A Trust-Based Approach for Data Sharing in the MQTT Environment

Liang Chen*, Stilianos Vidalis* and Su Yang†
*Department of Computer Science, University of Hertfordshire, Hatfield, UK
†Department of Computer Science, Swansea University, Swansea, UK

2023 20th Annual International Conference on Privacy, Security and Trust (PST) | 979-8-3503-1387-1/23/$31.00 ©2023 IEEE | DOI: 10.1109/PST58708.2023.10320147

***Abstract*—Internet of Things (IoT) is considered as a giant network of connected devices who collect data and share them with each other. There has been extensive developments on IoT standards and protocols that enable IoT devices to exchange data in a structured and meaningful way. Message Queuing Telemetry Transport (MQTT) is one of such developments receiving widely adoption for industrial applications. It is designed as a lightweight messaging protocol based on the publish-subscribe model by which clients publish messages to a broker who is responsible for distributing the messages to subscribed clients. MQTT is often deployed in a hostile environment in which IoT devices and brokers are vulnerable to attacks. While security for MQTT has received great attention, it does not adequately address the authorisation issues within a decentralised MQTT environment. Existing work adopts policy-based approaches to regulate data sharing across multiple brokers, which we believe, are unlikely to scale well. In this paper we propose a trust-based approach that can be easily incorporated into the existing implementation of MQTT broker. We introduce a way of computing trust rating of brokers and develop two means of using the trust ratings to control data flow across multiple broker domains. Our approach is capable of detecting and blocking malicious clients and brokers from sending false or malicious messages into the system.**

I. Introduction

The Internet of Things (IoT) represents the vast amount of devices that connect to the Internet to exchange information in real time. In general, any device that is capable to send and receive data over a network is considered to be an IoT device. This includes laptops, smartphones, TV, thermostats, but also resource-constrained sensors, microcontrollers and actuators. IoT solutions are enabling new ways to improve efficiency, flexibility, and productivity, which is evident from the buzzword "smart" appearing in the front of key areas: city, roads, homes, farming, health and logistic depots.

There has been extensive developments on IoT standards and protocols that enable IoT devices to exchange data in a structured and meaningful way. One of the most widely adopted protocols in IoT is *Message Queuing Telemetry Transport* (MQTT) [1]. MQTT is a lightweight and scalable messaging protocol, designed specifically for connecting a large scale of resource-constrained devices. It works on the principles of the publish-subscribe model to decouple the message sender (publisher) from the message receiver (subscriber). Instead, a third component, called a broker, filters all incoming messages from publishers and distribute them correctly to subscribers.

The more devices are connected to the Internet, the more attractive the data becomes for cyber attacks. The MQTT protocol specifies a few security mechanisms such as authentication of users and devices, authorisation between clients and the broker, integrity and confidentiality of message packets, but the way of implementing these mechanisms is up to application developers [1]. In this paper we have specifically examine the *authorisation solutions* provided by the modern implementations of MQTT brokers[1]. It turns out that access control lists and OAuth2.0 are among the most popular ones. Their implementation guidelines, however, introduce these authorisation solutions only for a single broker context, from which it is not obvious how they can be extended to a decentralised MQTT environment where multiple brokers connecting to each other for data sharing. Typically, such environment exhibits the following characteristics:

- The resource-constrained IoT devices running at hostile environments are vulnerable to security attacks such as becoming malware spreading false information into the IoT network.
- Data are often shared among devices belonging to different security domains controlled by different local brokers. In other words, one device may share data with another who is not known in advance.

Existing work use a policy-based approach (either imposing information flow policies [2] or attribute-based authorisation policies along with user preference [3]) to regulate data sharing in such a decentralised MQTT environment, which we believe, are unlikely to scale well and suffer from policy administrative burden. Trust management appears to be a natural approach, but as far as we are aware, has not been proposed yet. Such considerations are the focus of this paper. More specifically, we summarise our contributions as follows:

- We take the principle of OAuth2.0 to introduce the concept of *authorisation token* that is a set of permissions granted to IoT devices for publishing and subscribing to message topics. This token-based approach for authorisation is compliant with existing MQTT implementations.
- We propose a subjective trust model to compute the trustworthiness of brokers in terms of proper management of their devices and authorisation policies. When a broker receives a message publishing request along with its authorisation token, it determines whether to accept the token by evaluating the trustworthiness of brokers who signed the token.

