An Efficient Data Protection Scheme Based on Hierarchical ID-Based Encryption for Message Queueing Telemetry Transport

Chun-I Fan, Cheng-Han Shie, Yi-Fan Tseng* and Hui-Chun Huang

***Abstract*—As Internet of Things (IoT) thriving over the whole world, more and more IoT devices and IoT-based protocols have been designed and proposed in order to meet people's needs. Among those protocols, message queueing telemetry transport (MQTT) is one of the most emerging and promising protocol, which provides many-to-many message transmission based on the "publish/subscribe" mechanism. It has been widely used in industries such as the energy industry, chemical engineering, self-driving, etc. While transporting important messages, MQTT specification recommends the use of TLS protocol. However, computation cost of TLS is too heavy. Since topics in a broker are stored with a hierarchical structure, In this manuscript, we propose a novel data protection protocol for MQTT from hierarchical ID-based encryption. Our protocol adopts the intrinsic hierarchical structures of MQTT, and achieves constant-size keys, i.e. independent of the depth in hierarchical structures.**

***Index Terms*—Hierarchical ID-Based Encryption, Message Queueing Telemetry Transport, MQTT, Data Protection.**

I. Introduction

INTERNET of Things (IoT) has been used worldwide in the past decade. According to the report from Statista [1], the number of connected devices, not only for general customers but industries, will increase to around 25 billion in 2025, as shown in Figure 1. The term IoT generally refers to scenarios that are made up of things or devices connected by the Internet. They can interact with each other, absorb and share information. For IoT devices to communicate, a data protocol is required. As of now, there are several data protocols when it comes to connecting various devices in an IoT environment, such as CoAP [2] (Constrained Application Protocol), XMPP [3] (Extensible Messaging and Presence Protocol), MQTT [4] (Message Queueing Telemetry Transport) and so on. Among those protocols, MQTT is the first to be proposed and the most complete one. Besides, it is the only protocol that supports many-to-many transmissions (Figure 2). The introduction for MQTT is presented in Section II-A.

Chun-I Fan is with the Department of Computer Science and Engineering and the Information Security Research Center, National Sun Yat-sen University, Kaohsiung 804, Taiwan, and also with the Intelligent Electronic Commerce Research Center, National Sun Yat-sen University, Kaohsiung 804, Taiwan.

Cheng-Han Shie and Hui-Chun Huang are with the Department of Computer Science and Engineering, National Sun Yat-sen University, Kaohsiung 804, Taiwan

Yi-Fan Tseng is with the Department of Computer Science, National Chengchi University, Taipei 11605, Taiwan.

*Corresponding Author: yftseng@cs.nccu.edu.tw

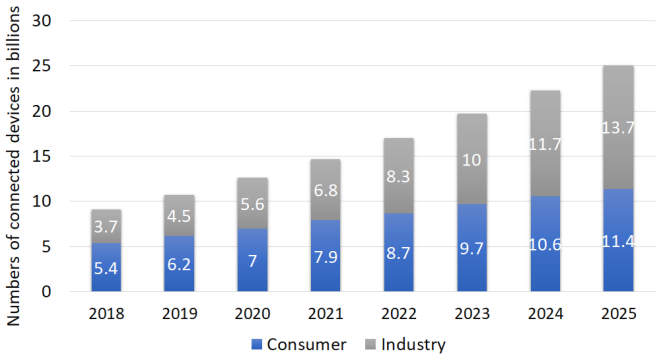

Fig. 1: Forecast numbers of Internet of Things (IoT) connected devices worldwide from 2018 to 2025 (in billions)

The concept of MQTT was first proposed by Andy Stanford-Clark and Arlen Nipper from IBM and Eurotech in 1999. It was a project to monitor the oil pipeline across deserts. The purpose was to provide data transmissions on a lightweight and little battery power consumption protocol because the connection between the devices was through an extremely expensive satellite link. In 2013, IBM submitted MQTT version 3.1 [5] to be the OASIS (Organization for the Advancement of Structured Information Standards) specification. Historically, instead of message queuing, the "MQ" in "MQTT" originally is the name of the IBM MQ product line. It applies a publish/subscribe mechanism to minimize the payload and overhead. Now, MQTT is widely used in IT departments and available in many open sources or programming languages. It is used not only to monitor oil pipelines in the energy industry mentioned above but also to monitor or send commands in chemical, medical, autonomous driving, and other industries. Even Facebook Messenger applies MQTT which is a common communication software. Therefore, the security of the MQTT protocol is important.

In the default situation, the transmissions with MQTT on port 1883 are not encrypted. For the sensitive information contained in the message, the MQTT specification recommends using the TLS protocol on port 8883 for protecting the data. Although most brokers and MQTT platforms support the TLS protocol, Mathews *et al.* [6] and Sadio *et al.* [7] mentioned that CPU usage and communication overhead come at the expense of limited devices. Once a message is published, TLS protocol needs to perform a handshake process. Although an

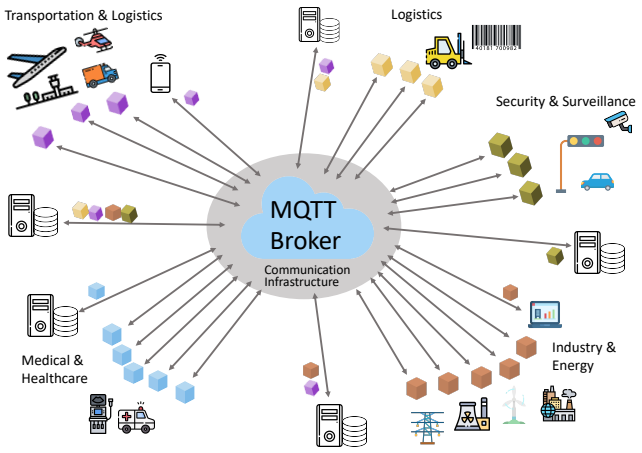

Fig. 2: MQTT used in the IoT M2M industry

IoT payload is small, it has to be transmitted frequently. If TLS protocol is often used, it always needs to reconnect and perform a handshake process due to the unstable signal of the IoT connection. That consumes a lot of power and computation time. Besides, the TLS session keeps connecting until the MQTT client finishes its work. In this case, it is not beneficial for short-lived connection. According to TLS 1.2 [8], a handshake protocol spends approximately 250 microseconds in the six steps (Figure 3). Even if with the improvements, TLS version 1.3 [9] still takes at least 150 microseconds. Therefore, it takes up most of the computation and time for those devices.

