International Journal of Enhanced Research in Science, Technology & Engineering
ISSN: 2319-7463, Vol. 11 Issue 4, April-2022, Impact Factor: 7.957

MQTT Security Challenges and Solutions

Prashant Verma

MS in Computer Science, Georgia Institute of Technology, Atlanta, Georgia, USA
MBA, Haas School of Business, University of California, Berkeley, California, USA
Senior Member IEEE, Member ACM

ABSTRACT

It is important for us to understand what MQTT security is and how it works in helping achieve secure communication between a client and a broker. In other words, it is a communication mechanism making machine to machine communication possible, but the end-users cannot interact with the servers directly, instead the client sets up a secret and a secure password which is authenticated by an MQTT server to identify the client's identity.

Although this mechanism is known to be safe, but the users have paid very little attention towards the security of the system. Due to limited resources and a complex environment system of MQTT, it is difficult to achieve the required security measures. There are different forms of attacks involved in the MQTT security such as a replay attack when data is transmitted from nodes. In this form of attack if the traffic between the client and the broker is intercepted by a person, the hacker can send repeated data to the broker, tricking it to assume that it is new information coming from an IoT device on the network.

Another severe security challenge faced by MQTT security is the Man in the Middle Attack (MITM), in which a third person gets involved between the client and the broker during their communication and the information can be modified even before it is reached to its destination. This is considered as a devastating security challenge, even worse than DDoS attack or eavesdropping. Traditional security technology makes it difficult to tack the MIMT attacks.

A DDoS attack against MQTT refers to a huge amount of publishing, subscribing, and connecting messages that eventually exhaust node resources and make the node unable to provide normal services. DDoS defense through SSL/TLS is not suitable for IoT nodes due to its resource consumption.

We have discussed different security challenges earlier, which compromises the MQTT security mechanism, but such attacks can be avoided with various IoT protocols and MQTT algorithms, which we will discuss later within this article.

Keywords — Client, broker, MQTT security, DoS, IoT

1. INTRODUCTION

In today's era of technology, the concept of IoT is taking over the world. From smart watches to bank transactions, IoT systems and devices are making peoples day to day lives easier and fast. With such devices, the amount of data is immensely increasing, and it is becoming difficult to secure such a large amount of data. It is very important to understand the fact that if such devices and the information is breached, this could incur heavy losses to the industry and the people associated with them as well.

Since the IoT systems operate with less memory, small processors and low power supply the Message Queuing Telemetry Transport (MQTT) is considered as one of the best security protocols for IoT systems since, the MQTT systems itself require less resources and small bandwidth.

2. ABOUT MQTT

MQTT protocol was developed in 1999 by Andy Stanford Clark and Arlen Nipper. It was designed to be lightweight, consume less power, and work with small devices until IBM took control of it and launched new versions.

A. Approaches to MQTT security

The MQTT protocol is present at the application layer. The application layer is responsible for providing an interface between a network and the IoT devices. If there should arise an occurrence of IOT gadgets, the application layer might be carried out either by the running working framework or by the firmware.

The most well-known and universal application layer convention utilized is Hypertext Transfer Protocol (HTTP). The HTTP convention has been intended for correspondence between a client and a server. The convention utilizes demand/reaction technique in which when a client needs information from the server then it sends a solicitation message to the server and server sends back a reaction message. HTTP is applied where just a single end can begin the information correspondence. That is, the client starts information solicitation and server answers it. The convention was intended to deliver application layer execution for PCs. It can in any case be utilized by IOT gadgets, be that as it may, IOT applications with requirement assets face constraints with the HTTP convention.

3. AUTHENTICATION WITH USERNAME AND PASSWORD

As discussed earlier, the clients interact with the broker directly through TCP/IP protocols using SSL/TLS channels with a trusted network. Although, SSL/TLS approach is not used as a standard approach towards client identification and authorization. The authorization process can also be achieved by a username and password.

B. Working

There is a possibility that a broker may require a client to have a valid username and a password before establishing a connection with them. In order to establish that connection, the client may send a CONNECT package to the broker, which helps the client and the broker to establish a connection with each other. Once that connection is established, the communication can start between the clients and the broker. The communication can be stated as one-to-many or one-to-one depending on the number of clients. The clients will only receive messages that are related to their chosen topic and in case a client wishes to check if the connection is still established or not with the broker, a ping request is sent to the broker known as a PINGREQ packet.

