International Journal of Applied Engineering Research ISSN 0973-4562 Volume 19, Number 1 (2024) pp. 7-10
https://dx.doi.org/10.37622/IJAER/19.1.2024.007-010

A Review: MQTT Architecture and Security and Application-Level Communication Approaches

G. Umasankar and Dr. N. Baskar

1Research Scholar, Department of Computer Science,
2Associate Professor, Department of Computer Science
Sri Ramakrishna Mission Vidyalaya College of Arts and Science.

Coimbatore-20.

Abstract

In the Evolution of the internet communication is significantly increased in recent decades. In this Evolution Internet of Things mostly use the device-to-device combination approach (M2M communication. In the IOT used many protocols for communication the data in network environment. In the IOT devices required lightweight and internet support protocols. And we here we saw the one of the IOT based protocol.

Keywords IOT, M2M. Device Communication. MQTT Security, MQTT Architecture.

INTRODUCTION

MQTT stands for Message Queuing Telemetry Transport Protocol. it was developed in the late 1990's as one of the SCADA protocols. It was used mainly for industrial automation and is, as its name suggests, for transporting short telemetry data messages. There is no standard for the format of data it transports Here we saw the MQTT (Message Queuing Telemetry Transport) it's leading protocol for IOT communication.

MQTT is a data communication protocol utilized in an IoT context that operates on top of TCP. IBM invented the protocol as an inexpensive machine-to-machine (M2M) interaction technique, which was subsequently recognized by OASIS [1]. Message Queuing Telemetry Transport (MQTT) is a lightweight messaging and bi-directional communication protocol. It can scale to connect millions of IOT devices. And it supports unreliable networks and reliable message delivery. And it works based on the publish-subscribe model.

MQTT Protocol has some feature advantage there are

- Low overhead and bandwidth
- Flexible and scalable
- Easy to implement and use

And also has some drawback for in real time communication

Challenges there are

- Security issues
- Limited features and standardization
- Network dependency

MQTT ARCHITECTURE

The MQTT protocol is run over a TCP connection. And it's worked based on the publish and subscribe mechanism. It's a bi-directional communication approach. In comparison with traditional HTTP. Both are run over a TCP connection. But the HTTP response only comes when there is a request from the client.

The MQTT has two components for broker and client. The broker work is similar to the HTTP server. But the difference is once client connected its worked bi -directional communication. The Traditional http is work based on URL and the MQTT is work based on Topics.

A. MQTT Components

1. **Broker:** The MQTT broker is the backend system which coordinates messages between the different clients.[2] Responsibilities of the broker include receiving and filtering messages, identifying clients subscribed to each message, and sending them the messages. It is also responsible for other tasks such as:

   a. Authorizing and authenticating MQTT clients

   b. Passing messages to other systems for further analysis

   c. Handling missed messages and client sessions

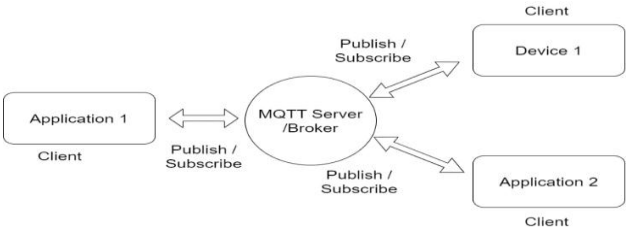


Figure 1 MQTT communication Model

2. **Client**: An MQTT client is any device from a server to a microcontroller that runs an MQTT library. If the client is sending messages, it acts as a publisher, and if it is receiving messages, it acts as a receiver. Basically, any device that communicates using MQTT over a network can be called an MQTT client device

3. **MQTT Connection:** Clients and brokers begin communicating by using an MQTT connection. Clients initiate the connection by sending a *CONNECT* message to the MQTT broker. The broker confirms that a connection has been established by responding with a *CONNACK* message. Both the MQTT client and the broker require a TCP/IP stack to communicate. Clients never connect with each other, only with the broker.