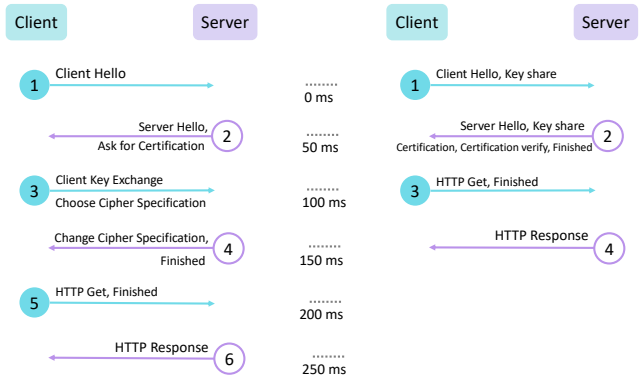

Fig. 3: The time cost of TLS version 1.2 and version 1.3

Concerning the IoT devices that are unsuitable for TLS protocol, payload encryption is a more appropriate method to protect messages. TLS protocol encrypts the payload of the TCP packet which is the entire MQTT packet including the header. In contrast with encrypting the whole MQTT packet, payload encryption only encrypts the content of the MQTT messages, the broker can directly check the MQTT header (Figure 4) without decryption. In 2015, Singh *et al.* [10] proposed an attribute-based encryption (ABE) scheme to encrypt the messages of MQTT. Except for the original hierarchical topic tree, Singh *et al.* needs to build another access tree for the attributes or identities of Subscriber. The access trees may not be shared, so Publisher or Broker need more space to store the access tree. Some of the IoT devices are resource-constrained, they may not have enough space to store or create the access trees. Singh *et al.*'s scheme needs to create attributes for Subscriber and convert them into an access structure, which requires a lot of time to compute and space to store. Moreover, each access tree may represent only a topic. The more topics exist, the more access structures Broker needs to store.

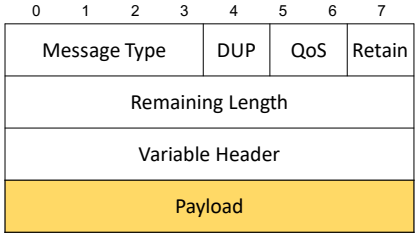

Fig. 4: The packet of MQTT

A. Related Work

In 2015, Singh *et al.* [10] propose a payload encryption scheme using KP-ABE [11] and CP-ABE [12], called SMQTT. They also demonstrate the feasibility of their protocol for various IoT environments through simulations to evaluating the performance. In their scheme, using CP-ABE scheme has higher complexity (storage and computation) than the KP-ABE one. In their experiment, they compare not only the scheme with the ABE scheme from X. Wang *et al.* [13], but also the performance between CP-ABE and KP-ABE. Message M is encrypted by 128 bit AES key that is encrypted by KP-ABE or CP-ABE. However, Singh *et al.*'s scheme does not present the details about how the parameters set and the security proof work.

B. Contributions

To solve the problems mentioned above, we propose payload encryption from hierarchical identity-based encryption (HIBE) for MQTT, called MQHIBE. Differing from Singh *et al.*'s scheme, hierarchical topics of MQTT perfectly match the HIBE. Compared with the MQTT messages protected by the TLS protocol and Singh *et al.*'s scheme, the proposed MQHIBE scheme is more efficient and achieves the property of hierarchical topics. Furthermore, as we will show in Section IV, the proposed scheme has better performance in time complexity and it can defend replay attacks. If an attacker resends the message, nothing will happen to the system, the subscriber just receives the same message again.

II. Preliminaries

This section provides the background knowledge, and the definition of hierarchical ID-based encryption (HIBE) followed by certain mathematical assumptions..

A. MQTT

MQTT is a publish/subscribe protocol that runs on top of TCP [14] network. A message packet in MQTT is presented as Figure 5. In an MQTT protocol, there are three main characters: Publisher, Broker, Subscriber, and the message transports via topics. Any Publisher or Subscriber that connects to the centralized Broker over networks is considered to be a client. In MQTT, messages are organized in a hierarchy of topics as shown in Figure 6. We briefly introduce the terminologies below.

- **Subscriber:** A client that subscribes to a topic or topics from Broker.
- **Publisher:** A client that publishes a message to Broker with the corresponding topic.
- **Broker:** Broker is a server that receives the messages sent from Publisher and forwards them to Subscriber. The clients must actively connect to Broker, then Broker will hold the connection to persistent clients. There are several platforms of MQTT Brokers include HiveMQ, AWS IoT.
- **Topic:** A topic of an MQTT Broker is a connection between Subscriber and Publisher. Each message belongs to a certain topic. A message topic is composed of different topic levels separated by a slash and represented as a string like Home/Yard/Pond/Water Level. Topics are different due to uppercase and lowercase, and the permutation of the topic is also important. If there is a topic Home/Yard/Pond, the message with topic Home/Pond/Yard will not be accepted by Broker because of the exchange between Pond and Yard.

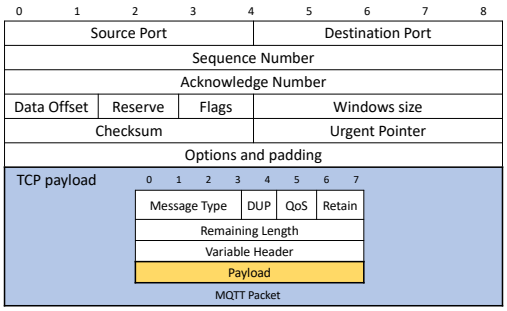

Fig. 5: An MQTT packet on a TCP network

Connecting with Broker, Subscriber sends a Subscribe packet to Broker to create one or more subscriptions. When Publisher sends a message to Broker, Broker will forward the messages to Subscriber that match those subscriptions. Publisher need not to know where Subscriber is, and Subscriber need not to know who sends the message. If Broker receives a message with a topic for which there are no current Subscriber, it will discard the topic unless Publisher indicates that the topic is to be retained.

1) Wildcard Characters: Subscriber can not only subscribe to an exact topic but also use a wildcard character to subscribe to multiple topics concurrently. A wildcard character can only

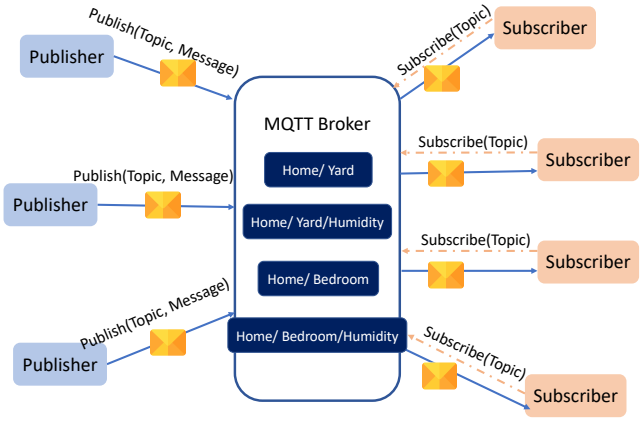

Fig. 6: The message transport in MQTT

be used for the topic subscription. There are two kinds of wildcards in MQTT: single-level wildcard and multi-level wildcard, which are denoted by the symbols "+" and "#", respectively.