In case a client wants to end their communication with the broker, they can simply send a DISCONNECT message to end their session. Finally, in the case of interruption in communication, the client receives the unsent messages again by the agent once they are back online.

C. Some Common MQTTmethods

- CONNECT
- CONNACK
- PUBLISH
- PUBACK
- PUBREC
- PUBREL
- PUBCOMP
- SUBSCRIBE
- SUBACK
- UNSUBSCRIBE
- UNSUBACK
- PINGREQ
- PINGRESP
- DISCONNECT

D. MQTT Quality of Service Options

1) **Send only once –** Message is sent once to all the clients, but it is unsure if the clients receive the message or not
2) **Send atleast once –** Message is sent to the clients multiple times but they might receive a single message multiple times
3) **Send once and ensure transmission –** The message is only sent once to all the clients, but it is made mandatory to ensure that the message is received by all the clients

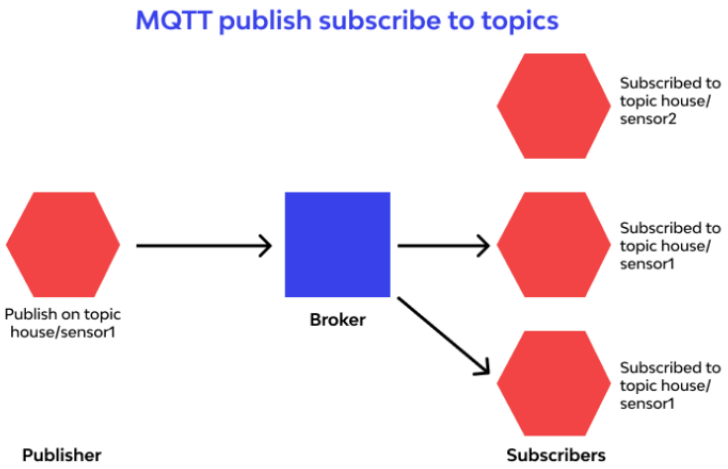

Figure 1. A sample diagram of working of MQTT systems, Publish/Subscribe model

4. HOW TO SECURE MQTT

E. Advanced Authentication Mechanisms

Client IDs: Client IDs are unique ID's that are given to each client when the client gets connected to the broker. But the ids are not just related to connecting with a broker, in fact, a client id is related to your subscription on a certain topic. Once the client chooses a certain subscription of a topic and connects to the broker, a unique id is linked to the topic. The broker can easily remember the client id and topic subscription of the client through persistent connections.

As an MQTT broker, you can either choose a certain name or an ID, which is unimportant for you as long as you can easily remember it for communication purposes. To provide further client security, the mosquito broker allows you to choose a certain prefix for your name, which you can use to easily identify your topic and subscription.

F. X509 Client Certificates

In terms of cryptography, a client certificate is used for making authenticated requests to a certain server. Similarly, in MQTT system, the client certificates make authenticated requests to the clients. Such an authentication form/method is used only for high level security requirements for a small number of clients. You can either issue an individual certificate to every client, which makes it difficult to implement as well but there is also an option for sharing a single certificate among various clients.

- **require_certificates –** Main setting tells client it needs to supply a certificate when set to true. Defaultfalse
- **use_identity_as_username**– – When set to true it tells mosquito not to use the password file but to take the username from the certificate (common name given to certificate when you create it). Default false
- **crlfile** – You can create a certificate revocation file that is used to revoke a client certificate.

5. AUTHORIZATION

Authorization refers to the access that the clients may have in accordance to their subscriptions within the MQTT system. Before diving into the working of authorization, let us discuss 3 important terms associated.

- Subject is known as the user who wants to have authorized access
- Object is that resource that needs to be accessed and protected from unauthorized access
- Policy defines if the certain user has access or not

Now, the subject or a user is the MQTT client who wishes to subscribe a certain topic. The object is that certain topic that a client wishes to subscribe, and the policies are a set of rules set by the broker where he either allows or denies access. Without proper authorization, any client can subscribe any topic or publish their own content, so the broker needs to implement certain permissions on their side. If a client is denied access by the broker, they cannot publish or subscribe to any content. If a client publishes a topic without broker's permission, the broker can either disconnect the client or ask them to use PUBACK OR PUBEL packets for normal fashioned publishing. The broker also reserves all the rights to accept or deny the subscription of a client on a certain topic. It is best advised to subscribe to topics in accordance to the associated client id.