[1]MQTT software: https://mqtt.org/software/

- We propose two ways of evaluating the acceptance of an authorisation token throughout the broker network, reflecting different degrees of security we can enforce for the decentralised MQTT environment.

II. Background

In this section we describe relevant background materials on the MQTT and subjective trust assessment.

A. MQTT

MQTT is the most popular messaging protocol for the Internet of Things (IoT), being used in a wide variety of industries ranging from automotive, smart home, logistics to manufacturing. It employs the *publish–subscribe* architecture for messaging and data exchange between IoT devices. An IoT device (named client) send (publish) messages on a *topic* to a server (named broker) that is responsible for distributing the messages to clients who previously subscribed for the given topic. Fig. 1 shows a star topology of the MQTT publish-subscribe architecture in which clients are decoupled from each other and the connection between them is handled by the broker.

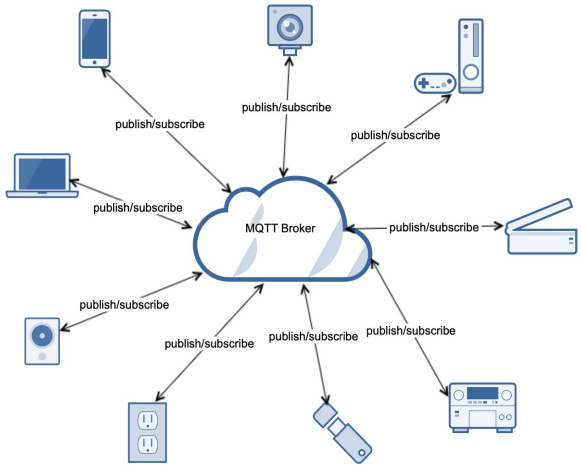

Fig. 1. MQTT publish-subscribe architecture

An interesting concept in MQTT is the *topic filter* which is an UTF-8 string that the broker uses to filter messages for each connected client. The topic filter consists of one or more topic levels and each topic level is separated by a forward slash. An example of a topic filter is $home/+/bedroom/\#$, where both $+$ and $\#$ are *wildcard* covering single level and multiple levels respectively. Note that a client can use wildcard to subscribe to multiple level of topics. For example, if a client is subscribed to topics: $home/+/bedroom/\#$, it means that it can listen to all messages regarding to the bedroom at all floors ($+$) and everything that is in there ($\#$). On publishing messages, a client can send a message (21.5°C) under a *specific* topic ($home/secondfloor/bedroom/temperature$) that is used by the broker to forward the message to all subscribed clients. It is important to observe that publishing clients cannot use wildcard characters in the topic names they publish.

MQTT provides some important features including Quality of Service (QoS) which gives clients the power to choose a level of service that matches its network reliability and application logic. The message broker manages the re-transmission of messages and guarantees delivery according to the levels clients chosen. A detailed study of features provided by the latest version of MQTT (MQTT v5.0) can be referred to the standard document [1].

B. Trust Model

While many more complex trust models exist, we adapt Josang's widely used Subjective Logic based approach [4], which is a relatively straightforward model grounded on Bayesian principles.

An opinion held by an agent x about agent y regarding issue i is a tuple $\omega^x_{y:i} = \langle \alpha^x_{y:i}, \beta^x_{y:i}, \gamma^x_{y:i}, \delta^x_{y:i} \rangle$, where $\alpha^x_{y:i} + \beta^x_{y:i} + \gamma^x_{y:i} = 1$ and $\delta^x_{y:i} \in [0, 1]$. The values of $\alpha^x_{y:i}$, $\beta^x_{y:i}$ and $\gamma^x_{y:i}$ represent the degrees of *belief*, *disbelief*, and *uncertainty* regarding the proposition that the agent y will behave as x expects with respect to issue i. The base rate parameter $\delta^x_{y:i}$ represents the *a priori* degree of trust that the agent x concerned has about y, before any direct evidence has been acquired.