- **Single-Level Wildcard** (+): A single-level wildcard can replace a topic level. The symbol "+" represents a single-level wildcard in the MQTT topic, and it can be put at any level of the topic. For example, as shown in Figure 7, a subscription to Home/Yard/+ can produce the following results:
  - Home/Yard/Pond
  - Home/Yard/PIR 1
  - Home/Yard/PIR 2
  - Home/Yard/Temperature
  - Home/ Yard/ Humidity

  Note that the symbol is allowed to be placed in the middle of the topic such as Home/+/Humidity, which indicates "Home/Yard/Humidity" and "Home/Bedroom/Humidity" in Figure 7.
- **Multi-Level Wildcard** (#): The multi-level wildcard can replace many topic levels at a time, and it must be placed as the last character. For a subscription to the topics with a multi-level wildcard, Subscriber will receive all the messages that own the same prefix to the topic. If a topic contains only a multi-level wildcard, it means a subscription to all the topics. For example, as shown in Figure 7, a topic Home/Bedroom/#, it means subscriptions below:
  - Home/Bedroom
  - Home/Bedroom/Temperature
  - Home/Bedroom/Humidity

B. Hierarchical ID-Based Encryption (HIBE)

Jeremy Horwitz and Ben Lynn proposed the first HIBE scheme [15] in 2002 which is a two level structure scheme. HIBE is an extension form of IBE [16] which is a public-key cryptography.

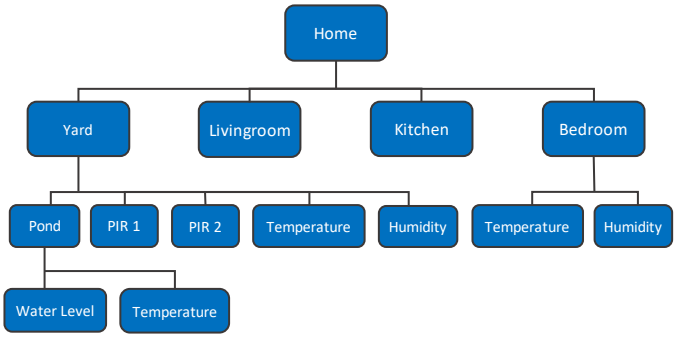

Fig. 7: The levels of hierarchical topics

Definition 1. An HIBE scheme consists of the following probabilistic algorithms:

- **Setup** $(SK_0, \mathcal{P})$ **:** PKG runs the function and a security parameter to generate a master key MK_0 (which we also call the level-0 key) and a set $\mathcal{P}$ of system parameters.
- **KeyGen** $(SK_{i-1}, ID_i, \mathcal{P}) \rightarrow SK_i$ **:** The algorithm takes system parameters $\mathcal{P}$, master secret SK_{i-1}, and identity ID_i as input. It outputs a private key SK_i corresponding to identity ID_i where i denotes the i^{th} level of the ID.
- **Encrypt** $(\mathcal{P}, ID_i, M) \rightarrow CT$ **:** A data owner runs the algorithm to generate a ciphertext CT. It takes the set of system parameters $\mathcal{P}$, a message M, and an identity ID_i as input. Then, the data owner can generates a ciphertext CT.
- **Decrypt** $(\mathcal{P}, ID_i, CT, SK_i) \rightarrow M$ **:** A receiver performs the algorithm to obtain the message. It takes ciphertext CT, system parameters $\mathcal{P}$, identity ID_i and private key SK_i as input. Eventually, the receiver can get message M.

C. Bilinear Mapping

Let G and G_1 be two cyclic multiplicative groups of prime order p. A bilinear mapping $e : G \times G \rightarrow G_1$ satisfies the following properties [16] in which g is a generator of G.

- **Bilinearity:** $e(g^a, g^b) = e(g, g)^{ab}$, $\forall a, b \in \mathbb{Z}_p$.
- **Non-Degeneracy:** The function does not map all pairs in $G \times G$ to the identity of G_1. Since G and G_1 are groups of the same prime order, it implies that if g is a generator of G, then $e(g, g)$ is a generator of G_1.
- **Computability:** There exists an efficient algorithm to compute $e(g, g)$, $\forall g \in G$.

D. Weak Bilinear Diffie-Hellman Inversion (wBDHI) Assumption

Let G and G_1 be two cyclic groups of prime order p, g be a generator of G, and $e : G \times G \rightarrow G_1$ be a bilinear mapping.

Definition 2. Given $< G, G_1, e, g, h, g^{\alpha}, g^{\alpha^2}, \cdots, g^{\alpha^{\ell}}, Z >$ for some random $\alpha \in \mathbb{Z}_p^*$ and $g, h \in G$, decide if Z is equal to $e(g, h)^{\alpha^{\ell+1}}$.

Definition 3. An algorithm $\mathcal{A}$ with an output $b' \in \{0, 1\}$ is said to have the advantage ϵ in solving the ℓ-wBDHI problem if

$$|\Pr[\mathcal{A}(g, h, \vec{y}, e(g, h)^{\alpha^{\ell+1}})] = 1 - \Pr[\mathcal{A}(g, h, \vec{y}, Z)] = 1| \geq \epsilon$$

where $\vec{y} = (g^{(\alpha^i)})_{i=1,\ldots,\ell} \in G^{\ell}, \alpha \in_R \mathbb{Z}_p$, $g, h \in G$ and $Z \in_R G_1$. We say that the ℓ-wBDHI assumption [17], [18] holds if no polynomial-time algorithm has non-negligible advantage in solving the ℓ-wBDHI problem.

III. The Proposed MQHIBE Scheme

In this section, we demonstrate a secure scheme based on hierarchical ID-based encryption (HIBE) [17] for the MQTT protocol used in IoT environments, called MQHIBE. Our scheme only encrypts MQTT messages to publish, and thus it will not modify the MQTT structure or cause other problems. Our are four algorithms in our scheme: *Setup, Subscription, Publication, Reception*. The system model of the proposed scheme is illustrated in Figure 8, where Broker is considered honest-but-curious. The notations used in the proposed scheme are shown in TABLE I.

