently, the world has seen a significant progress on both research and implementation of Internet of Things (IoT) concept. In an IoT point of view, a system is composed by several devices equipped with sensor or actuator, communication adapter, and microprocessor with limited processing and storage capabilities[1]. Once those devices are connected, they are expected to either exchange their sensor gathered data or send those data to the cloud based data center for further processing [3]. In order to accommodate that requirement, several communication protocols in physical, transport, network and application layers have been invented and standardized among researchers.

One of the popular application layer protocol in IoT world is Message Queue Telemetry Transport or simply called MQTT. As difference with HTTP, the MQTT is a TCP based messaging protocol with publish subscribe architecture. There exists three kind of actors in publish subscribe architecture : publisher, subscriber and broker [4]. Publisher sends the message identified by a specific topic to broker which then forward that message to every subscriber who are interesting to that particular topic. In this case, the subscriber does not need to know from whom the message are originating from while the publisher does not care to whom its message are relayed. This kind of architecture is suitable for IoT case since it can provide a more data oriented protocol which can reduce the burden of a constrained device for sending/receiving message. However, since both publisher and subscriber does not know each other, an authentication method is highly required to ensure the validity of sender and receiver node.

In order to perform that kind of validation, several MQTT broker software provides username-password as its basic authentication mechanism. With this method, the publisher or subscriber need to send its username and password during the connection establishment phase. While it can provide a relatively good basic security feature, the username-password scheme may faces several problems. First, since the publisher/subscriber needs to always sends its credential during connection establishment phase, it could be easier for passive attacker to acquire that credential using a wireless sniffing mechanism. In addition, there is no expiration concept, which means, once the attacker get the credentials, he/she can use it forever as long as the credential remains unchanged. To avoid that problem, the broker can store a session information related to a valid device. Therefore, the client does not need to resend its credential during connection. However, it may adds an additional burden for the broker to store those session information especially if there exists a vast amount of publisher/subscriber in a system. In this case, an authentication that utilize the client storage for storing session information is required to cope with both aforementioned problems.

Taking account those challenging problems, we propose the design and implementation of token based authentication of MQTT protocol in constrained devices. The proposed design consists of four components : publisher, subscriber, MQTT broker and token authentication server. Publisher/subscriber first sends its username password to authentication server to get the token. Notice that, the token generating process is only performed at following conditions : 1) when token has not been generated yet and 2) when token has been expired. Once publisher get a valid token, it will store that token in its local storage and use it for further authentication. Therefore, the broker does not need to store the session on its database while avoiding the publisher/subscriber to periodically sends its credential. In this paper, we utilize the JSON Web Token (JWT) authentication server due to its compact message size. Furthermore, its message is self-contained which means it contains all required information about a user avoiding the requirement to store and query the user information in broker

978-1-5386-3546-9/17/$31.00 ©2017 IEEE

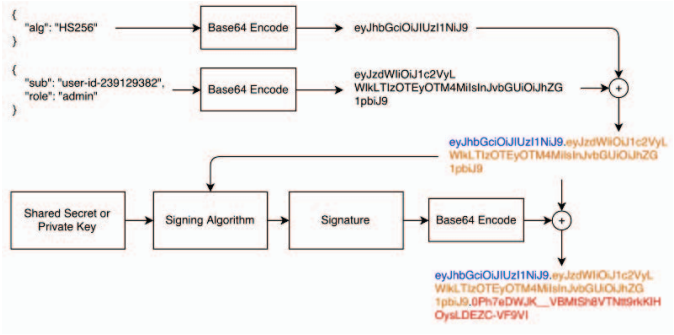

Fig. 1. Illustration of JWT Token Request.

Listing 1. Header part of JWT Token Request

```
{
"alg": "HS256",
"typ": "JWT"
}
```

database. From the functional and performance testing, we have successfully design and implement the token based authentication on constrained device i.e. NodeMCU board equipped with 128KB memory as publisher/subscriber and Raspberry Pi as broker.

The remainder of this paper is organized as follows. We first explain our preliminary study on IoT authentication at section II. Then, in section III, we describe the design and implementation of our proposed system. We then discuss the testing result of proposed system on in section IV. Finally, we conclude our paper at section V.