Opinions are formed and updated using binary assessments of past performance: $r^x_{y:i}$, being the positive experiences with y regarding issue i; and $s^x_{y:i}$, being those assessed as negative. An agent x's *opinion* may then be computed: $\alpha^x_{y:i} = r^x_{y:i}/(r^x_{y:i} + s^x_{y:i} + 2)$; $\beta^x_{y:i} = s^x_{y:i}/(r^x_{y:i} + s^x_{y:i} + 2)$; and $\gamma^x_{y:i} = 2/(r^x_{y:i} + s^x_{y:i} + 2)$. For an initial opinion with no evidence, therefore, $\alpha^x_{y:i} = 0$, $\beta^x_{y:i} = 0$, $\gamma^x_{y:i} = 1$, and $\delta^x_{y:i}$ is typically set to 0.5. Given an opinion computed about y regarding i, a single-valued *trust rating*, which can be used to rank and compare individuals, can be obtained: $\tau^x_{y:i} = \alpha^x_{y:i} + \delta^x_{y:i} \cdot \gamma^x_{y:i}$.

III. Our Trust-based Approach

A. Authorisation Tokens

An MQTT client can basically do two things after it has connected to a broker: publish messages and subscribe to topics. Without proper authorisation, each authenticated client can publish and subscribe to all available topics, including malicious clients. Since MQTT is often deployed in a hostile communication environment, the standard advocates the provision of an authorisation mechanism that is capable of restricting clients to publishing and subscribing to only *authorised* topics.

The MQTT clients may be resource-constrained devices and have limited computation power, thus an authorisation mechanism needs to be implemented on the broker side. Such a mechanism would provide an authorisation decision by evaluating an access request with respect to an *authorisation policies*. To define authorisation policies, we use a capability-based approach which has been extensively studied for authorisation in the IoT context [5]. Also, the latest implementation of authentication and authorisation for MQTT [6] advocates the

use of OAuth 2.0[2], which is a capability based authorisation delegation protocol.

We follow the principle of OAuth 2.0 to assume the existence of an *authorisation server* (AS) who is responsible for registering clients and managing authentication and authorisation policies on behalf of a broker. When a client registers with AS, AS will issue an *authorisation token* (AT) which defines a set of permissions granted to the client. In other words, an authorisation token simply enumerates all authorised requests with respect to the client. In this paper we do not focus on how AS authenticate a client for it to issue an AT, which depends on many different factors such as client ID, trustworthiness, location of client, client owner' policies, etc. Fig. 2 shows the authentication and authorisation flow for a client connecting to a broker. To simplify our discussions for multiple brokers case later, we say that AT is issued by a broker instead of AS from now on.

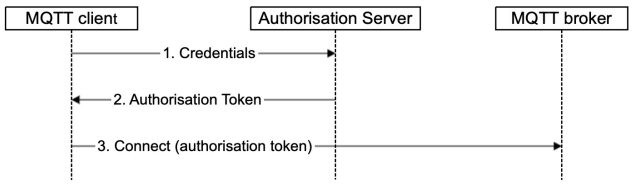

Fig. 2. Client authorisation flow in MQTT

Let F be a set of topic filters. Given two topic filters, $f, f' \in F$, we write $f \subseteq f'$ if the topics contained in f is a subset of topics contained in f'. For example, let $f = home/+/bedroom/\#$ and $f' = home/\#$, we have $f \subset f'$. Let C be a set of clients. We define an AT as a tuple $\langle b, c, S, F_p, F_s \rangle$, where b is a broker issuing the AT, c is the client who receives the token, S is a set of brokers who digitally sign the token, and F_p is a set of topic filters that c is permitted to publish, and F_s is a set of topic filters to which c is permitted to subscribe. In a single broker environment, $S = \{b\}$, which means AT is issued and signed by the single broker b, but the structure we defined is for a general case where AT may be signed by multiple brokers when it goes through the broker network. We study the general case in the next section. Let us now take an example of $AT = \langle b, c, \{b\}, F_p, F_s \rangle$, where $F_s = \{home/firstfloor/\#\}$, and $F_p = \{home/groundfloor/kitchen\}$, suggesting that client c is authorised to publish a message on the topic $(home/groundfloor/kitchen)$ and to subscribe to all topics under $home/firstfloor$.