In the proposed scheme, Publisher and Subscriber are the clients who have been authenticated by Broker. Publisher can be imagined as a sensor, e.g., a thermometer, and it periodically sends out the message about temperatures. Subscriber can be imagined as a smartphone or a device to record temperatures. When Publisher sends a message, and Broker will forward it to Subscriber. In the proposed scheme, the public encryption key for a topic $T_1/T_2/\cdots/T_q$ is viewed as a vector in $(\mathbb{Z}_p^*)^q$. This can be done by regarding T_i as the corresponding integer of its binary representation.

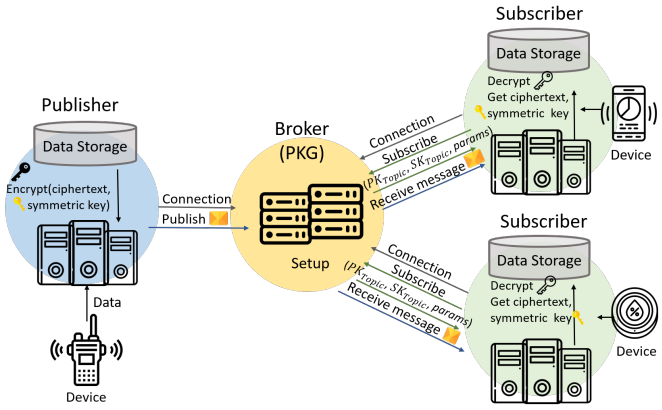

Fig. 8: The sysyem model of the proposed MQHIBE scheme

A. Setup Algorithm

Broker plays the role of the public key generator (PKG) to generate the public parameters the master secret key as follows. Let ℓ be the maximum depth of HIBE.

1) Construct the parameters for bilinear map $e : G \times G \rightarrow G_1$, where G is a bilinear group of prime order p.

TABLE I: The Notations

Notation	Meaning
G	a cyclic multiplicative group of prime order p
G_1	a cyclic multiplicative group of prime order p
e	a bilinear mapping; $e : G \times G \rightarrow G_1$
PK_{Topic}	a public key, $PK_{Topic} = (T_1/T_2/\cdots/T_q) \in (\mathbb{Z}_p^*)^q$
SK_{Topic}	a secret key to the public key PK_{Topic}.
$PK_{Topic\|q-1}$	a public key at the $(q-1)$-th level, $PK_{Topic\|q-1} = (T_1/T_2/\cdots/T_{q-1}) \in (\mathbb{Z}_p^*)^{q-1}$
$SK_{Topic\|q-1}$	a secret key to the public key $PK_{Topic\|q-1}$.
SE	a symmetric key encryption algorithm
SD	a symmetric key decryption algorithm
$\mathcal{K}$	the symmetric key space
H	a collision resistant hash function $H : \{0,1\}^* \rightarrow \mathbb{Z}_p^*$.
M	a message
CT	a ciphertext
ℓ	the maximum depth of the HIBE
q	the q-th level of the HIBE

2) Choose a generator $g \in G$, a number $\alpha \in \mathbb{Z}_p$ at random and set $g_1 = g^\alpha$.
3) Select a collision resistant hash function $H : \{0,1\}^* \rightarrow \mathbb{Z}_p^*$.
4) Choose random elements $\{c_1, c_2, h_1, ..., h_\ell\} \in G$.
5) Set public parameters $params = (g, g_1, c_1, c_2, h_1, ..., h_\ell)$ and a master secret key c_1^α.
6) Choose and publish a secure symmetric encryption/decryption algorithms $(SE, SD, \mathcal{K})$ where the key space $\mathcal{K} = G_1$.

B. Publication Algorithm

Publisher encrypts message M using symmetric encryption k. Then, Publisher encrypts symmetric key k with the corresponding PK_{Topic} and a random number s as follows. Finishing the encryption, Publisher sends ciphertext CT to Broker.

1) Generate a symmetric key $k \in \mathcal{K}$.
2) Let the public key $PK_{Topic} = (T_1/T_2/\cdots/T_q) \in (\mathbb{Z}_p^*)^q$.
3) Choose random $s \in \mathbb{Z}_p$.
4) Compute $CT = (e(g_1, c_1)^s \cdot k, g^s, (h_1^{T_1} \cdots h_q^{T_q} \cdot c_2)^s, SE_k(M))$.
5) Send CT to Broker.

C. Subscription Algorithm

When Subscriber sends a Subscribe packet to Broker, Broker generates the corresponding secret key to Subscriber. The algorithm takes public key PK_{Topic}, master secret key c_1^α, and public parameters $params$ as input. The details are shown as follows.

1) Choose a random $r \in \mathbb{Z}_p$.
2) Compute secret key SK_{Topic} under PK_{Topic}.
$PK_{Topic} = (T_1/T_2/\cdots/T_q) \in (\mathbb{Z}_p^*)^q$
$SK_{Topic} = (c_1^\alpha \cdot (h_1^{T_1} \cdots h_q^{T_q} \cdot c_2)^r, g^r, h_{q+1}^r, \cdots, h_\ell^r)$.
3) Send (PK_{Topic}, SK_{Topic}) to Subscriber.

Subscription Algorithm with Multi-Level Wildcard Character #: We next discuss the case when Subscriber submits a topic with multi-level wildcard character. The most significant feature of HIBE is that the secret key of the children can be generated from the parent node's secret key. When Subscriber sends a Subscribe packet with # to Broker, Broker generates the corresponding secret key and the parameters, and then send the public keys, the secret key, and the parameters back to Subscriber.

For example, assume that there are the subscription of the two topics are at the $(q-1)$-th and q-th levels separately, and the two topics are parent-child relationships. The secret key at the q-th level can be generated from the parent topic at the $(q-1)$-th level. The public key and secret key of the parent topic at $(q-1)$-th level are represented as $PK_{Topic|q-1}, SK_{Topic|q-1}$ below. Let $PK_{Topic|q-1} = (T_1/T_2/\cdots/T_{q-1})$. Broker takes a random r', public key $PK_{Topic|q-1}$, master secret key c_1^α, and public parameters $params$ as input to generate secret key $SK_{Topic|q-1}$. Then, Broker sends t, public key $PK_{Topic|q}$, parent secret key $SK_{Topic|q-1}$, and public parameters $params$ to Subscriber. The details are shown as follows.