II. Preliminary Study

A. Preliminary Study on JWT

JSON Web Token (JWT) is an open standard for authentication defined in RFC 7519. This standard proposes compact and self-contained way for securely transmitting data between two communicating actors using a JSON formatted text. In general, there exists two main actors : client and authentication server. The client first perform a token request by sending a credential i.e. username and password. The authentication server then query on its database to perform credential validity checking. Once it is valid, the server send back its response containing a valid token which can be used for further authentication.

The token itself contains three main components : header, payload and signature. The header is in JSON format and contains the hashing algorithm used for signature verification as shown in Listing 1. In following part, the payload contains

Listing 2. Payload part of JWT Token Request

```
{
"name": "Alice",
"admin": true
}
```

Listing 3. Signature part of JWT Token Request

```
HMACSHA256(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload),
  secret)
```

user information as shown in Listing 2. With those informations, the authentication server does not need to store any user session in its memory or database since any information related to a user is self-contained in payload part. In order to ensure that the token is not changed along the transmission, the JWT communicating entities can add a signature which is formed by hashing three part : header, payload and a secret using HMAC SHA256 algorithm. The example of a signature is presented in Listing 3. Both of header and payload are encoded using Base64 and appended in "*header.payload*" sequence. The example of overall JWT Token request is illustrated in Fig. 1

B. Related Work on IoT Authentication

The security becomes one of important issue in the IoT world [8], [2]. In literature, there exists several work related to the authentication methods used in IoT world. Those works can be classified into two main categories : the cloud and device authentication. In first category, the author of [7] an authentication method personal cloud environment using Oauth method. In another work, the author of [6] proposes the design of authorization system in MQTT using Oauth with username-password authentication method. While the work on first category emphasize the authentication method in cloude, the work in second category put underline on the authentication in device part. The author of [9], [10] proposed the identity based authentication in wireless sensor network environment. As far as we are aware of, there is no previous work which deals with the implementation of token based authentication in both the device and server/cloud part.

III. System Design

Fig. 2 illustrates the general architecture of proposed system. The proposed design consists of four components : publisher, subscriber, MQTT broker and JWT authentication server. In an MQTT-based system, publisher has a role for producing information consumed by subscriber. In between the publisher and subscriber, there exists a broker as relay entity. To ensure the authenticity of publisher and subscriber, we introduce JWT server whose role is to release a token and check its validity against any authentication request.

In detail, the sequence diagram and flowchart of proposed system is illustrated in Fig. 3 and Fig. 4, respectively. Publisher/subscriber first sends its credential i.e. username and password to authentication server for obtaining a JWT token. Once the authentication server checks the requested credentials against its database, it then return a valid JWT token containing the header, payload and signature as described in section II. While a publisher/subscribe is going to publish/subscribe a

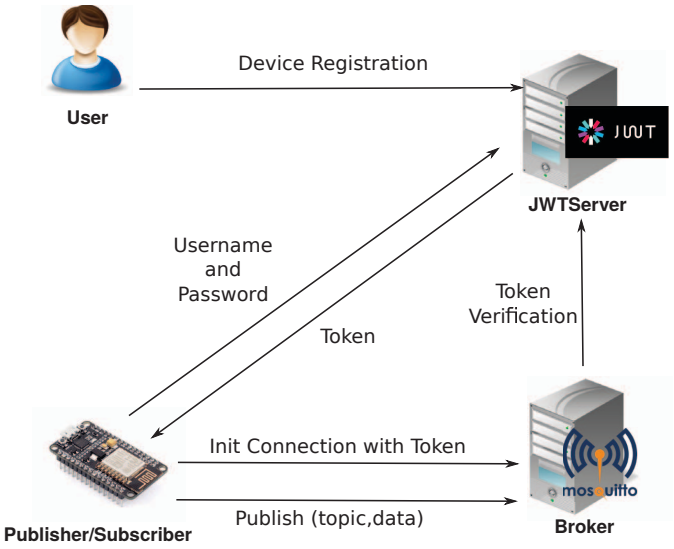

Fig. 2. System Architecture.