6. TLS/SSL

TLS and SSL are known as Transport Layer Security and Secure Sockets Layer respectively. They are used to provide a smooth communication channel among the clients and servers. Although the MQTT system implies on TCP transport protocol but since it does not use an encrypted communication, the MQTT brokers rely more on the TLS protocol. Even when you are wanting to acquire authentication through username and password using the CONNECT packets, TLS is considered as the best mechanism. In simple words, the TLS will provide you with an encrypted pipe through which your messages can flow securely.

Ports 1883 or 8883 is a standard MQTT secured connection. x509 client certificates are also an example of using TLS/SSL for authentication purposes.

SECURING MQTT SYSTEMS

Let us discuss about deploying MQTT systems in a secured and controlled environment. For that purpose, we will be discussing different layers of security to prevent attacks in the most efficient manner.

G. Infrastructure

- A firewall should be passed by a broker in accordance to certain rules and authorized permissions as discussedearlier
- Since MQTT uses TCP, we can ignore all the UDP datagram packets
- Block traffic in extra ports except the 1883 and8883 ports
- The concept of load balancer is to distribute MQTTtraffic among multiple brokers although this is not considered very efficient.

H. Operating System

We already know that the MQTT brokers are installed on a server and run on the application layer. There is an obvious chance that your operating system or software's are vulnerable of attacks. So it is important to:

- Keep your software's and data libraries up to date
- Install brute force attack detectors such as Fail2ban
- Security enhanced Linux for Linux based operating systems

I. MQTT Brokers

- Authentication and authorization with certain set of rules on subject, object and policy basis
- TLS/SSL protocols for secure and encrypted communication between client and server
- Maximum message size of 256 MBs so you shouldn't overload the system

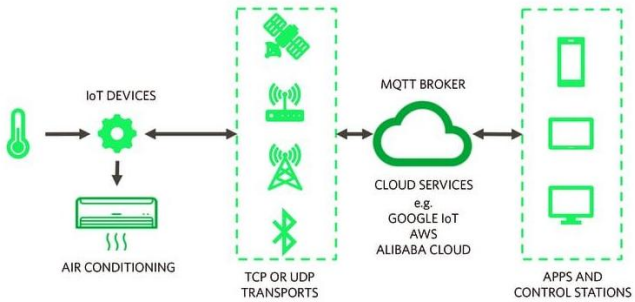

Figure 2. Diagram reperesentation of MQTT security layers

7. FURTHER SECURITY CONCEPTS AND IMPLEMENTATIONS WITH MQTT

J. OAuth 2.0 and MQTT

OAuth 2.0 is an authorization framework, in which the clients can access a certain resource owned by another person without providing their private credentials. To understand the working of this framework, we need to see the different roles that are involved with OAuth 2.0:

- **Resource Server** – Resource is provided only to the authorized user containing an access token
- **Resource Owner** – The person who owns the resource
- **Client** – The person who wishes to access that certain resource
- **Authorization Server** – Central server controlling all resources and token issuing and revoking

K. OAuth 2.0 Grant Types

Four different flows can be used to get a token from the authorization server:

- **Authorization Code** – Designed for websites, only for special clients
- **Implicit** – Designed for public clients with interaction between authorization server and resource owner
- **Resource OwnerCredentials** - Enter your secret credentials to access and refresh token from an authorization server
- **Client Credentials** – Authenticate himself by any means to access token in return

Once a client has accessed a token, he can send the token to the broker through CONNECT message. Once the broker gets the token, he can check its expiration date, validity of signature and revoke through authorization server.

L. Payload Encryption

Payload encryption is an application specific encryption in which the broker is not involved. This means that this provides end-to-end encryption of data and can be used in an untrusted environment as well. Since it is application specific, all the PUBLISH metadata stays in its original form and only the payload data is encrypted.

M. Message Data Integrity

It is vital to check the honesty of messages that are shipped off the specialist from untrusted MQTT clients or MQTT clients that you don't control (particularly on the off chance that you don't utilize TLS). Information uprightness checks let you guarantee that outsiders changed no satisfied of your MQTT messages.

MQTT PUBLISH parcels can contain an advanced mark/MAC/checksum that confirms the items in the bundle. This determined stamp is normally added to the payload (for instance, toward the start of the payload). The recipient of the bundle can check the trustworthiness of the information byrecalculating/approving the stamp. This approval guarantees that the message was not altered by a noxious outsider.

Components for making and approving stamps There are three well known ways of making stamps:

- Checksum/Hash Algorithms (for instance, MD5, CRC, SHA1/2)
- Macintosh (for instance, HMAC, CBC-MAC)
- Advanced marks

These components have their assets and shortcomings.