1) Choose a random number $r \in \mathbb{Z}_p$ for SK_{Topic}.
2) Compute the secret key of the parent node at the $(q-1)$-th level,
$SK_{Topic|q-1} = (c_1^\alpha \cdot (h_1^{T_1} \cdots h_{q-1}^{T_{q-1}} \cdot c_2)^{r'}, g^{r'}, h_q^{r'}, \cdots, h_\ell^{r'}) = (a_0, a_1, b_q, \cdots, b_\ell)$.
3) Choose a random number $t \in Z_p$, and set $r = r' + t$.
4) Send $t, SK_{Topic|q-1}$ and PK_{Topic} to Subscriber.

Receiving the message from Broker, Subscriber gets t, $params$, and the key pairs $(PK_{Topic|q-1}, SK_{Topic|q-1})$ of the parent topic at level $(q-1)$-th. and it compute the secret key as $SK_{Topic} = (a_0 \cdot b_q^{T_q} \cdot (h_1^{T_1} \cdots h_q^{T_q} \cdot c_2)^t, a_1 \cdot g^t, b_{q+1} \cdot h_{q+1}^t, ..., b_\ell \cdot h_\ell^t)$.

D. Reception Algorithm

Upon receiving ciphertext CT from Publisher, Broker forwards it to whom subscribes to the same topic. Subscriber uses secret key SK_{Topic} to decrypt ciphertext CT and gets symmetric key k and encrypted message $SE_k(M)$. Utilizing symmetric key k, Subscriber can decrypt $SE_k(M)$ and get message M. The details are shown as follows.

1) Let $CT = (e(g_1, c_1)^s \cdot k, g^s, (h_1^{T_1} \cdots h_q^{T_q} \cdot c_2)^s, SE_k(M)) = (A, B, C, D)$.

2) Let $SK_{Topic} = (c_1^{\alpha} \cdot (h_1^{T_1} \cdots h_q^{T_q} \cdot c_2)^r, g^r, h_{q+1}^r, \cdots, h_\ell^r) = (a_0, a_1, b_{q+1}, \cdots, b_\ell)$.

3) Compute

$$\frac{e(a_1, C)}{e(B, a_0)} = \frac{1}{e(g_1, c_1)^s}$$

and retrieve the symmetric key

$$k = A \cdot \frac{e(a_1, C)}{e(B, a_0)}.$$

4) To retrieve the message, compute

$$M = SD_k(D).$$

Correctness of decryption of cyphertext CT is demonstrated as follows.

$$\begin{aligned}
&\frac{e(a_1, C)}{e(B, a_0)} \\
&= \frac{e(g^r, (h_1^{T_1} \cdots h_q^{T_q} \cdot c_2)^s)}{e(g^s, c_1^{\alpha}(h_1^{T_1} \cdots h_q^{T_q} \cdot c_2)^r)} \\
&= \frac{e(g^r, (h_1^{T_1} \cdots h_q^{T_q} \cdot c_2)^s)}{e(g^s, c_1^{\alpha}) \cdot e(g^s, (h_1^{T_1} \cdots h_q^{T_q} \cdot c_2)^r)} \\
&= \frac{e(g^r, (h_1^{T_1} \cdots h_q^{T_q} \cdot c_2)^s)}{e(g^s, c_1^{\alpha}) \cdot e(g^r, (h_1^{T_1} \cdots h_q^{T_q} \cdot c_2)^s)} \\
&= \frac{1}{e(g, c_1)^{s\alpha}} \\
&= \frac{1}{e(g^{\alpha}, c_1)^s} \\
&= \frac{1}{e(g_1, c_1)^s}.
\end{aligned}$$

Compute symmetric key k

$$k = A \cdot \frac{e(a_1, C)}{e(B, a_0)} = e(g_1, c_1)^s \cdot k \cdot \frac{1}{e(g_1, c_1)^s}$$

and get message M

$$M = SD_k(D).$$

IV. Comparison

In this section, we compare the proposed scheme with [10] and [4] in terms of properties and performances. We summarize the functionality comparison between [10] in TABLE II. The comparison with [10] and MQTT standard using TLS protocol are presented in TABLE V and TABLE VI. Some computation costs for cryptographic primitives are shown in TABLE IV.

A. Properties Comparison

Our MQHIBE scheme adopts the properties of hierarchical encryption, and its security is also guaranteed in Section **??**. In the following, we present differences from scheme of [10], shown in TABLE II.

- **Encryption for Hierarchical Structure**: As an IoT data protocol, MQTT publishes/subscribes the messages that rely on hierarchical topics. Once a client publishes a message on a specific topic, Broker will forward the message that matches the topic subscription. An MQTT topic is a UTF-8 string that consists of one or more topic levels. Every topic level is separated by a slash character that makes the topic presence hierarchically in the string. Compared with Singh *et al.*'s scheme, ours is more tailored for MQTT.
- **Security**: In chosen-plaintext attacks (CPA), the adversary can choose several plaintexts to be encrypted and have access to the generated ciphertexts. In chosen-ciphertext attacks (CCA), the adversary can additionally gather information by a decryption oracle with chosen ciphertexts. The CCA security is more strong than the CPA security since an CCA adversary is allowed to access more resources. Besides, STD and ROM denote *the standard model* and *the random oracle model* respectively. In the standard model, the adversary is only restricted to reasonable runtime and computation ability. In the random oracle model, there is an additional restriction that the adversary is asked to access hash oracles to obtain hash values, rather than compute the values by itself. Due to the additional restriction, the standard model is more preferable for a security proof.

B. Performance Evaluation and Discussion

We analyze the performance of the encryption and decryption algorithms via python libraries [19], [20] on a Ubuntu 18.04.4 LTS Linux system with Intel Core i9-9940X 3.30GHz. Standard MQTT with TLS protocol encrypts the payload of TCP packets, which is an entire MQTT packet. The cryptographic primitives of TLS protocol apply to version 1.2 and the latest version 1.3. To compare with the protocols of [8], [10] and our MQHIBE, we implement RSA algorithm, SHA-384, AES algorithm, and other primitives via python libraries. The information for the environment is shown in TABLE III, and the time consumption for each primitive is shown in TABLE IV. We note that AES-GCM is a kind of symmetric cryptosystem extended from AES. We use AES-GCM as the symmetric encryption/decryption algorithms used in each protocols. In the following, we analyze the performance under the scenario that the message is encrypted under the topic Home/Bedroom/Temperature, and a maximum of 3 levels in MQHIBE. The structure for the topics is shown in Fig. 9.