Given a request from client c to publish a topic message $f{:}m$, denoted by $req_p(c, f{:}m)$, we say that $req_p(c, f{:}m)$ is granted by broker b if c holds an $AT = \langle b, c, \{b\}, F_p, F_s \rangle$ and there exists $f' \in F_p$ such that $f = f'$. Given a request from client c for subscribing to topics f, denoted by $req_s(c, f)$, we say that $req_s(c, f)$ is granted by broker b if c holds an $AT = \langle b, c, \{b\}, F_p, F_s \rangle$ and there exists $f' \in F_s$ such that $f \subseteq f'$.

[2]OAuth 2.0: https://oauth.net/2/

B. Messaging Across A Broker Network

In a complex IoT deployment, brokers are connecting with each other to form a broker network, in order to facilitate data sharing between clients across different domains. Our approach to controlling data sharing across a broker network is simple with little change of the original MQTT specification.

Fig. 3 shows an example of a broker network in which red nodes represents clients and green nodes represent brokers. Broker b_1 connects to two brokers b_2 and b_3, and b_3 connect to b_4 and b_5, which is reflected as edges in the network. We assume that each client can only be registered with one broker at a time. After authenticating with the broker, the client receives an AT from that broker which defines the topics it is allowed to publish and subscribe to. When a client publishes a message on a topic, the client may not aware that the message will be shared by the broker to clients in other domains, because clients are authorised to publish a message on a *topic*, not concerning with the receivers. That is the principle of *decoupling* with the publish-subscribe model.

For two connected brokers, they register with each other and receive an AT from each other. For example, when b_2 registers with b_1, b_1 treats b_2 as one of its clients and issues an AT defining the topics to which b_2 is permitted to subscribe, which is in the form of $\langle b_1, b_2, S, \emptyset, F_s \rangle$. It would be the same semantics when b_1 registers with b_2. Note that the AT a broker receives only contain *subscription permissions* not publishing, that is because the role of broker is to manage authorisation and to forward messages to its authorised clients.

Given a broker b, we have set up two-way message flow through b. One is from neighbouring brokers who forward "inbound" messages to which b is subscribed. The other way is that b forwards "outbound" messages to its neighbouring brokers on the topics they have subscribed to. With our approach, a message may go through one broker to another, enabling the communications of clients in different domains.

We now use the example in Fig. 3 to explain how each broker b makes a decision on whether to distribute an arriving message to its subscribed clients and brokers. When broker b_1 receives a request of publishing a message from one of its clients c, it checks the request against the token AT held by c. If the token is valid and permits the request, the message is forwarded to its subscribed clients at b_1. Note that these requests are evaluated according to the mechanism we defined in Section III-A. If broker b_2 is subscribed to the topic associated with the message, b_1 forwards the request along with the AT to b_2. Broker b_2 treats the request and AT as if coming from its own clients but perform specific operations for checking the *acceptance* of AT as follows:

1) Evaluate validity of AT by verifying the signature of b_1 on the token;
2) Compute trust rating of $trust(b_1, b_2)$ which represents b_2's opinion on the trustworthiness of b_1 in term of proper management of its authorisation policies;