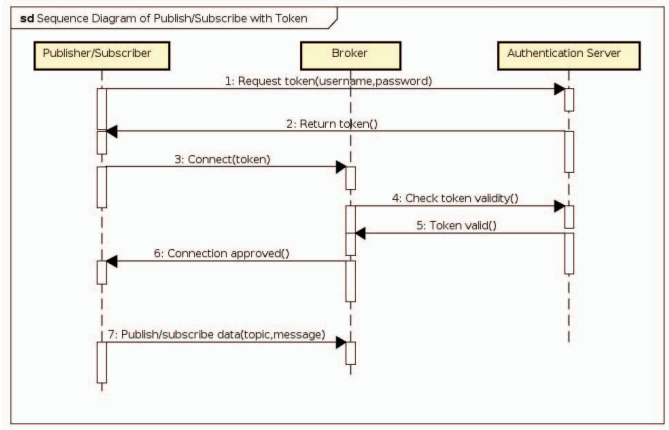

Fig. 3. Sequence Diagram of Publish/Subscribe with Token.

data, it firsts initiate a connection to broker using the obtained token. The broker then checks the validity of requested token to authentication server. One it is performed, the publisher/-subscriber is able to connect and publish/subscribe to a specific topic of information.

In this proposed system, the token generating process is only performed at following conditions : 1) when token has not been generated yet and 2) when token has been expired. Once publisher get a valid token, it will store that token in its local storage. The stored token are then sent to broker during the connection initialization. Upon reception, broker validates the received token to authentication server. If token is valid, the publisher/subscriber is then allowed to publish the sensor data or subscribe to a specific topic.

IV. Result and Discussion

In this section, we present the testing of our proposed system. In general, we perform two testing : functional testing to measure the usability of proposed system and performance testing to measure the performance of proposed system.

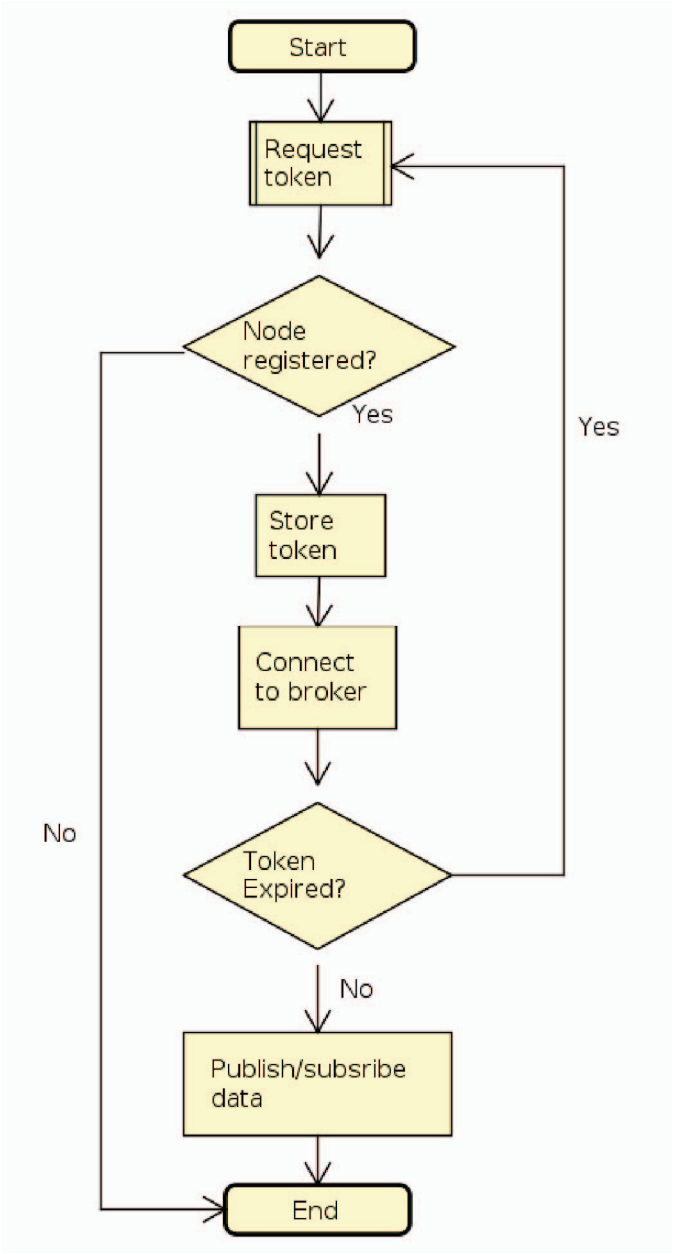

Fig. 4. Flowchart of Proposed Authentication System.

A. Functional Testing

The functional testing is performed by running the overall system with several scenarios including : authentication with valid and expired token.