C. Comparison with Singh et al.'s ABE-Based Methods

- Singh *et al.* [10] based on CP-ABE [12]:
  According to Figure 9, we construct an access tree of Subscriber's identities for CP-ABE illustrated as

TABLE II: Properties Comparison

Scheme	Hierarchical	Encryption	Assumption	Security proof
Singh *et al.* [10]	No	CP-ABE [12]	None	ROM/CPA
		KP-ABE [11]	DBDH	STD/CPA
The MQHIBE scheme	Yes	HIBE	–	–

TABLE III: Simulation Environment

Operating System	Linux Ubuntu 16.04 LTS 32bit
CPU	Intel(R)Core(TM) i7-4650U CPU @ 1.70GHz
Memory	7.8 GB
Motherboard	Apple Inc. 121.0.0.0.0

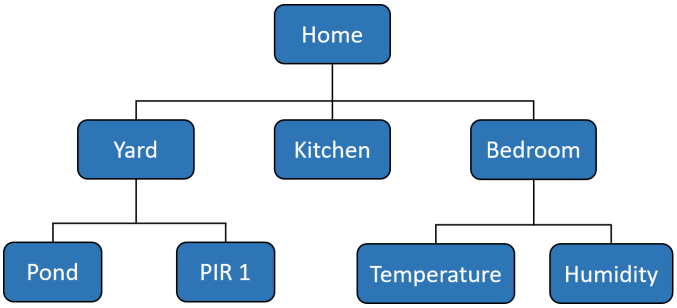

Fig. 9: The Structure of Hierarchical Topics

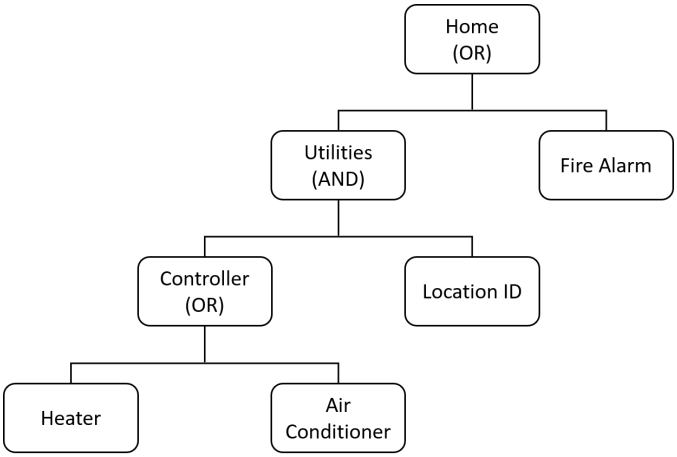

Fig. 10: The access structure of Subscriber's identities in CP-ABE and KP-ABE

Figure 10. There are four attributes only for topic Home/Bedroom/Temperature. If Publisher needs to send another message that is no relation to "temperature" like Home/Bedroom/Humidity, the message cannot use the same access structure shown in Figure 10. Therefore, Broker has to store additional attributes for Subscriber's identities and provides Publisher to generate the access tree.

- **Key Generation**: The cost of key generation for topic Home/Bedroom/Temperature is $T_s + T_m + 4 \cdot (T_s + T_{h_{384}} + T_s + T_s) \approx 13T_s + 4T_{h_{384}} + T_m \approx 0.247 + 0.004 + 0.001 \approx 0.252$ ms.
- **Encryption**: Publisher first generates a symmetric key to encrypt the plaintext, that is, an AES-GCM key. Then, the ABE scheme is used to protect the symmetric key. The cost of generating ciphertext is $T_p + T_a + T_m + T_s + 4 \cdot (T_s + T_{h_{384}} + T_s) + T_{AES-GCM_{Enc}} \approx 9T_s + 4T_{h_{384}} + T_p + T_a + T_m + T_{AES-GCM_{Enc}} \approx 0.171 + 0.004 + 33.524 + 0.025 + 0.001 + 0.003 \approx 33.728$ ms.
- **Decryption**: Using the decryption algorithm of the ABE scheme, we can get the symmetric key, and then recover the palintext. The cost of decrypting the ciphertext is $2 \cdot T_p + T_a + T_m + 7T_a + T_p + 2T_a + T_{AES-GCM_{Dec}} \approx 10T_a + 3T_p + T_m + T_{AES-GCM_{Dec}} \approx 0.25 + 100.572 + 0.001 + 0.231 \approx 101.054$ ms.

- Singh *et al.* [10] based on KP-ABE [11]:
  According to Figure 9, we construct an access tree of Subscriber's identities for KP-ABE illustrated as

Figure 10. There are four attributes only for topic Home/Bedroom/Temperature. If Publisher needs to send another message that is no relation to "temperature" like Home/Bedroom/Humidity, the message cannot use the same access structure shown in Figure 10. Therefore, Broker has to store additional attributes for Subscriber's identities and generates the access tree for Publisher in advance.

- **Key Generation**: The cost of key generation for topic Home/Yard/Pond is $4 \times (T_m + T_s) \approx 4 \times 0.02 \approx 0.08$ ms.
- **Encryption**: The analysis is similar to the CP-ABE case. The cost of generating ciphertext is $T_a + T_s + 4 \cdot T_s + T_{AES-GCM_{Enc}} \approx T_a + 5T_s + T_{AES-GCM_{Enc}} \approx 0.025 + 0.095 + 0.003 \approx 0.123$ ms.
- **Decryption**: The analysis is similar to the CP-ABE case. The cost of decrypting ciphertext is $2 \cdot T_p + T_a + T_m + 4T_a + T_a + T_{AES-GCM_{Dec}} \approx 2T_p + 6T_a + T_m + T_{AES-GCM_{Dec}} \approx 67.048 + 0.15 + 0.001 + 0.231 \approx 67.43$ ms.