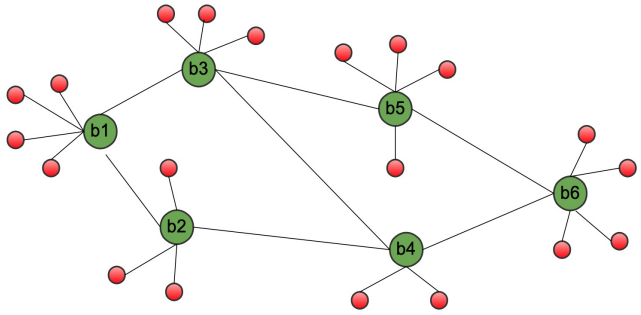

Fig. 3. Example of a broker network

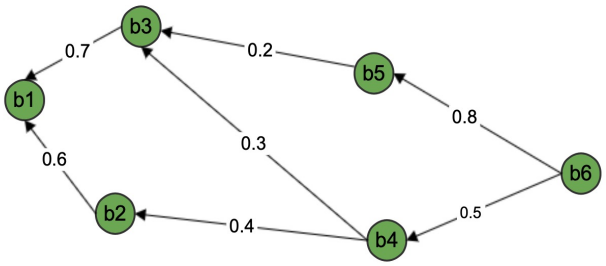

Fig. 4. Example of a weighted graph

3) If $trust(b_1, b_2) \geqslant \pi$, where π is a *trust threshold* defined globally by the system, then b_2 signs the AT. This acts as an "introducer" of b_1 to other b_2 connected brokers;
4) Distribute the message to its subscribed client and forward the request and updated AT to its connected brokers b_4 other than b_1.

Assume that broker b_4 is also subscribed to the message. When b_4 receives the request and AT, it performs the same operations as described above, but if $trust(b_2, b_4) < \pi$, should b_4 reject the request, or accept and forward it further down the network? Before exploring this question, let us first figure out how to assess brokers' trustworthiness.

C. Computing Trust Ratings

The *trustworthiness* of brokers is a building block for determining whether an AT is acceptable. Our approach to evaluating broker's trustworthiness is able to detect and eliminate malicious or compromised brokers in a network.

The trust rating of a broker b_i by broker b_j, denoted by $trust(b_i, b_j)$, represents how much b_j trusts b_i to manage authorisation for its clients properly. This is reflected by b_j's previous interactions with b_i regarding to the "quality" of data being received from b_i. Let H_i^j be a history of message publication events held by broker b_j, whose members are the form of $\langle b_i, m, f\rangle$, representing that a message m on the topic f was authorised to be forwarded from broker b_i. Given a history H_i^j, broker b_j may evaluate history events: $E_j : H_i^j \to \{\mathsf{pos}, \mathsf{neg}\}$. Through this function, therefore, an observed event $\langle b_i, m, f\rangle \in H_i^j$ may feed positive or negative trust updates. If b_2 deems the message m being good quality, which means it was indeed the information that b_j is expected to receive, then it results in a positive update, while a negative update otherwise. One of our prioritised future work is to specify how E_j works, which depends on many factors. An obvious one may be the direct feedback from b_j' clients who subscribed to the topic message $f{:}m$. These feedback may be supplied by the owners whose IoT devices (clients) deployed at broker b_j.

Given E_j function, we like to capture a situation in which trust increases gradually by positive experience in interactions and drops significantly by a negative experience. For each $h \in H_i^j$ such that $E_j(h) = \mathsf{pos}$, we have $r_i^j = r_i^j + 1$. When there exists $h \in H_i^j$ such that $E_j(h) = \mathsf{neg}$, we make $s_i^j = s_i^j + \mu$, where $\mu \in \mathbb{Z}$ and $\mu > 1$. The choice of μ makes s_i^j grows quicker when a negative interaction occurs. Broker b_j may periodically calculate $trust(b_i, b_j) = \alpha_i^j + \delta_i^j \cdot \gamma_i^j$, where $\alpha_i^j, \gamma_i^j, \delta_i^j$ are calculated based on r_i^j and s_i^j.

D. Making Decisions

In Section III-B, we have not explicitly presented how a broker makes a decision on whether a publishing request and AT is acceptable. A straightforward approach is for the broker to check whether its neighbouring brokers have signed the token and evaluate the trustworthiness score of these brokers. Formally, we construct a directed acyclic weighted graph $G = \langle B, E, T\rangle$, where B is a set of brokers, and $T : E \to [0, 1]$ is a function mapping directed edges to their trust rating. Given an edge $e = (b, b') \in E$, b is said to be *adjacent to* b' and b' is said to be *adjacent from* b. Given a broker $b \in B$, we write $\overrightarrow{A}(b) = \{b' \in B : (b', b) \in E\}$ to represent a set of brokers that are adjacent to b. Essentially, $T(e) = trust(b, b')$ computing the dynamic trust rating of b by b'. Given an $AT = \langle b, c, S, F_p, F_S\rangle$, each broker $b' \in B$, who receives AT, would accept it if there exists $b'' \in \overrightarrow{A}(b')$ such that $b'' \in S$ and $trust(b'', b') \geqslant \pi$, and reject it otherwise. It simply means that the broker b' would accept the token AT only if one of his trustworthy neighbours has signed the token. In other words, broker b' only trusts the introducers who are directly connecting to her and have positive interactions with her in terms of authorised message exchange. This approach is intuitively reasonable but it may be too restrictive for message dissemination in the MQTT environment.