1) Authentication with Valid Token: In this scenario, we try to send the valid token generated from the authentication server. Fig. 5 and 6 shows the server and broker response on a valid token, respectively. From those figures, we can conclude that both the broker and authentication server can perform a correct validation upon the right token request.

From this testing we can conclude that the broker can successfully accept the connection request from publisher with a valid token.

2) Authentication with Expired Token: In this scenario, we try to send the already expired token previously generated

```
1496562954: |-- ** checking backend jwt
1496562954: |-- url=http://127.0.0.1:3000/auth
1496562954: |-- data=topic=&acc=-1&clientid=
1496562954: |-- getuser(eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6IkZ
JTEtPTV8xIiwiaWQiOiI1OGU1Zjc2MDFkNDFjODA0YWZjMTg0NTAiLCJpYXQiOjE0OTY1NjI5NDUsImV
4cCI6MTQ5NjU2MzEyNX0.rTlUiCF2AnJhr1beWP4zKYxCV49-6tivOUwETHsdkes) AUTHENTICATED=
1 by jwt
```

Fig. 5. Broker response on valid token.

```
  exp: 1496563125 }
user { _id: 58e5f7601d41c804afc18450,
  email: 'wildandolen@gmail.com',
  username: 'wildanhabibi',
  password: 'pbkdf2_sha256$30000$S10ylPVjEzNE$hr3Bemowjgks4a/XepDPG89PG5n6jo1hgg
WmSZsgW1U=',
  first_name: 'Moh. Wildan',
  last_name: 'Habibi',
  is_admin: 0 }
wildanhabibi/eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6IkZJTEtPTV8xIi
wiaWQiOiI1OGU1Zjc2MDFkNDFjODA0YWZjMTg0NTAiLCJpYXQiOjE0OTY1NjI5NDUsImV4cCI6MTQ5Nj
U2MzEyNX0.rTlUiCF2AnJhr1beWP4zKYxCV49-6tivOUwETHsdkes
jwt decoded { username: 'FILKOM_1',
  id: '58e5f7601d41c804afc18450',
  iat: 1496562945,
  exp: 1496563125 }
user { _id: 58e5f7601d41c804afc18450,
  email: 'wildandolen@gmail.com',
  username: 'wildanhabibi',
  password: 'pbkdf2_sha256$30000$S10ylPVjEzNE$hr3Bemowjgks4a/XepDPG89PG5n6jo1hgg
WmSZsgW1U=',
  first_name: 'Moh. Wildan',
  last_name: 'Habibi',
  is_admin: 0 }
```

Fig. 6. Authentication Server response on valid token.

from the authentication server. Fig. 7 and 8 shows the server and broker response on a invalid token, respectively. From those figures, we can conclude that both the broker and authentication server can reject the authentication request if the client sends an expired token.

From this testing we can conclude that the broker can successfully reject the connection request from publisher with an expired token.

B. Performance Testing

The aim of this testing is to measure the performance of proposed system in term of time required by publisher/subscriber to get the token from authentication server. In this case, the time is calculated from the NodeMCU device send the request until it get the response from authentication server. Fig. 9

```
1496563188: |-- ** checking backend jwt
1496563188: |-- url=http://127.0.0.1:3000/auth
1496563188: |-- data=topic=&acc=-1&clientid=
1496563188: |-- getuser(eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6IkZ
JTEtPTV8xIiwiaWQiOiI1OGU1Zjc2MDFkNDFjODA0YWZjMTg0NTAiLCJpYXQiOjE0OTY1NjI5NDUsImV
4cCI6MTQ5NjU2MzEyNX0.rTlUiCF2AnJhr1beWP4zKYxCV49-6tivOUwETHsdkes) AUTHENTICATED=
0 by none
```