Data Integrity: The recipient can make sure that the data was not modified (accidentally).

Authentication: The recipient can make sure that the message originates from a trusted sender because only trusted parties have access to the key that is required to create and verify the stamp.

Non-Repudiation: Only the sender of the message with access to the private key can create the stamp. Other parties can verify the signature with the public key but they cannot create the stamp themselves.

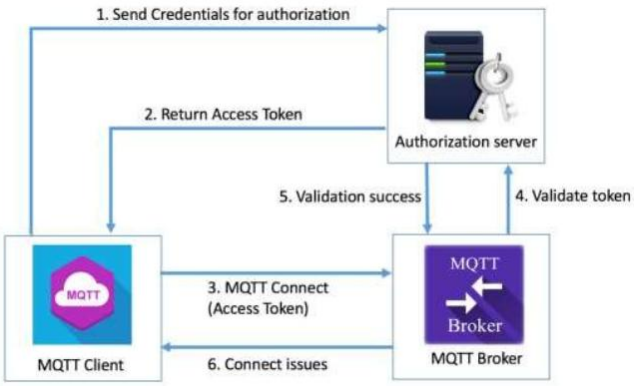

Figure 3. OAuth 2.0 authorization model for MQTT server

CONCLUSION

With a day by day increases and advancements in technology as well as the amount of data, it is very important to understand that security is a very important aspect.

We have discussed how the MQTT and IoT systems interact with each other to ensure sustainable security and an easy life for the potential users.

ACKNOWLEDGEMENTS

Place your acknowledgements before References. Do not mention the sponsors and/or financial support obtained in this section, as they must be included in an unnumbered footnote on the first page of the paper. You have to delete the footnote on the first page if there is no sponsorship information to add.

REFERENCES

[1]. Emrah Atilgan, Iiker Ozcelig, Esra. N Yolacan. MQTT security at a glance. RKEY 978-1-6654-0776-2/21/$31.00 ©2021 IEEE
[2]. Fu Chen, Jianming Zhu, Yujia Huo, Dan Fan. A Review on the study of MQTT security challenges. 978-1-7281-6547-9/20/$31.00 ©2020 IEEE
[3]. (2020) The MQTT website. [Online]. Available: https://mqtt.org/
[4]. (14 JULY 2020) MQTT security fundamentals in industrial automation [Online]. Available: https://www.industrialshields.com/blog/arduino-industrial-1/post/mqtt-security-fundamentals-in-industrial-automation-235
[5]. MQTT: The enabler of smooth and hassle free information exchange for an IOT system. [Online]. Available: https://www.wallarm.com/what/mqtt-concept
[6]. (2019) Introduction to MQTT protocol for IOT applications. [Online]. Available: https://www.concurrency.com/blog/june-2019/introduction-to-mqtt-protocol-for-iot-applications
[7]. (M Priya) Understanding MQTT protocol: IOT part 14. [Online]. Available: https://www.engineersgarage.com/understanding-mqtt-protocol-iot-part-14/#:~:text=Message%20Queue%20Telemetry%20Transport%20(MQTT,usually%20implemented%20by%20the%20browser.
[8]. (27 SEPTEMBER 2021) Mosquitto username and password authentication – Confiiguration and testing. [Online]. Available:http://www.steves-internet-guide.com/mqtt-username-password-example/#:~:text=Username%20and%20password%20authentication%20is,use%20TLS%20to%20secure%20it
[9]. (2 JANUARY 2021) Creating and using client certificates with MQTT and Mosquito. [Online]. Available: http://www.steves-internet-guide.com/creating-and-using-client-certificates-with-mqtt-and-mosquitto/
[10]. (12 FEBRUARY 2021) Introduction tpo MQTT security mechanisms. [Online]. Available: http://www.steves-internet-guide.com/mqtt-security-mechanisms/
[11]. (2018) OAuth based Secured authentication mechanism for IoT Applications. [Online]. Available: https://www.researchgate.net/publication/337731224_OAuth_based_Secured_authentication_mechanism_for_IoT_Applications
[12]. Authenticating & Authorizing Devices using MQTT with Auth0. [Online]. Available: https://auth0.com/docs/customize/integrations/authenticate-devices-using-mqtt
[13]. (2019) Why MQTT is everywhere and the issues it faces. [Online]. Available: https://blog.paessler.com/why-mqtt-is-everywhere-and-the-security-issues-it-faces