In the following we introduce a *less restrictive* approach that allows a broker to forward messages to other brokers without the need of signing the AT. We introduce a local threshold $\theta_j \in [0, 1]$ whose value is determined by broker b_j, representing the *minimum* trust value for b_j to accept the AT. Of course, we require that $\theta_j \leqslant \pi$ which means that broker b_j can accept the token but not wish to sign it. Given a directed acyclic weighted graph $G = \langle B, E, T\rangle$, a *path* between b_1 and b_n is a sequence of brokers $b_1, b_2, \ldots, b_n$ such that each consecutive pair $(b_i, b_{i+1}) \in E$ and $1 \leqslant i < n$. Then we define a trust rating of b_1 by b_n along the path $b_1, \ldots, b_n$ to be the average of all the edge trust ratings along the path,

that is $\sum_{i=1}^{n-1} trust(i, i+1)/n-1$, denoted by $trust(b_1, \ldots, b_n)$. Note that there may be many paths between b_1 and b_n, each of which may have different trust ratings. One or more of these paths has the minimum trust rating, which we call *least-trustworthy* path. Given that an $AT = \langle b_1, c, S, F_p, F_s \rangle$ arrives at broker b_n, b_n would accept AT if there exists $b \in S$ such that, among all the paths from b to b_n, the least-trustworthy path $b, \ldots, b_n$ whose rating $trust(b, \ldots, b_n) \geqslant \theta_n$. In other words, when b_n receives the token AT, it determines whether to accept it by checking if there is a broker b who signs the token and the minimum trust rating path from b to b_n is over the threshold θ_n. The rationale for b_n to check on the least-trustworthy path from b is to emphasise *safety*, which means when b_n decides to accept the token, it makes sure that there does not exist a path from b to b_n whose trust rating is less than threshold θ_n. Of course, we can further impose a condition, like PGP [7], requiring at least two or three signers from whom the least-trustworthy path to b_n is over θ_n.

Let us take the example in Fig. 4. Suppose that b_6 receives a token $AT = \langle b_1, c, \{b_1, b_2, b_3\}, F_s, F_p \rangle$ and $\theta_6 = 0.5$. b_6 decides whether to accept the AT by looking at the least-trustworthy paths from b_1, b_2 and b_3. Let us assume that b_6 first evaluates the path b_2, b_4, b_6 whose $trust(b_2, b_4, b_6) = 0.45$, which is less than θ_6. It then evaluates two paths from b_3, that is b_3, b_4, b_6 and b_3, b_5, b_6. We can see that the least-trustworthy path between b_3 and b_6 is b_3, b_4, b_6 and the trust rating for this path $trust(b_3, b_4, b_5) = 0.4$ which is less than θ_6. Finally b_6 looks back at the paths from b_1. There are three paths between b_1 and b_6: b_1, b_2, b_4, b_6, and b_1, b_3, b_4, b_6, and b_1, b_3, b_5, b_6, two of which are the least-trustworthy paths, that is b_1, b_2, b_4, b_6 and b_1, b_3, b_4, b_6. The trust rating of the two paths is the same which is 0.5 equals to θ_6. Thus b_6 would accept the token and distribute the message to its subscribed clients.

Note that the problem of checking the acceptance of token by a broker essential reduces to computing the least-trustworthy paths from signers of the token to the broker. Intuitively, no broker has complete information about the trust rating of all network edges. Instead, each broker begins with only the knowledge of the trust ratings of its own directly connected brokers. Then, through an iterative process of calculation and exchange of information with its neighbouring brokers, a broker gradually computes the least-trustworthy path to a destination or set of destinations (token signers). These are the features of the Distance-Vector (DV) routing algorithm [8] which is based on the celebrated Bellman–Ford equation, namely: $d_x(y) = min_v\{c(x, v) + d_v(y)\}$, where the min_v in the equation is taken over all of x's neighbours. We use the DV algorithm to calculate the least-trustworthy paths in an asynchronous and iterative manner, in order to effectively make decisions for token acceptance.