Fig. 7. Broker response on expired token.

```
wildanhabibi/eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6IkZJTEtPTV8xIi
wiaWQiOiI1OGU1Zjc2MDFkNDFjODA0YWZjMTg0NTAiLCJpYXQiOjE0OTY1NjI5NDUsImV4cCI6MTQ5Nj
U2MzEyNX0.rTlUiCF2AnJhr1beWP4zKYxCV49-6tivOUwETHsdkes
jwt decoded { username: 'FILKOM_1',
  id: '58e5f7601d41c804afc18450',
  iat: 1496562945,
  exp: 1496563125 }
user { _id: 58e5f7601d41c804afc18450,
  email: 'wildandolen@gmail.com',
  username: 'wildanhabibi',
  password: 'pbkdf2_sha256$30000$S10ylPVjEzNE$hr3Bemowjgks4a/XepDPG89PG5n6jo1hgg
WmSZsgW1U=',
  first_name: 'Moh. Wildan',
  last_name: 'Habibi',
  is_admin: 0 }
wildanhabibi/eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6IkZJTEtPTV8xIi
wiaWQiOiI1OGU1Zjc2MDFkNDFjODA0YWZjMTg0NTAiLCJpYXQiOjE0OTY1NjI5NDUsImV4cCI6MTQ5Nj
U2MzEyNX0.rTlUiCF2AnJhr1beWP4zKYxCV49-6tivOUwETHsdkes
invalid token!
```

Fig. 8. Authentication Server response on expired token.

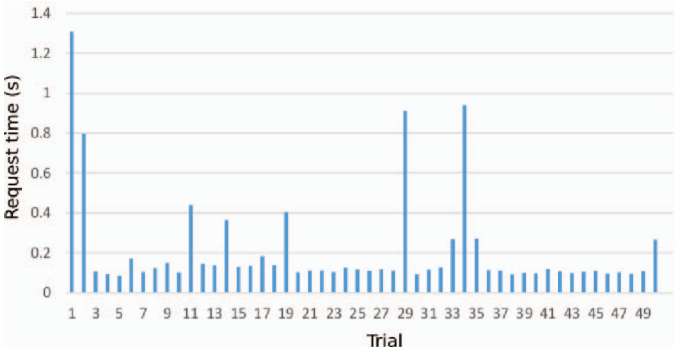

Fig. 9. Performance testing result.

shows the result of this testing in second. From the result, we can conclude that even though there is some fluctuation, the average time is still below 1 second which is acceptable for this kind of system.

V. Conclusion

In this paper, we designed and implemented the token based authentication system in MQTT protocol. The proposed design consists of four components : publisher, subscriber, MQTT broker and token authentication server. Publisher/subscriber first sends its username password to authentication server to get the token. From usability and performance testing result, the proposed system can perform the authentication of valid and expired token in acceptable time i.e. less than 1 seconds in average.

Acknowledgement

This work has been funded by Information Centric Networking Research Group (ICN-RG), Faculty of Computer Science, University of Brawijaya.

References

[1] Atzori, L., Iera, A., & Morabito, G. (2010). The Internet of Thinags: A survey. Computer Networks, 54(15), 27872805.

[2] Granjal, J., Monteiro, E., & Sa Silva, J. (2015). Security for the Internet of Things: A Survey of Existing Protocols and Open Research Issues. IEEE Communications Surveys & Tutorials, 17(3), 12941312.

[3] Al-fuqaha, A., Member, S., Guizani, M., Mohammadi, M., & Member, S. (2015). Internet of Things : A Survey on Enabling, 17(4), 23472376.

[4] Banks, Andrew, and Rahul Gupta. "MQTT Version 3.1. 1." OASIS standard 29 (2014).

[5] Sebastin E. Peyrott. "JWT Handbook". Auth0. 2017.

[6] Niruntasukrat, Aimaschana, et al. "Authorization mechanism for mqtt-based internet of things."*Communications Workshops (ICC), 2016 IEEE International Conference on. IEEE*, 2016.

[7] McCarthy, Dnal, et al. "Personal cloudlets: implementing a user-centric datastore with privacy aware access control for cloud-based data platforms." *Proceedings of the First International Workshop on TEchnical and LEgal aspects of data pRIvacy. IEEE Press*, 2015.

[8] Pawar, Ankush B., and Shashikant Ghumbre. "A survey on IoT applications, security challenges and counter measures." Computing, Analytics and Security Trends (CAST), International Conference on. IEEE, 2016.

[9] Yussoff, Yusnani Mohd, Habibah Hashim, and Mohd Dani Baba. "Identity-based trusted authentication in wireless sensor network." arXiv preprint arXiv:1207.6185 (2012).

[10] Nguyen, Tien Dung, and Eui-Nam Huh. "A Dynamic ID-Based Authentication Scheme for M2M Communication of Healthcare Systems." Int. Arab J. Inf. Technol. 9.6 (2012): 511-519.