B. How it's Works: An overview of how MQTT works is given below. An MQTT client establishes a connection with the MQTT broker. Once connected, the client can either publish messages, subscribe to specific messages, or do both. When the MQTT broker receives a message, it forwards it to subscribers who are interested. Let's break down the details for further understanding.

MQTT Topic: The term 'topic' refers to keywords the MQTT broker uses to filter messages for the MQTT clients. Topics are organized hierarchically, similar to a file or folder directory. For example, consider a smart home system operating in a multilevel house that has different smart devices on each floor. In that case, the MQTT broker may organize topics as:

Our home/ground floor/living room/light

Our home/first floor/kitchen/temperature

MQTT Publish

MQTT clients publish messages that contain the topic and data in byte format. The client determines the data format such as text data, binary data, XML, or JSON files. For example, a lamp in the smart home system may publish a message *on* for the topic

Living room/light.

MQTT Subscribe

MQTT clients send a SUBSCRIBE message to the MQTT broker, to receive messages on topics of interest. This message contains a unique identifier and a list of subscriptions. For example, the smart home app on your phone wants to display how many lights are on in your house. It will subscribe to the topic *light* and increase the counter for all *on* messages.

Publishers and subscribers both use MQTT clients to send and receive messages. When an MQTT client wants to publish a message to a topic, it sends a PUBLISH packet specifying the desired QoS level.

Let's study more about the different QoS levels and what they mean now that we have a better understanding of MQTT QoS.

MQTT QoS Levels: MQTT QoS has three levels'

QoS 0: at most once

QoS 1: at least once

QoS 2: exactly once

QoS	Publisher	Subscriber
0	Will send a message only once.	Might receive or might not receive the message.
1	Will send a message at least once as long as an acknowledgement is received or the command to end the transmission is received.	It is likely to receive the message at least once (it is possible that the message can be received more than once).
2	Will only send a message once.	Will only receive the message once.[11]

MQTT SECURITY LEVEL AND SECURITY CHALLENGES

The MQTT main goal is to provide a lightweight and easy-to-use communication protocol for the Internet of Things. The protocol itself specifies only a few security mechanisms. Here is a high-level summary of security levels in MQTT.[3]

Network Level: Using a physically secure network or VPN for all client and broker communications is one way to provide a secure and reliable connection. This technique works well for gateway applications in which the gateway is connected to devices on the one hand and to the broker over a VPN on the other.

Transport Level: In data Communication confidentiality is a priority, TLS/SSL is usually used to encrypt the transport. This method is a secure and proven way to ensure that data cannot be read in transit and provides a client certificate to verify the identity of both parties.

Application Level: On the transport level, communication is encrypted and identities are authenticated. The MQTT protocol provides a client identifier and username/password credentials to authenticate devices on the application level. These properties are provided by the protocol itself. Authorization or control of what each device is allowed to do is defined by the specific broker implementation. Additionally, it is possible to use payload encryption on the application level to secure the transmitted information (without the need for full-fledged transport encryption).

Security Challenges in MQTT

A 2018 IOT Security Foundation survey concluded that less than 10% of consumer IOT companies follow vulnerability disclosure guidelines. [4].

The research from Avast (LSE: AVST), the global leader in cyber security products, found more than 49,000 Message Queuing Telemetry Transport (MQTT) servers publicly visible on the internet due to a misconfigured MQTT protocol. This includes more than 32,000 servers with no password protection, putting them at risk of leaking data. [5]

A new IDC forecast estimates that there will be 41.6 billion connected IOT devices, or "things," generating 79.4 zetta bytes (ZB) of data in 2025. [6].

How to Secure MQTT Application: A recent Palo Alto Networks' data sheet titled IOT Security says, "98% of all IOT device traffic is unencrypted [7], exposing personal and confidential data on the network. Together with 57% of IOT

devices also being vulnerable to medium- or high-severity attacks, this makes IOT a low-hanging fruit for attackers."