IV. Concluding Remarks

We are running an experimental evaluation of our trust-based approaches, which is divided into two parts. The first part is to implement our proposed mechanisms on the top of an open source MQTT broker (Eclipse Mosquitto[3]). The second part is to simulate an evaluation environment where clients and brokers exhibiting different *profiles*, such as honest, malicious, and even colluded brokers who manipulate their trust ratings. With populating a reasonable number of nodes for the broker network, the simulation would give a thorough assessment of the effectiveness of our trust-based approach.

In terms of related work, there exists a sizeable body of study on proposing security models for preserving confidentiality for the general publish-subscribe middleware systems [9]. One of the work closed to ours is due to Pesonen *et al.* [10] who use the SPKI authorisation certificates for propagating authorisation across different domains. We believe that the discovery of SPKI/SDSI certificate chain is a special case of our approach, that is finding a path along a set of brokers all of which are required to be signers of a token, but does not require explicit trust calculation and assessment among the brokers. In other words, our approach is more flexible and able to detect compromised clients and brokers by assessing their trustworthiness. Another strand of work related to ours is the study of authorisation for MQTT [2], [3] and its cryptographic enforcement [11], none of which however incorporates the concept of trust for a fully decentralised MQTT environment.

In this paper, we take such an initiative to develop a flexible trust approach that is compliant with MQTT standard and available implementations. In particular, we provide a means of computing a trust rating between two neighbouring brokers, and explore two ways of using these trust ratings to control data sharing in a multi-broker network.

References

[1] "MQTT version 5.0," OASIS, Standard, March 2019.

[2] J. C. F. Carranza and P. W. L. Fong, "Brokering policies and execution monitors for IoT middleware," in *Proceedings of the 24th ACM Symposium on Access Control Models and Technologies*, 2019, pp. 49–60.

[3] P. Colombo, E. Ferrari, and E. D. Tümer, "Regulating data sharing across MQTT environments," *Journal of Network and Computer Applications*, vol. 174, p. 102907, 2021.

[4] A. Jøsang, R. Hayward, and S. Pope, "Trust network analysis with subjective logic," in *Proceedings of the 29th Australasian Computer Science Conference*, 2006, pp. 85–94.

[5] S. Gusmeroli, S. Piccione, and D. Rotondi, "A capability-based security approach to manage access control in the internet of things," *Mathematical and Computer Modelling*, vol. 58, no. 5-6, pp. 1189–1205, 2013.

[6] M. Michaelides, C. Sengul, and P. Patras, "An experimental evaluation of MQTT authentication and authorization in IoT," in *Proceedings of the 15th ACM Workshop on Wireless Network Testbeds, Experimental evaluation & CHaracterization*, 2021, pp. 69–76.

[7] A. Abdul-Rahman, "The PGP trust model," *EDI-Forum: the Journal of Electronic Commerce*, vol. 10, no. 3, pp. 27–31, 1997.

[8] C. Hedrick, "Routing information protocol," Network Working Group, RFC 1058, June 1988.

[9] E. Onica, P. Felber, H. Mercier, and E. Rivière, "Confidentiality-preserving publish/subscribe: A survey," *ACM Computing Surveys*, vol. 49, no. 2, pp. 27:1–27:43, 2016.

[10] L. I. W. Pesonen, D. M. Eyers, and J. Bacon, "Access control in decentralised publish/subscribe systems," *Journal of Networks*, vol. 2, no. 2, pp. 57–67, 2007.

[11] K. Spielvogel, H. C. Pöhls, and J. Posegga, "TLS beyond the broker: Enforcing fine-grained security and trust in publish/subscribe environments for IoT," in *Proceedings of the 17th International Workshop on Security and Trust Management*, vol. 13075, 2021, pp. 145–162.

[3]Eclipse Mosquitto: https://mosquitto.org/