- The MQHIBE scheme:
  - **Key Generation**: According to the assumptions, the cost of key generation for three-level topic Home/Yard/Pond is $T_s + T_a + 3 \cdot T_s + 3 \cdot T_a + T_s + T_s \approx 6T_s + 4T_a \approx 0.114 + 0.1 \approx 0.214$ ms.
  - **Encryption**: Publisher first generates an AES-GCM key to encrypt the plaintext, then, uses the MQHIBE encryption algorithm to protect the AES-GCM key. The cost of generating ciphertext is $T_a + 3T_s + T_{AES-GCM_{Enc}} \approx 0.025 + 0.057 + 0.003$ ms ≈ 0.085 ms.
  - **Decryption**: By using the decryption algorithm of

TABLE IV: Computation Costs of Cryptographic Primitives in millisecond (ms)

Notation	Meaning	Key size	Cost
$T_{AES-GCM_{Enc}}$	the cost of an AES-GCM encryption	256 bits	0.003 ms
$T_{AES-GCM_{Dec}}$	the cost of an AES-GCM decryption	256 bits	0.231 ms
T_{ECDHE}	the cost of an ECDHE operation	-	62.972 ms
$T_{RSA_{Enc}}$	the cost of an RSA encryption	2048 bits	2.903 ms
$T_{RSA_{Dec}}$	the cost of an RSA decryption	2048 bits	109.462 ms
$T_{h_{384}}$	the cost of a 384 bits hash operation	-	0.001 ms
T_p	the cost of a pairing operation	-	33.524 ms
T_m	the cost of a modular multiplication in $\mathbb{Z}_q$	-	0.001 ms
T_s	the cost of a scalar multiplication in an additive group or an exponentiation in a multiplicative group	-	0.019 ms
T_a	the cost of an addition in an additive group or a multiplication in a multiplicative group	-	0.025 ms

the MQHIBE scheme, we can get the symmetric key and recover the plaintext. The cost is $2T_p + 2T_a + T_{AES-GCM_{Dec}} \approx 67.048{+}0.05{+}0.231\text{ms} \approx 67.329$ ms.

D. Comparison with TLS

The following comparison is between the standard MQTT with TLS and the proposed MQHIBE scheme via three aspects: Preparation, Encryption cost, Decryption cost. The results are shown in TABLE VI. We assume that the published message of topic is Home/Yard/Pond for the convenience in comparison.

- Standard MQTT with TLS protocol:
  - **Preparation**: In preparation, TLS protocol needs to do handshake protocol and key exchange. The cost of the preparation is $T_{ECDHE}{+}T_{RSA_{Enc}}{+}T_{RSA_{Dec}}{+}17T_{h_{384}} \approx 62.972 + 2.903 + 109.462 + 0.017 \approx 175.354$ ms
  - **Encryption**: After preparation, Publisher sends the plaintext to Broker. TLS protocol previously encrypts the plaintext before transmission by using symmetric cryptosystem such as AES-GCM. The cost of generating a ciphertext is $T_{AES-GCM_{Enc}} + T_{h_{384}} \approx 0.003 + 0.001 \approx 0.004$ ms.
  - **Decryption**: After receiving the message, Broker decrypts the ciphertext with the symmetric key. If there is a subscription related to the message, Broker needs to create another TLS secure channel to encrypt the plaintext again. The more subscriptions to the topic of the message the more TLS secure channels need to be created. The cost of decrypting ciphertext is $T_{AES-GCM_{Dec}} + T_{h_{384}} \approx 0.231 + 0.001 \approx 0.232$ ms.
- The MQHIBE scheme:
  - **Preparation**: In preparation, the MQHIBE scheme needs to perform key generation after *setup*. The cost of key generation for three-level topic Home/Yard/Pond is $T_s{+}T_a{+}3{\cdot}T_s{+}3{\cdot}T_a{+}T_s{+}T_s \approx 0.114 + 0.1 \approx 0.214$ ms.
  - **Encryption**: The cost of generating ciphertext is $T_a + 3T_s + T_{AES-GCM_{Enc}} \approx 0.025 + 0.057 + 0.003$ ms ≈ 0.085 ms.
  - **Decryption**: The cost of decrypting ciphertext is $2T_p + 2T_a + T_{AES-GCM_{Dec}} \approx 67.048 + 0.05 + 0.231\text{ms} \approx 67.329$ ms.

All the clients in MQTT are IoT sensors or devices for specific jobs, hence the subscription will not often be canceled or changed. Only when a subscription to new topics occurs, or Broker updates all the keys, Subscriber needs to get a new HIBE or ABE keys from Broker. Unlike the public-key cryptography, MQTT using TLS protocol always needs to perform key exchange before sending the message.

V. Conclusion

In consideration of message confidentiality in MQTT, researches presented different encryption mechanisms in the literature. It is a worth-focusing issue because of the rising number of connected IoT devices, and the MQTT protocol has been widely used in recent years. The MQTT specification suggests that either the TLS protocol or other cryptographic schemes is a good option for protecting sensitive messages. Many companies and MQTT platforms choose the TLS protocol to encrypt messages due to the generality and convenience. Yet, the TLS protocol has high computation cost and time consumption. Some researches turned to study other encryption methods, e.g. ABE, and implement them in the MQTT environment, but without detailed and complete security proofs for the schemes.

To cope with the problem, a novel MQTT encryption scheme, i.e. MQHIBE, is designed using hierarchical ID-based encryption during the communications. In an MQTT protocol, every message belongs to a topic which is a hierarchical namespace stored in the broker. This is the reason why the proposed scheme utilized hierarchical ID-based encryption to protect the messages. Different from the scheme with the ABE, it needs to give values to attributes that represent Subscriber. Moreover, every attribute requires a specific value, but the proposed scheme does not need to do so. Furthermore, the proposed scheme meets the need of the subscription by a multi-level wildcard character. The most significant feature of MQHIBE is that the root node can hierarchically generate the private keys of the descendants, and the private key of a node can be generated from the private key of its parent node. As a result, we take the advantage and use it in a multi-level wildcard character when subscription.

TABLE V: Performance Comparison with Singh *et al.*'s Scheme

Scheme		Key generation	Encryption cost	Decryption cost
Singh *et al.*	CP-ABE	0.252 ms	33.728 ms	101.054 ms
	KP-ABE	0.08 ms	0.123 ms	67.43 ms
The MQHIBE scheme		0.214 ms	0.085 ms	67.329 ms

TABLE VI: Performance Comparison with MQTT using TLS

	Standard MQTT with TLS protocol	The proposed MQHIBE scheme
Preparation	$T_{ECDHE} + T_{RSA_{Enc}} + T_{RSA_{Dec}} + 17T_{h_{384}}$ $\approx 62.972 + 2.903 + 109.462 + 0.017$ ms ≈ 175.354 ms	$T_s + T_a + 3 \cdot T_s + 3 \cdot T_a + T_s + T_s$ $\approx 0.114 + 0.1$ ≈ 0.214 ms
Encryption cost	$T_{AES-GCM_{Enc}} + T_{h_{384}}$ $\approx 0.003 + 0.001$ ms ≈ 0.004 ms	$T_a + 3T_s + T_{AES-GCM_{Enc}}$ $\approx 0.025 + 0.057 + 0.003$ ms ≈ 0.085 ms
Decryption cost	$T_{AES-GCM_{Dec}} + T_{h_{384}}$ $\approx 0.231 + 0.001$ ms ≈ 0.232 ms	$2T_p + 2T_a + T_{AES-GCM_{Dec}}$ $\approx 67.048 + 0.05 + 0.231$ ms ≈ 67.329 ms
Total cost	≈ 175.59 ms	≈ 67.628 ms

With the advantages mentioned before, the proposed MQHIBE scheme is suitable for MQTT environment and guarantees secure message transmission. In the future, how to achieve provably security will be a further study. In addition, the quality of service and quality of message transmission (such as data recovery) in MQTT are an open challenges to be investigated in the near future.