First thing we need to ensure the data encryption in MQTT application. Configure MQTT brokers and clients with a robust firewall, allowing protocol-specific inbound ports 1883 or 8883, or use a private network or VPN for IOT devices [10].

Authentication and Authorization is most common security approach for internet-based applications.

Authentication is the process of verifying the users' using credentials. In this process for create a user identity for the server and client. In this user identity has contains the roles and permissions.

In the authorization process its check the user identity based to control access rights by granting or denying specific permission to an authenticated user [9]

MQTT brokers with the right authorization policies for clients and limit their ability to subscribe and publish MQTT messages. [8]

In my review with mosquito MQTT broker support and allow username and password authentication is network. Based on the MQTT password security file configuration. But it's not enough to include more devices and users. while being able to integrate virtually any external system like OAuth 2.0 (JWT), Lightweight Directory Access Protocol (LDAP), or any type of database (SQL or no-SQL). Its good options to communication with enterprise level IOT environment.

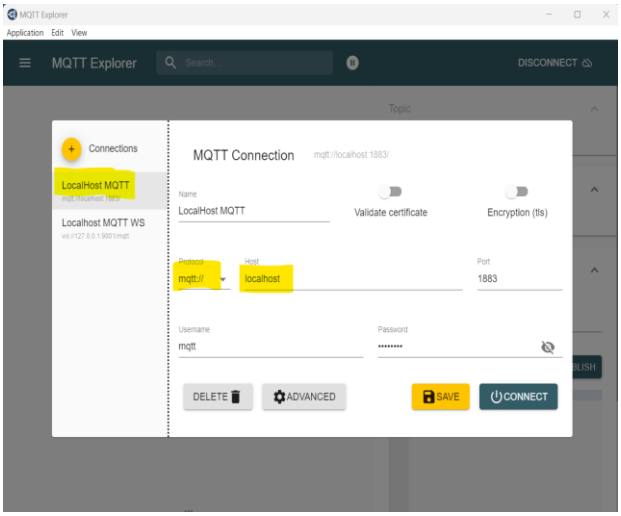
APPLICATION-LEVEL COMMUNICATION APPROACHES

In my analysis for the basic level implementation is MQTT broker support two types of protocol for communication. One is MQTT and second one is MQTT over Web Sockets.

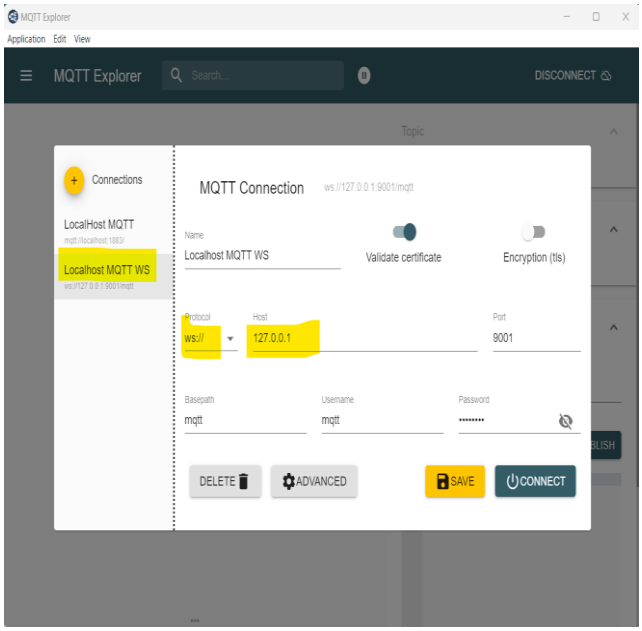
In the native applications are using for MQTT Protocol and the web-based application support MQTT over Web Sockets.

I was checking the communication with the basic authentication. If two are more clients connected in the same broker.one of the client is connected using the native MQTT protocol and Second Client Connect using the MQTT over Web socket Protocol. In this testing I performed using the MQTT Explorer. And basic integration performed with the MQTT native (MQTT Protocol) and web-based java script library (MQTT over Web Socket).

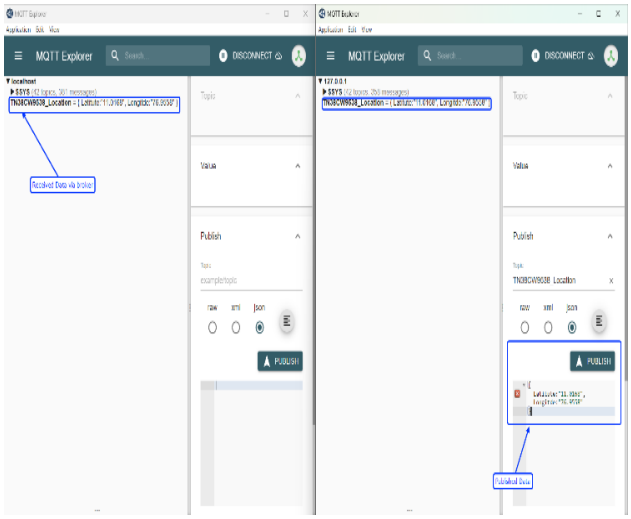
The MQTT Explorer is One tool for perform this both Protocols. Each protocol is communicating to the broker. Once it's received the request via MQTT Topics is publish the data to other clients to connect which are the client and subscribed to the particular topics.



Connecting via MQTT protocol



Connecting Via MQTT over Web Socket



Perform the MQTT Communication with MQTT and MQTT Over Web Socket.

Futher Work

In my analysis based The MQTT broker support MQTT and web socket and it support only basic level authentication in based on username and password. In web based approached required dynamic and RBAC and devices also support OAUTH based Client credentials flow support for the enterprise level device communication. So, in my further analysis and implementation is based on the Role based access and client credentials approaches for the Devices communications. In one our pervious paper is based on OAuth based external authentication in web external login approach [9]. We will check and integrate RBAC in web users and the client credential flow for the native or devices communication.

CONCLUSION

In this paper presents the MQTT architecture, communication and analysis the security approaches and further work. In IOT is a trending technology. So, it's required the security environment. So, our further work is based on the OAUTH. How to secure the MQTT application/ server for the enterprise's usage.

REFERENCES

[1] S. M. Metev and V. P. Veiko, *Laser Assisted Microtechnology*, 2nd ed., R. M. Osgood, Jr., Ed. Berlin, Germany: Springer-Verlag, 1998.

[2] https://aws.amazon.com/what-is/mqtt/#:~:text=The%20MQTT%20broker%20is%20the,and%20sending%20them%20the%20messages.

[3] https://www.hivemq.com/blog/introducing-the-mqtt-security-fundamentals/

[4] https://iotsecurityfoundation.org/less-than-10-of-consumer-iot-companies-follow-vulnerability-disclosure-guidelines/

[5] https://press.avast.com/avast-research-finds-at-least-32000-smart-homes-and-businesses-at-risk-of-leaking-data

[6] https://www.businesswire.com/news/home/20190618005012/en/The-Growth-in-Connected-IoT-Devices-is-Expected-to-Generate-79.4ZB-of-Data-in-2025-According-to-a-New-IDC-Forecast

[7] https://www.paloaltonetworks.co.uk/cyberpedia/what-is-iot-security#:~:text=In%20addition%20to%20these%20challenges,the%20network%20to%20outside%20risks.

[8] https://www.hivemq.com/blog/mqtt-security-fundamentals-authorization/#:~:text=This%20kind%20of%20topic%20permission,certain%20quality%20of%20service%20level.

[9] https://www.irjet.net/archives/V5/i5/IRJET-V5I5188.pdf

[10] https://www.hivemq.com/article/ultimate-guide-on-how-to-use-mqtt-with-node-js/#:~:text=The%20two%20most%20commonly%20used, Layer%2FTransport%20Layer%20Security).

[11] https://cedalo.com/blog/understanding-mqtt-qos/