REFERENCES

[1] S. O'Dea. Global industrial/consumer IoT connected objects 2018-2025. Statista Ltd. Feb 27, 2020. https://www.statista.com/statistics/976079/number-of-iot-connected-objects-worldwide-by-type/.

[2] I. E. T. F. (IETF), "The constrained application protocol (coap)," Internet Engineering Task Force (IETF), Tech. Rep., 2014.

[3] ——, "Extensible messaging and presence protocol (xmpp): Core," Internet Engineering Task Force (IETF), Tech. Rep., 2011.

[4] OASIS, "Mqtt version 5.0," OASIS, Tech. Rep., 2019.

[5] ——, "Mqtt version 3.1.1." OASIS, Tech. Rep., 2014.

[6] S. P. Mathews and R. R. Gondkar, "Protocol recommendation for message encryption in MQTT," in *2019 International Conference on Data Science and Communication (IconDSC)*, 2019, pp. 1–5.

[7] O. Sadio, I. Ngom, and C. Lishou, "Lightweight security scheme for mqtt/mqtt-sn protocol," in *2019 Sixth International Conference on Internet of Things: Systems, Management and Security (IOTSMS)*, 2019, pp. 119–123.

[8] I. E. T. F. (IETF), "The transport layer security (tls) protocol version 1.2," Internet Engineering Task Force (IETF), Tech. Rep., 2008.

[9] ——, "The transport layer security (tls) protocol version 1.3," Internet Engineering Task Force (IETF), Tech. Rep., 2018.

[10] M. Singh, M. A. Rajan, V. L. Shivraj, and P. Balamuralidhar, "Secure mqtt for internet of things (iot)," in *2015 Fifth International Conference on Communication Systems and Network Technologies*, 2015, pp. 746–751.

[11] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," 2006, pp. 89–98.

[12] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *2007 IEEE Symposium on Security and Privacy (SP '07)*, 2007, pp. 321–334.

[13] X. Wang, J. Zhang, E. M. Schooler, and M. Ion, "Performance evaluation of attribute-based encryption: Toward data privacy in the iot," in *2014 IEEE International Conference on Communications (ICC)*, 2014, pp. 725–730.

[14] I. E. T. F. (IETF), "Transmission control protocol," Internet Engineering Task Force (IETF), Tech. Rep., 1981.

[15] J. Horwitz and B. Lynn, "Toward hierarchical identity-based encryption," in *Advances in Cryptology — EUROCRYPT 2002*, L. R. Knudsen, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, pp. 466–481.

[16] D. Boneh and M. Franklin, "Identity-based encryption from the Weil pairing," in *Advances in Cryptology — CRYPTO 2001*, J. Kilian, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 213–229.

[17] D. Boneh, X. Boyen, and E.-J. Goh, "Hierarchical identity based encryption with constant size ciphertext," in *Advances in Cryptology – EUROCRYPT 2005*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 440–456.

[18] J. H. Seo and K. Emura, "Revocable hierarchical identity-based encryption: History-free update, security against insiders, and short ciphertexts," in *Topics in Cryptology — CT-RSA 2015*, K. Nyberg, Ed. Cham: Springer International Publishing, 2015, pp. 106–123.

[19] Python Package Index (PyPI). https://pypi.org/.

[20] The Python Standard Library. https://docs.python.org/3/library/.

Chun-I Fan received the M.S. degree in computer science and information engineering from National Chiao Tung University, Hsinchu, Taiwan, in 1993, and the Ph.D. degree in electrical engineering from National Taiwan University, Taipei, Taiwan, in 1998. From 1999 to 2003, he was an Associate Researcher and a Project Leader with Telecommunication Laboratories, Chunghwa Telecom Company, Ltd., Taoyuan, Taiwan. In 2003, he joined as a faculty with the Department of Computer Science and Engineering, National Sun Yat-sen University (NSYSU), Kaohsiung, Taiwan. He has been a Full Professor since 2010 and a Distinguished Professor since 2019. He also is the Dean of College of Engineering and the Director of Information Security Research Center at NSYSU, and he was the CEO of "Aim for the Top University Plan" Office, NSYSU. And he is currently an outstanding fac- ulty in Academic Research in NSYSU. His current research interests include applied cryptology, information security, and communication security. He received the Best Student Paper Awards from the National Conference on Information Security in 1998, the Dragon Ph.D. Thesis Award from Acer Foundation, the Best Ph.D. Thesis Award from the Institute of Information and Computing Machinery in 1999, and the Y. Z. Hsu Science Paper Award (Information and Communication Science and Technology Category) in 2020. He won the Engineering Professors Award from Chinese Institute of Engineers — Kaohsiung Chapter in 2016, and the Outstanding Technical Achievement Award from IEEE Tainan Section in 2020. He is the Chairman of Chinese Cryptology and Information Security Association, and was the Chief Executive Officer of Telecom Technology Center in Taiwan.

Cheng-Han Shie was born in Kaohsiung. He is currently studying for a doctorate in computer science from National Sun Yet-sen University, Kaohsiung, Taiwan. His research interests include network security, software-defined network security, AI security, information security, cryptographic protocols, and applied cryptography.

Yi-Fan Tseng was born in Kaohsiung, Taiwan. He received the Ph.D. degree and MS degree in computer science and engineering from National Sun Yat-sen University, Taiwan, in 2014 and 2018, respectively. From 2018 to 2019, as a postdoctoral researcher, he joined the research group of Taiwan Information Security Center at National Sun Yat-sen University (TWISC@NSYSU). In 2019, he has joined the faculty of the Department of Computer Science, National Chengchi University, Taipei, Taiwan. His research interests include cloud computing and security, network and communication security, information security, cryptographic protocols, and applied cryptography.

Hui-Chun Huang was born in Taoyuan, Taiwan. She received the B.S. degree in Information Management at Chang Gung University, Taoyuan, Taiwan, in 2017, and the M.S. degree in Computer Science and Engineering at National Sun Yat-sen University in 2020. Her current research intrests include applied cryptography and IoT security.